Paper 042-30

Make the Invisible Visible – A Case Study of Using ODS Inline Formatting Style

Zizhong Fan, Westat, Rockville, MD

ABSTRACT

Starting from Version 8.2, SAS® supports the ability to insert simple formatting text into ODS output, which is called Inline Formatting. This is accomplished by using the statement ODS ESCAPECHAR='*escape-character*'. This paper will use a practical example to show the pros and cons of using Inline Formatting Style in DATA steps and STYLE= options in PROC steps. SAS product: Base SAS. Skill level: Intermediate or advanced.

INTRODUCTION

ODS (Output Delivery System) output can be conveniently created by the default templates that SAS has provided. ODS output can also be customized by using different tools such as creating style sheets by using PROC TEMPLATE or simply using STYLE= options in PROC steps. One of the new features that have been introduced in version 8.2 is Inline Formatting. It allows us to insert simple formatting text into ODS output. The types of formatting that are available include: specifying a style option to customize the font, color, etc; specifying a superscript and subscript; specifying a dagger or sigma character; specifying raw text to be inserted into the document. This paper will use a real world example to demonstrate the pros and cons of using Inline Formatting Style in DATA steps and STYLE= options in PROC steps.

EXAMPLE AND SOLUTION

This example is a simulation of the Center for Medicare & Medicaid Services (CMS) Quality Improvement Organizations Customer Satisfaction Survey. The data values have been altered for confidentiality. The data values are only for the purpose of this paper and do not represent any factual data values and rates.

Part of the data set Response is shown below. It is a summary data set containing survey response rates. There are 5 variables: State, ProviderType, NoResponse (number of responses), SampSize (sample size), RRate (response rate, which is derived as RRate =NoResponse/SampSize). There are 16 states and 6 provider types for each state in this example data set.

Part of data set Response

State	ProviderType	NoResponse	SampSize	RRate
AZ	1	38	40	0.95
AZ	2	39	42	0.93
AZ	3	48	50	0.96
AZ	4	39	42	0.93
AZ	5	48	50	0.96
AZ	6	38	40	0.95
CA	1	37	40	0.93
CA	2	45	45	1.00
CA	3	45	50	0.90
CA	4	45	45	1.00
CA	5	45	50	0.90
CA	6	37	40	0.93
CO	1	38	40	0.95
CO	2	41	43	0.95
CO	3	12	25	0.48
CO	4	41	43	0.95
CO	5	35	40	0.88
CO	6	38	40	0.95
DC	1	40	45	0.89
DC	2	15	15	1.00

Our task here is to present the response rate in a PDF file. One of the requirements is that both the response rate (RRate) and sample size (SampSize) be flagged as follows:

1. If the response rate (RRate) is less than 60%, show the rate figure in red font.
2. If the sample size is less than 40, shade the background with yellow.

This can be conveniently accomplished by using the STYLE= options in the two compute blocks in Sample Code 1.

Sample Code 1

```
title "Response Rate by State and Provider Type";
ods pdf body="Response.pdf" notoc;
proc report data=Rate nowd;
 column State ProviderType RRate SampSize;
 define State/order;
 define ProviderType/order center;
 define RRate/"Response Rate" format=percent5.;
 define SampSize/"Sample Size" noprint;
  compute RRate;
   if RRate.sum < .6 then
    call define(_col_, 'style', 'style=[foreground=red]');
  endcomp;
  compute SampSize;
   if SampSize.sum < 40 then
    call define(_row_, 'style', 'style=[background=yellow]');
  endcomp;
run;
ods pdf close;
```

Part of the report is shown in table 1. For example, CO Provider Type 3 is flagged in red font and yellow background because its response rate is below 60% and its sample size is less than 40. DC Provider Type 2 is flagged by yellow background because its sample size is below 40, but its response rate is 100% so that the foreground is not flagged in red font.

Table 1: Output of Sample Code 1
Response Rate by State and Provider Type

State	ProviderType	Response Rate
AZ	1	95%
	2	93%
	3	96%
	4	93%
	5	96%
	6	95%
CA	1	93%
	2	100%
	3	90%
	4	100%
	5	90%
	6	93%
CO	1	95%
	2	95%
	3	48%
	4	95%

State	ProviderType	Response Rate
	5	88%
	6	95%
DC	1	89%
	2	100%

The problem is that this report might not satisfy our client because it is a vertically long report and wastes a lot of space horizontally. It is a table consisting of 3 columns but 96 rows. So our client comes back to us and requests a report in a better format shown in table 2, in which there are 7 columns and 16 rows and all the States and Provider Types are on the same page.

