Paper RF-07-2014

Color Me Impressed: Creating Colorful PROC REPORT Output

Erica Goodrich, Priority Health, Grand Rapids, MI
Daniel Sturgeon, Priority Health, Grand Rapids, MI

ABSTRACT

In some instances, we need to highlight our data for reporting to bring attention to our readers. One of the easiest ways to do this is through colors in your report. This presentation will discuss ways and uses of PROC REPORT to highlight cell colors for individual cells, rows, columns, and even diagonal reporting leveraging the CALL DEFINE statement and STYLE options.

INTRODUCTION

My first major experience with PROC REPORT outside of school was when I held a Biostatistics internship at a local CRO. These reports were always very straight to the point and reported in black and white and similar to tables seen in research literature. My second experience was working with grant funded chronic disease programs in a local hospital. One of my tasks during this internship was to create a report for a transition matrix, and then color the diagonal cells of the matrix. The cells on these diagonals were in respect to one another so this colorful representation made it easier to present the graph to outside sources. At the time of working through this idea, it was rather challenging and not heavily documented using two variables to create a matrix style PROC REPORT using colors. This paper will discuss SAS logic to color rows, columns, and cells for reporting and a finalized version of the transition matrix. In many common instances on the topic, reports will pull from many different variables, which often only represent one row or one column, however that is not the case in this paper. This instance is a bit unique in the fact that it focuses on working with only two variables for the entire PROC REPORT.

THE BASICS

This paper assumes that the reader has a baseline understanding of PROC REPORT and the syntax needed for it. For the sake of this paper and the nature of working with matrix reporting in PROC REPORT, a dataset containing only two variables is used. This dataset is simulated data based on hospital admissions and readmissions for a group of patients who participated in a program to help manage their health while living with a chronic disease. The two variables are admission and readmission. Admission is a categorized time frame (in months) in which patient is admitted to a hospital after joining the program. Readmission is a time frame (in months) in which patient is readmitted to a hospital after an original admission to the hospital after joining the program. There could be some interpretation made from this matrix to show that if a patient had the first admission to a hospital in relation to a readmission to the hospital. The dataset is at the end up the paper under "syntax" which also includes PROC FORMAT used to define each of the time periods. The basic PROC REPORT syntax used for creating this matrix is as follows:

```
PROC REPORT DATA=hosp_admit NOWD COMPLETEROWS COMPLETECOLS MISSING SPLIT='*';
    COLUMN admission readmission;
    DEFINE admission / '' GROUP FORMAT=hospf. PRELOADFMT ORDER=data
    STYLE(column)=[just=c width=.6in];
    DEFINE readmission / '' ACROSS FORMAT=readmitf. PRELOADFMT ORDER=data
    STYLE(column)=[just=c width=.8in];
RUN;
QUIT;
```

A few extra commands were added in this PROC REPORT. The first worth pointing out is the COMPLETEROWS and COMPLETECOLS options in the procedure line which works with the PRELOADFMT line in each of the define statements. When these are used, all values in the PROC FORMAT will show up as a row or column even if there are no values needed for them. In this instance, the dataset is rather sparse so it is an easy way to show all values needed in the matrix. A final note: All graphs are being shown from rtf output using the SAS style Meadows.

	Readmitted within 30	Readmitted between 31 and 60	Readmitted between 61 and 90	Readmitted between 91 and 120	Readmitted between 121 and 150	Readmitted after 151
Hospitalized in 30 days	2	1	0	0	1	1
Hospitalized in month 1	0	1	2	0	0	0
Hospitalized in month 2	0	0	0	1	0	0
Hospitalized in month 3	0	0	0	0	0	0
Hospitalized in month 4	0	0	0	0	0	0
Hospitalized GE month 5	0	0	0	0	0	0

Figure 1. Output of the basic PROC REPORT

COLORING COLUMNS

By specifying a COMPUTE statement after the final DEFINE statement of the above PROC REPORT, once the conditional IF-THEN statement will is met, the CALL DEFINE statement will lead for the columns meeting this condition to turn yellow. It should be noted that readmission is the second variable in the COLUMN statement of the PROC REPORT. Since this is the second variable of the PROC REPORT COLUMN statement, the first _COL_ that is associated with this variable is 2. If the second column is desired (where readmissions = 1, or “Readmitted between 31 and 60 days”) then the if statement would change to “`IF _col_ = 3 THEN DO;`”. Different colors can be picked by changing the color to a SAS predefined color names or by a HEX value.

