PharmaSUG 2012 - Paper DG11

Gilding the Lily: Boutique Programming in TAGSETS.RTF

Rohini Rao, Omeros Corporation, Seattle, WA
Paul Hamilton, Omeros Corporation, Seattle, WA

ABSTRACT

PROC REPORT has long been a powerful and flexible tool in clinical programming for producing tables and listings. The ODS TAGSETS.RTF destination was introduced in SAS version 9.2, enabling programmers to produce higher quality output with less effort. This paper demonstrates a few tips and tricks to subtly enhance the æsthetics of the reports.

Topics covered include: the use of decimal tabs to properly align numerical output; the usage of Unicode to insert special characters; the deletion of excess blank lines at the top and bottom of a page; the insertion of page-specific footnotes; varying the number of lines per page to improve the flow of output; and the repetition of a header row when the body of the section spans multiple pages.

SAS version 9.2 on the Windows platform is used in this paper. It is catered towards an audience with an intermediate level of SAS and ODS knowledge.

INTRODUCTION

"Gilding the Lily" is a phrase first used by Shakespeare, meaning to embellish something that really does not need much improvement. In our paper, we look at techniques to embellish the appearance of our reports. We start with tables and listings that are functional and deliver accurate content and then add 'boutique' touches to enhance their appearance. These techniques allow us to have more control over our output and give us the luxury of crafting more user friendly reports.

DECIMAL TABS

Reports in the pharmaceutical industry contain mostly numerical output. Using decimal tabs allows us to properly align numerical output and improve readability. We first look at a sample table created without decimal tabs and revisit it after adding the decimal tabs code.

Table 1.1 is a section of a sample vital signs table *without* decimal tabs:

Table 1.1: Vital Signs

	Placebo (N=200)	**Active (N=200)**
Diastolic BP (mmHg)		
Baseline		
N	200	200
Mean(SD)	73.8 (10.3)	74.0 (10.7)
Median	73.0	74.0
Min,Max	43, 107	49, 103
5 Minutes from Start of Procedure		
N	200	200
Mean(SD)	73.1 (10.3)	73.1 (10.3)
Median	72.0	75.0
Min,Max	43, 107	49, 103

Table 1.2 is the same table *with* decimal tabs:

Table 1.2: Vital Signs

	Placebo (N=200)	Active (N=200)
Diastolic BP (mmHg)		
Baseline		
N	200	200
Mean(SD)	73.8 (10.3)	74.0 (10.7)
Median	73.0	74.0
Min,Max	43, 107	49, 103
5 Minutes from Start of Procedure		
N	200	200
Mean(SD)	73.1 (10.3)	73.1 (10.3)
Median	72.0	75.0
Min,Max	43, 107	49, 103

SAS styles provide the ability to specify decimal tabs, However they are not as flexible or attractive as what the native RTF codes allow. The RTF code is embedded within PROC REPORT (displayed in red). Col0 is the vital signs category, Col1 is the placebo drug and Col2 is the active drug.

The required syntax to specify a decimal tab is: `pretext = "^R'\ql\tqdec\txNNN '".` The '`ql`' code turns on left justification, the '`tqdec`' code specifies a decimal tab and the '`tx`*NNN*' code determines where the tab is placed within the cell. *NNN* is a number supplied by the SAS programmer and the unit for this number is a twip. A twip is defined as 1/1440 of an inch and in our example, NNN=650 twips. This technique is equally useful in Listing reports, especially Vital Signs or Laboratory data values.

```
Options nodate nonumber nobyline orientation = landscape missing = ' ';
ods escapechar='^';

ods listing close;
ods tagsets.omsrtf file = "&outpath.\&pgmout" pagePanels = none uniform;

proc report data = master missing nowindows split = '|' spanrows;
  columns pagenum panel col0 col1 col2;
  define pagenum / order noprint;
  define panel / order noprint;
  define col0/ display  " "
                        style(column) = [cellWidth = 3.5in
                                         just = left];
  define col1/ display  "Placebo|(N=&col1)"
                        style(column) = [cellWidth = 1.25in
                                         just = left
                                         pretext = "^R'\ql\tqdec\tx650 '"];
  define col2/ display  "Active|(N=&col2)"
                        style(column) = [cellWidth = 1.25in
                                         just = left
                                         pretext = "^R'\ql\tqdec\tx650 '"];
  compute after panel;
    line@1 '';
  endcomp;
run;

ods tagsets.omsrtf close;
ods listing;
```

INSERTION OF SPECIAL CHARACTERS

The usage of special characters in a report is a perennial problem. The SAS community has been able to produce these characters in the original RTF destination, but with more effort and difficult code. SAS version 9.2 provides direct support of UNICODE characters.