Table 2: Output of Sample Code 2
Response Rate by State and Provider Type

State	ProviderType1	ProviderType2	ProviderType3	ProviderType4	ProviderType5	ProviderType6
AZ	95%	93%	96%	93%	96%	95%
CA	93%	100%	90%	100%	90%	93%
CO	95%	95%	48%	95%	88%	95%
DC	89%	100%	45%	96%	83%	89%
FL	95%	91%	92%	95%	92%	91%
HI	75%	97%	20%	97%	92%	75%
MA	50%	25%	33%	88%	93%	95%
MD	100%	56%	50%	94%	100%	100%
NC	95%	90%	97%	93%	97%	96%
NJ	96%	95%	58%	95%	63%	96%
NY	90%	92%	72%	92%	72%	90%
OH	87%	91%	82%	91%	82%	87%
PA	89%	91%	50%	100%	93%	89%
SC	95%	83%	78%	89%	90%	95%
VA	40%	90%	20%	55%	90%	90%
WV	93%	100%	83%	90%	83%	93%

In order to produce this report, we first rearrange the data set by using the TRANSPOSE procedure:

```
proc transpose data=Response out=Trans (drop=_name_) prefix=ProviderType;
  by state ;
  id ProviderType;
  var rrate;
run;
```

Data set Trans (no Inline Formatting)

State	ProviderType1	ProviderType2	ProviderType3	ProviderType4	ProviderType5	ProviderType6
AZ	0.95	0.93	0.96	0.93	0.96	0.95
CA	0.93	1.00	0.90	1.00	0.90	0.93
CO	0.95	0.95	0.48	0.95	0.88	0.95
DC	0.89	1.00	0.45	0.96	0.83	0.89
FL	0.95	0.91	0.92	0.95	0.92	0.91
HI	0.75	0.97	0.20	0.97	0.92	0.75
MA	0.50	0.25	0.33	0.88	0.93	0.95
MD	1.00	0.56	0.50	0.94	1.00	1.00
NC	0.95	0.90	0.97	0.93	0.97	0.96
NJ	0.96	0.95	0.58	0.95	0.63	0.96
NY	0.90	0.92	0.72	0.92	0.72	0.90
OH	0.87	0.91	0.82	0.91	0.82	0.87
PA	0.89	0.91	0.50	1.00	0.93	0.89
SC	0.95	0.83	0.78	0.89	0.90	0.95
VA	0.40	0.90	0.20	0.55	0.90	0.90
WV	0.93	1.00	0.83	0.90	0.83	0.93

Then we might want to use a program similar to sample code 1 to generate the report. But you may have realized that we already lost one of the flags: SampSize, which we need to produce the color coding background in the report. In sample code 1, the yellow background flag relied on the variable SampSize in the same row with RRate. Now the SampSize is ***invisible***. This is where the ODS Inline Formatting can help us out to make the invisible ***visible***.

The key issue here is to insert the formatting **before** we lose the flag variable SampSize. Before transposing the data, we can use the following code (Sample Code 2-1) to insert the Inline Formatting Style.

Sample Code 2-1

```
data Response;
 length Rate $75 Ratio $4;
 set Response;
 ratio=trim(put(RRate*100, 3.))||"%";
 if RRate  <.6 then do;
  if SampSize>=40 then
   Rate="~S={foreground=red}"||left(ratio); else
  if SampSize<40 then Rate="~S={background=yellow foreground=red}"||left(ratio);
 end; else
 if RRate>=.6 then do;
  if SampSize>=40 then Rate=left(ratio); else
  if SampSize <40 then
     Rate="~S={background=yellow}"||left(ratio);
  end;
  drop ratio;
run;
```

Then we can transpose the data set ***with*** the Inline Formatting (Sample Code2-2).

Sample Code 2-2

```
proc transpose data=Response out=Trans prefix=ProviderType (drop=_name_);
  by state ;
  id ProviderType;
  var rate;
run;
```

Part of the data set Trans (with inline formatting)

State	ProviderType1	ProviderType2
AZ	95%	93%
CA	93%	100%
CO	95%	95%
DC	89%	~S={background=yellow} 100%
FL	~S={background=yellow} 95%	91%
HI	75%	97%
MA	~S={background=yellow foreground=red} 50%	~S={foreground=red} 25%
MD	100%	~S={background=yellow foreground=red} 56%
NC	~S={background=yellow} 95%	90%
NJ	96%	95%
NY	90%	92%
OH	87%	91%
PA	89%	91%
SC	95%	~S={background=yellow} 83%
VA	~S={foreground=red} 40%	90%
WV	93%	~S={background=yellow} 100%

We can see here that both the font and background flags are in the appropriate cells. Either a PROC REPORT or PRINT (Sample Code 2-3) can finish the rest of the job. Here we use PROC PRINT. The output is shown in table 2. ODS ESCAPECHAR='~'; has to be specified in order for the PROC step to recognize the Inline Formatting Styles imbedded in the cells. It specifies the escape character, in our example '~'. S={} contains the styles. Note that the compute blocks with STYLE= options are not needed here because the formatting has been pre-inserted. Now the invisible sample size flag becomes visible in the report shown in table 2.