The code added and the results can be seen below:

```
COMPUTE readmission;
     IF _col_ = 2 THEN DO;
             CALL DEFINE(_col_,'style','style={background=yellow}');
     END;
ENDCOMP;
```

	Readmitted within 30	Readmitted between 31 and 60	Readmitted between 61 and 90	Readmitted between 91 and 120	Readmitted between 121 and 150	Readmitted after 151
Hospitalized in 30 days	2	1	0	0	1	1
Hospitalized in month 1	0	1	2	0	0	0
Hospitalized in month 2	0	0	0	1	0	0
Hospitalized in month 3	0	0	0	0	0	0
Hospitalized in month 4	0	0	0	0	0	0
Hospitalized GE month 5	0	0	0	0	0	0

Figure 2. PROC REPORT with column coloring

COLORING ROWS

Just like above, a COMPUTE statement using the 2nd variable (readmission) can be used to color a row. In this instance, wherever admissions are equal to 0, you would have a yellow row. This can be extended to multiple rows, if so desired. While the COMPUTE statement is using the readmission variable, the conditional statement inside of that statement is determined by the admission variable. Like defining colors of columns, multiple rows could change colors based on changing the if statement.

```
COMPUTE readmission;
    IF admission = 0 THEN DO;
        CALL DEFINE(_col_,'style','style={background=yellow}');
    END;
ENDCOMP;
```

	Readmitted within 30	Readmitted between 31 and 60	Readmitted between 61 and 90	Readmitted between 91 and 120	Readmitted between 121 and 150	Readmitted after 151
Hospitalized in 30 days	2	1	0	0	1	1
Hospitalized in month 1	0	1	2	0	0	0
Hospitalized in month 2	0	0	0	1	0	0
Hospitalized in month 3	0	0	0	0	0	0
Hospitalized in month 4	0	0	0	0	0	0
Hospitalized GE month 5	0	0	0	0	0	0

Figure 3. PROC REPORT with row coloring

COLORING CELLS

By choosing a certain cell in this situation, you would use a combination of the logic used to color columns and rows together. By combining the same logic as above, the first cell would be colored yellow.

```
COMPUTE readmission;
   IF admission = 0 and _col_ = 2 THEN DO;
       CALL DEFINE(_col_,'style','style={background=yellow}');
   END;
ENDCOMP;
```

	Readmitted within 30	Readmitted between 31 and 60	Readmitted between 61 and 90	Readmitted between 91 and 120	Readmitted between 121 and 150	Readmitted after 151
Hospitalized in 30 days	2	1	0	0	1	1
Hospitalized in month 1	0	1	2	0	0	0
Hospitalized in month 2	0	0	0	1	0	0
Hospitalized in month 3	0	0	0	0	0	0
Hospitalized in month 4	0	0	0	0	0	0
Hospitalized GE month 5	0	0	0	0	0	0

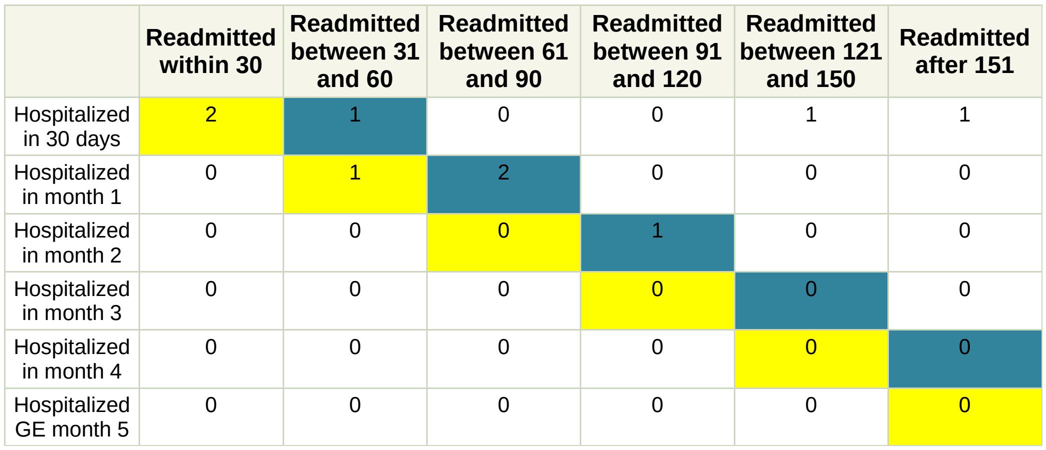
Figure 4. PROC REPORT with coloring a single cell