Table 2.1 is a sample table from a clinical trial of a drug to prevent urinary incontinence. A standard way to display the number of urinary incontinence episodes is as follows: (a)<2; (b)>=2 to <5 and so on. However, we can use Unicode characters to display >=2 as ≥2. We also look at RTF code to display the superscript characters that appear in the table.

Table 2.1: Supportive Analysis of Primary Endpoint

	Placebo (N=200)	Active (N=200)
Subjects with incontinent episodes per day		
<2	43	73
≥2 to <5	41	30
…		
≥10	33	22
p-value[a]		0.0316

[a] Chi-square test

The categories are displayed by embedding Unicode in our SAS code:

```
Proc format;
  value pan1stat
     0 - 2      =    "<2"
     2 - < 5 = "^{UNICODE 2265}2 to <5"
     …
     10 - 100   = "^{UNICODE 2265}10";

  value pan2stat
    1 = "p-value ^R'\super ' a";
run;

data pan1B;
  set pan1;
  uiCat=put(uiNum,pan1stat.);
run;

proc freq data=pan1B noprint;
  by trt01p;
  table uiCat/out=pan1F (drop=percent);
run;

proc sort data=pan1F; by uiCat; run;

proc transpose data=pan1F out=pan1T (drop=_NAME_ _LABEL_);
  by uiCat;
  var count;
  id trt01p;
run;
```

Here, the Unicode number to display the '≥' symbol is 2265. Defining the Unicode symbol in our proc format will yield the results we want.

As well, in our example, the RTF code to display the footnote character 'a' in superscript font is included in the `pan2stat` format as follows: `^R'\super ' a`.

DELETION OF EXTRA BLANK LINES AND PAGE SPECIFIC FOOTNOTES

Our example here is the demographics table from the same clinical trial. Col0 is the demographic category, Col1 is the placebo drug, Col2 is the clinical trial drug and Col3 is the total. The categories, or the 'panels' of data are separated by one blank line. One method of inserting a blank line after a panel of data is displayed in red below:

```
options nodate nonumber nobyline orientation = landscape missing = ' ';
ods escapechar='^';

ods listing close;
ods tagsets.omsrtf file = "&outpath.\&pgmout" pagePanels = none uniform;

proc report data = master missing nowindows split = '|';
  columns pagenum panel col0 col1 col2 col3;
  define pagenum / order noprint;
  define panel / order noprint;
  define col0/ display  " "
                        style(column) = [cellWidth = 3in
                                         just = left];
  define col1/ display  "Placebo|(N=&col1)|n(%) "
                        style(column) = [cellWidth = 1.3in
                                         just = left
                                       pretext = "^R'\ql\tqdec\tx650 '"];
  define col2/ display  "OMS302|(N=&col2)|n(%) "
                        style(column) = [cellWidth = 1.3in
                                         just = left
                                       pretext = "^R'\ql\tqdec\tx650 '"];
  define col3/ display  "Total|(N=&col3)|n(%) "
                        style(column) = [cellWidth = 1.3in
                                         just = left
                                       pretext = "^R'\ql\tqdec\tx650 '"];
  compute after panel;
    line @ 1 ' ';
  endcomp;
run;
ods tagsets.omsrtf close;
ods listing;
```

The above method inserts a blank line after the last panel on the page as well, which is not wrong but may be deemed unnecessary. In order to make the table more æsthetically pleasing, we can omit the blank line at the bottom of each page by using the `$varying` format and the panel numbers.

```
compute after panel;
  msg=" ";
  if panel in (4,7) then len= 0;
  else                   len=10;
  line @1 msg $varying10. len;
endcomp;
```

In this particular report, panels four and seven are the last panels on the page. We suppress the output of the blank line after these panels by defining a length of zero to `len`. The length of ten assigned to the visible blank lines is arbitrary.

This logic can be applied to produce page specific footnotes as well.

```
compute after pagenum;
  msg2="This footnote will appear only on page 2";
  if pagenum=2 then len=150;
  else              len=  0;
  line @1 msg2 $varying150. len;
endcomp;
```

Table 3.1 has no extra blank line at the bottom of page one and a footnote appearing only on page two.