Sample Code 2-3

```
ods escapechar='~';
title "Response Rate by State and Provider Type";
ods pdf file="Response.pdf" notoc;
proc print data=Trans noobs;
  var State ProviderType1 ProviderType2 ProviderType3
            ProviderType4 ProviderType5 ProviderType6;
run;
ods pdf close;
```

DISCUSSION

When Inline Formatting Style is used in DATA steps, it serves the same way as STYLE= options in PROC steps (e.g. PROC REPORT, PROC PRINT, and PROC TABULATE). Some of the pros and cons are discussed here.

Pros:

1. In situations where flagging is based on certain variables, and the those variables cannot be included in the report data set (as the example shown in this paper), Inline Formatting may be the most convenient method to insert the flags into the cells before losing the flagging information.

2. When using Inline Formatting in a DATA step, there are more choices for output procedures. For the example of this paper, PROC PRINT, PROC REPORT, or PROC SQL can all be used if using Inline Formatting Style. Whereas, PROC REPORT might be the most conveniently available, if not the only, choice when using STYLE= options, since compute blocks are needed.

3. When a group of variables need a same STYLE specification, Inline Formatting may make programming less complex. For example, if we only want to flag the font of RRate in red font in table 2 layout, PROC REPORT would require 6 compute blocks with the same STYLE= option for the 6 Provider Type columns. Or a macro do-loop needs to be constructed. Inline formatting would need arrays in the data step to put the same style option into the appropriate cells. If the same

STYLE options will be used multiple times in multiple procedures, Inline formatting may save programming (cut-and-paste) time. The comparison is coded as below.

```
****Multiple compute blocks with the same style option****;
%macro rpt;
ods rtf body="Response.rtf";
proc report data=Trans nowd;
 columns state ProviderType1-ProviderType6;
 define state/order;
 %do i=1 %to 6;
 define ProviderType&i/format=percent5.;
 compute ProviderType&i;
   if ProviderType&i..sum < .6 then
    call define(_col_, 'style', 'style=[foreground=red]');
 endcomp;
 %end;
run;
ods rtf close;
%mend;

%rpt

***Applying same style option to multiple variables using arrays***;
data Trans;
 set Trans;
 array PType    {*}   ProviderType1 – ProviderType6;
 array Provider {*} $ Provider1  – Provider6;
  do i=1 to dim(Provider);
  if PType(i) <.6 then
   Provider(i)="~S={foreground=red}"|| trim(put(PType(i)*100, 3.))||"%";
    else Provider(i)=trim(put(PType(i)*100, 3.))||"%";
  end;
  drop i ProviderType1-ProviderType6;
run;
```

Cons:

1. When using Inline Formatting Style in a DATA step, the variables have to be character values or changed to be character values in order for the style options to be inserted.

2. If only one single style option is needed or when different style options apply to different variables, using STYLE= options in a PROC step may be a better choice because Inline Formatting Style requires the extra DATA step. In other words, one step has better performance than two steps.

REFERENCES

McNeill, Sandy. *Changes & Enhancements for ODS by Example (through V8.2)* http://www2.sas.com/proceedings/sugi26/p002-26.pdf.

Schellenberger, Brian. *Presentation-Quality Tabular Output via ODS* http://support.sas.com/rnd/base/topics/odsprinter/qual.pdf.

Center for Medicare & Medicaid Services (CMS). *Quality Improvement Organizations Customer Satisfaction Survey Round 1 Report, June 2004.*

ACKNOWLEDGEMENTS

I would like to thank Duke Owen and Mike Rhoads for reviewing and giving advice on this paper.

CONTACT INFORMATION

Zizhong Fan
Westat
1650 Research Boulevard
Rockville, MD 20850
(240) 314 -2486
E-mail: JamesFan@westat.com

DISCLAIMER

The contents of this paper are the work of the author and do not necessarily represent the opinions, recommendations, or practices of Westat.