COLORING THE MATRIX

By using the method of selecting a cell and changing the color, you can build off of the logic and continue down an entire diagonal. After that point you can select a different color and repeat the process for as many rows and columns that are desired. In the code mentioned below, the middle diagonal would create yellow cells, with the diagonal to the right of that being colored teal. This can continue out as far as you would be interested in, but it is a monotonous process and becomes a large amount of code. This process may be justified since a desired output of a matrix using these colors may be rather small, so once the code is created, it does not need much changing.

```
COMPUTE readmission;
  /*First diagonal - yellow*/
  IF admission = 0 and _col_ = 2 THEN DO;
  CALL DEFINE(_col_,'style','style={background=yellow}');END;

        IF admission = 1 and _col_ = 3 THEN DO;
        CALL DEFINE(_col_,'style','style={background=yellow}'); END;
            IF admission = 2 and _col_ = 4 THEN DO;
            CALL DEFINE(_col_,'style','style={background=yellow}'); END;
                 /*...Continues down the matrix...*/
    /*Second diagonal - teal*/
    IF admission = 0 and _col_ = 3 THEN DO;
    CALL DEFINE(_col_,'style','style={background=teal}'); END;
         IF admission = 1 and _col_ = 4 THEN DO;
         CALL DEFINE(_col_,'style','style={background=teal}'); END;
```



	Readmitted within 30	Readmitted between 31 and 60	Readmitted between 61 and 90	Readmitted between 91 and 120	Readmitted between 121 and 150	Readmitted after 151
Hospitalized in 30 days	2	1	0	0	1	1
Hospitalized in month 1	0	1	2	0	0	0
Hospitalized in month 2	0	0	0	1	0	0
Hospitalized in month 3	0	0	0	0	0	0
Hospitalized in month 4	0	0	0	0	0	0
Hospitalized GE month 5	0	0	0	0	0	0

Figure 5. PROC REPORT with diagonal coloring with two colors

MATRIX COLORING MACRO

The above logic can be far less tedious by using the following macro. The necessary information for the macro includes 'Start' the starting value, 'Columns' the number of diagonal cells there are, and 'Color' the color you want to use on the diagonal.

```
%MACRO DIAG_COLORS(Start, Columns, Color,);
       %LET J = 0;
       %DO I = &Start %TO &Columns;
              %LET J = %SYSEVALF(&J + 1);
              IF admission = (-1 + &J) and _COL_ = (&start+ &J) THEN DO;
                     CALL DEFINE(_COL_,'style',"STYLE={background=&color}");
              END;
       %END;
%MEND;
```

The logic in the code allows the user to count cells as though row or column headers do not exist. For the first diagonal in yellow to be provided, a user would provide 1 as the Start token. Columns refers to how many columns diagonally should be colored. In the above example, 6 columns are colored yellow. Finally for color, either SAS defined colors or

hex codes can be used. Like in the example above, the word yellow would satisfy.
So by feeding the macro %DIAG_COLORS(1,6, yellow) the process is done in one line of code. This macro would be used in the COMPUTE statement just like the previous examples. While the entire macro in use can be viewed in the syntax appendix, the snippet of code would look like the following:

```
COMPUTE readmission;
    /*First diagonal - yellow*/
    %DIAG_COLORS(1,6, yellow);
ENDCOMP;
```

	Readmitted within 30	**Readmitted between 31 and 60**	**Readmitted between 61 and 90**	**Readmitted between 91 and 120**	**Readmitted between 121 and 150**	**Readmitted after 151**
Hospitalized in 30 days	2	1	0	0	1	1
Hospitalized in month 1	0	1	2	0	0	0
Hospitalized in month 2	0	0	0	1	0	0
Hospitalized in month 3	0	0	0	0	0	0
Hospitalized in month 4	0	0	0	0	0	0
Hospitalized GE month 5	0	0	0	0	0	0

Figure 6. Finalized matrix with diagonal coloring.

CONCLUSION

After a bit of experience using two variable reporting in PROC REPORT, adding colors is rather simple. This same logic can be expanded into changing fonts or font colors for varying cells with the use of CALL DEFINE and STYLE commands additionally.