Table 3.1: Demographics and Subject Characteristics

	Placebo (N=200)	Active (N=200)	Total (N=400)
Age (year)			
N	200	200	400
Mean(Std)	66.7	67.0 (9.6)	69.4 (9.7)
Median	69	69	69
Min,Max	32.0, 89.0	31.0, 88.0	31.0, 89.0
Gender			
Male	22	27	49
Female	178	173	351
Ethnicity			
Hispanic Or Latino	33	17	50
Not Hispanic Or Latino	167	183	350
Race			
American Indian or Alaska Native	2	3	5
Asian	17	14	31
Black or African American	21	22	43
Native Hawaiian or Other Pacific Islander	0	0	0
White	159	161	320
Other	1	0	1

Page 2…

	Placebo (N=200)	Active (N=200)	Total (N=400)
Baseline number of incontinent episodes per day			
>2	36	36	72
≥2 to <5	105	100	205
…			
≥10	7	7	14

This footnote will appear only on page 2

CONTROLLING THE NUMBER OF LINES OF OUTPUT PER PAGE

A typical Listing report has several lines of data per subject. Often, we run into issues such as a subject having just one line of data at the bottom of a page and the remaining data on the following page. We can cluster the data more meaningfully by controlling the number of output lines per page.

```
proc format;
  value lpp
    1 = '14'
    2 = '16'
    3 = '14'
    …
    Other = ‘21’;

data master;
  format pagenum row 5.;
  set final;
  by trt site;
  if (_N_ eq 1) then do;
    pagenum = 1;
    row = 0;
  end;
  lpp = input(put(pagenum, lpp.), best.);
  row + 1;
  if ((row gt lpp) then do;
    row = 1;
    pagenum + 1;
  end;
run;
```

In Listing 4.1, we are formatting the number of lines to be fourteen on the first page in order to fit subjects 001 and 002 on the first page and force subject 003 to start on the next page.

Listing 4.1: Post-Void Residual (Page 1)

		PVR Exam		Was the PVR Obtainable?	
Site	Subject	Date	Performed?	#	If no, why?
100	001	2011-12-09	Yes	1	Yes
		2011-12-10	Yes	1	Yes
		2011-12-16	Yes	1	Yes
		2012-01-03	Yes	1	Yes
		2012-01-07	Yes	2	Yes
		2012-01-09	Yes	1	Yes
		2012-02-18	Yes	2	Yes
	002	2011-12-02	Yes	1	Yes
		2011-12-05	Yes	1	Yes
		2011-12-09	Yes	1	Yes
		2012-01-18	Yes	1	Yes
		2012-01-20	Yes	2	Yes
		2012-01-21	Yes	1	Yes
		2012-01-23	Yes	2	Yes

Listing 4.1 Continued (Page 2)

		PVR Exam			Was the PVR Obtainable?	
Site	Subject	Date	Performed?	#		If no, why?
100	003	2011-12-02	Yes	1	Yes	
		2011-12-03	Yes	1	Yes	
		2011-12-08	Yes	1	Yes	
		2011-11-29	Yes	1	Yes	
		2011-11-29	Yes	2	Yes	
		2011-12-15	Yes	1	Yes	
		2011-12-15	Yes	2	Yes	

REPETITION OF A HEADER ROW WHEN DATA SPAN MULTIPLE PAGES

An adverse event table often has data displayed for each Preferred Term under a particular System Organ Classification. Some SOC groups have several Preferred Terms, all of which do not fit on one page. In such cases, where the Preferred Terms span multiple pages, it is useful to have the SOC name repeat on the first row of each page. In our example, we look at subject incidence of treatment emergent adverse events.