REFERENCES

Defining Colors using HEX Values < http://support.sas.com/techsup/technote/ts688/ts688.html>

SYNTAX

```
/*Data set*/
DATA hosp_admit;
INPUT admission readmission;
CARDS;
0 4
0 5
2 3
1 2
0 1
0 0
1 2
1 1
0 0
;
/*Formatted values for admission and readmission variables*/
PROC FORMAT;
VALUE readmitf
        0  =    "Readmitted within 30"
        1  =    "Readmitted between 31 and 60"
        2  =    "Readmitted between 61 and 90"
        3  =    "Readmitted between 91 and 120"
        4  =    "Readmitted between 121 and 150"
        5  =    "Readmitted after 151"
```

```
;
VALUE hospf
        0  =     "Hospitalized in 30 days"
        1  =     "Hospitalized in month 1"
        2  =     "Hospitalized in month 2"
        3  =     "Hospitalized in month 3"
        4  =     "Hospitalized in month 4"
        5  =     "Hospitalized GE month 5"
;
/*Coloring macro*/

%MACRO DIAG_COLORS(Start, Columns, Color,);
%LET J = 0;
%DO I = &Start %TO &Columns;
%LET J = %SYSEVALF(&J + 1);
     IF admission = (-1 + &J) and _COL_ = (&start+ &J) THEN DO;
           CALL DEFINE(_COL_,'style',"STYLE={background=&color}");
     END;
%END;
%MEND;

/*Finalized PROC REPORT with diagonal coloring*/
ods rtf style=meadow;
PROC REPORT DATA=hosp_admit NOWD COMPLETEROWS COMPLETECOLS MISSING SPLIT='*';
    COLUMN admission readmission;
    DEFINE admission / '' GROUP FORMAT=hospf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.6in];
    DEFINE readmission / '' ACROSS FORMAT=readmitf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.8in];
RUN;
QUIT;

PROC REPORT DATA=hosp_admit NOWD COMPLETEROWS COMPLETECOLS MISSING SPLIT='*';
    COLUMN admission readmission;
    DEFINE admission / '' GROUP FORMAT=hospf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.6in];
    DEFINE readmission / '' ACROSS FORMAT=readmitf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.8in];
COMPUTE readmission;
     IF _col_ = 2 THEN DO;
CALL DEFINE(_col_,'style','style={background=yellow}');
     END;
ENDCOMP;

RUN;
QUIT;

PROC REPORT DATA=hosp_admit NOWD COMPLETEROWS COMPLETECOLS MISSING SPLIT='*';
    COLUMN admission readmission;
    DEFINE admission / '' GROUP FORMAT=hospf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.6in];
    DEFINE readmission / '' ACROSS FORMAT=readmitf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.8in];
COMPUTE readmission;
    IF admission = 0 THEN DO;
        CALL DEFINE(_col_,'style','style={background=yellow}');
    END;
ENDCOMP;

RUN;
QUIT;

PROC REPORT DATA=hosp_admit NOWD COMPLETEROWS COMPLETECOLS MISSING SPLIT='*';
```

```
    COLUMN admission readmission;
    DEFINE admission / '' GROUP FORMAT=hospf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.6in];
    DEFINE readmission / '' ACROSS FORMAT=readmitf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.8in];

     COMPUTE readmission;
     IF admission = 0 and _col_ = 2 THEN DO;
CALL DEFINE(_col_,'style','style={background=yellow}');
     END;
     ENDCOMP;

RUN;
QUIT;

PROC REPORT DATA=hosp_admit NOWD COMPLETEROWS COMPLETECOLS MISSING SPLIT='*';
    COLUMN admission readmission;
    DEFINE admission / '' GROUP FORMAT=hospf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.6in];
    DEFINE readmission / '' ACROSS FORMAT=readmitf. PRELOADFMT ORDER=data
STYLE(column)=[just=c width=.8in];
    COMPUTE readmission;
    /*First diagonal - yellow*/
    %DIAG_COLORS(1,6, yellow);
    /*Second diagonal - teal*/
    %DIAG_COLORS(2,6, teal);
    /*Third diagonal - pink*/
    %DIAG_COLORS(3,6, pink);
    /*Forth diagonal - brown*/
    %DIAG_COLORS(4,6, CXA66921);
    /*Fifth diagonal - peach*/
    %DIAG_COLORS(5,6, CXFA8072);
    /*Sixth diagonal - blue*/
    %DIAG_COLORS(6,6,light blue);
    ENDCOMP;

    RUN;
     QUIT;
ods rtf close;
```

CONTACT INFORMATION

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

Name: Erica Goodrich
Enterprise: Priority Health
Address: 1231 East Beltline Ave NE
City, State ZIP: Grand Rapids, MI 49525
Work Phone: 616-464-8142
Fax: 616-942-0024
E-mail: erica.goodrich@priorityhealth.com

Name: Daniel Sturgeon
Enterprise: Priority Health
Address: 1231 East Beltline Ave NE
City, State ZIP: Grand Rapids, MI 49525
Work Phone: 616-575-5740
Fax: 616-942-0024
E-mail: daniel.sturgeon@priorityhealth.com