Here, the 'Renal and Urinary Disorders' SOC has several Preferred Terms on both pages one and two. We want *'Renal and Urinary Disorders (continued)'* to appear on the top of page two. In the master data set, we know that the last line of data on page one is on row twenty. In the master2 data set, we output every record of the master data set first and then output '*Renal and Urinary Disorders (continued)*' on row twenty and a half, which is the first line on page two.

```
data master2;
  set master;
  output;
  *Row = 0 when the SOC name is first written. The SOC name is repeated on row 20.5;
  if panel = 1 and row = 0 then do;
     row = 20.5;
     col0 = strip(tranwrd(col0, '\b', '\b\i')) || ' (continued)';
     call missing(col1, col2, col3, col4);
    output;
  end;
run;

proc sort data=master2;
  by panel row;
run;

options nodate nonumber nobyline orientation = landscape missing = ' ';
ods escapechar='^';
ods listing close;
ods tagsets.omsrtf file = "&outpath.\&pgmout" pagePanels = none uniform;

proc report data = master2 missing nowindows split = '|';
  columns panel row col0 col1 col2;
  define panel / order noprint;
  define row / order noprint;
  define col0/ display  " "
                        style(column) = [cellWidth = 3.5in
                                         just = left];
  define col1/ display  "Vehicle|(N=&col1)"
                        style(column) = [cellWidth = 1.25in
                                         just = left
                                           pretext = "^R'\ql\tqdec\tx650 '"];
```

```
  define col2/ display  "Active|(N=&col2)"
                        style(column) = [cellWidth = 1.25in
                                         just = left
                                         pretext = "^R'\ql\tqdec\tx650 '"];
run;

ods tagsets.omsrtf close;
ods listing;
```

In table 5.1, the Preferred Terms for Renal and Urinary Disorders are on pages one and two. The SOC name repeats on the top of page two.

Table 5.1: Subject Incidence of Treatment-Emergent Adverse Events by System Organ Class and Preferred Term (Page 1)

	Vehicle (N=50)	Active (N=50)
Any Event	47	36
Renal and Urinary Disorders	3	1
Allergic Cystitis	2	1
Anuria	2	1
Bacterial pyelonephritis	7	5
Bladder candidiasis	9	5
Bladder Spasm	3	1
Calculus urinary	1	1
Dysuria	3	0
Fluid retention	12	9
Fungal cystitis	3	1
Haematuria	4	2
Haemorrhage, urinary tract	1	0
Nephrolithiasis	7	2
Pyrexia	7	6

Table 5.1 (Page 2)

	Vehicle (N=50)	Active (N=50)
Renal and Urinary Disorders (continued)		
Renal haematoma	1	0
Renal impairment	5	2
Renal syphilis	3	1
Renal tuberculosis	2	1
Strangury	1	0
Trigonitis	1	0
Urethral pain	2	4
Urinary retention	15	11
Urine flow decreased	7	3

CONCLUSION

We have compiled a list of boutique programming techniques that may be used to enhance the appearance of clinical data reports. We have introduced decimal tabbing with in-line styling to vertically align numerical output. As a result, the user is able to easily recognize the varying magnitude of numbers within a cell. We have demonstrated the ease of UNICODE usage in inserting special characters in a report. The `$varying` format can be applied in the `line` statement to not only remove excess blank lines, but also to insert page specific footnotes. In our opinion, the latter

method is a simple yet elegant usage of the `$varying` format. We have also discussed the technique of varying the number of lines per page. The user can hopefully employ this trick to more meaningfully cluster data on a page. Last but not least, we have shown a way of repeating a header row at the top of relevant pages, when a block of data spans multiple pages. The intended goal here is to avoid requiring the user to flip pages back and forth while reading the report. This list is by no means intended to be comprehensive. It is merely a discussion of a few methods of 'gilding our lilies'.

APPENDIX

The code that creates `tagsets.omsrtf` is below:

```
proc template;
   define tagset Tagsets.Omsrtf;
      notes "This is the RTF_SAMPLE tagset which is inherited from rtf.tpl";
      define event realraw;
         start:
            put value /if value;
      end;
      default_style = "styles.rtf";
      parent = tagsets.rtf;
      uniform;
   end;
run;
```

REFERENCES

Hamilton, Paul. "ODS to RTF: Tips and Tricks". Available at: http://www2.sas.com/proceedings/sugi28/024-28.pdf

Rashleigh-Berry, Roland. ""Roland's SAS Macros". Available at: http://www.datasavantconsulting.com/roland/condline.html.

CONTACT INFORMATION

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

Name: Rohini Rao and Paul Hamilton
Enterprise: Omeros Corporation
Address: 1420 Fifth Avenue, Suite 2600
City, State ZIP: Seattle, WA 98101
Work Phone: (206) 676-5000
E-mail: rrao@omeros.com; phamilton@omeros.com

The content of this paper does not include actual clinical trial data. The opinions expressed herein are those of the authors and not necessarily those of Omeros Corporation.