Paper AD73

PROC RANK, PROC SUMMARY and PROC FORMAT Team Up and a Legend is Born!

Christianna Williams, PhD, Chapel Hill, NC

ABSTRACT

The task was to produce a figure legend that gave the quintile ranges of a continuous measure corresponding to each color on a five-color choropleth map. Actually, figure legends for several dozen maps for several dozen different continuous measures and time periods…so, the process needed to be automated. A method was devised using SAS® PROC RANK to generate the quintiles, PROC SUMMARY to get the data value ranges within each quintile, and PROC FORMAT (with the CNTLIN= option) to generate and store the legend labels. And then, of course, these were rolled into a macro to apply the method for the many different figure legends. Each part of the method is quite simple – even mundane – but together these techniques allowed us to standardize and automate an otherwise very tedious process. The same basic strategy could be used whenever one needs to dynamically generate data "buckets" but then keep track of the bucket boundaries – whether for producing labels or legends or so that future data can be benchmarked against the stored categories.

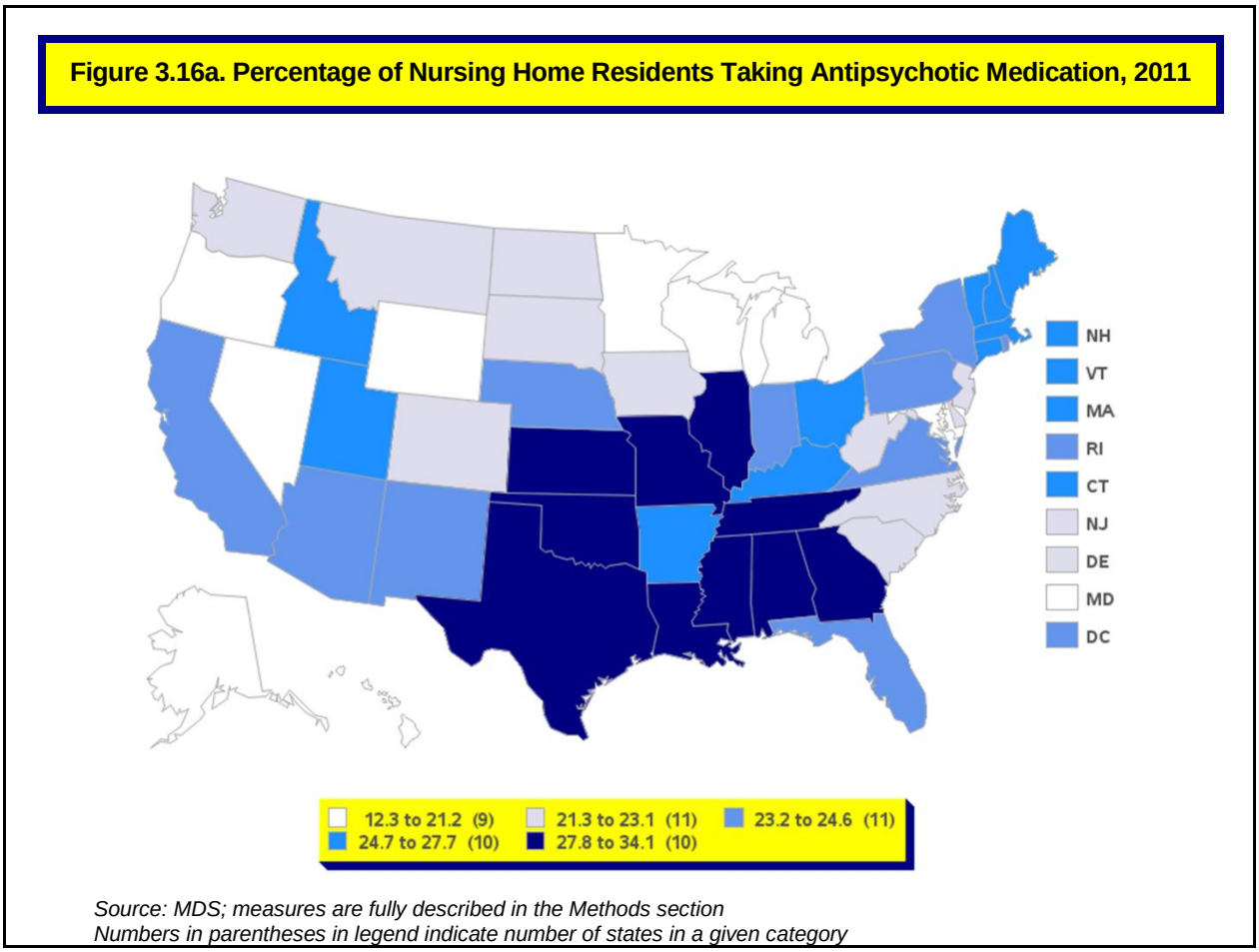

Figure 1. Example figure with legend indicating range for each of the five colors and number of states in that category

INTRODUCTION

One of my ongoing projects is to help in the annual production of a lengthy report published by the Centers for Medicare and Medicaid Services (CMS) called the Nursing Home Data Compendium (CMS, 2014). This roughly 150-page book presents descriptive summary information in the form of tables, figures and maps on all Medicare- and Medicaid-certified nursing homes in the US and the residents in these facilities. Data on nursing home characteristics (such as occupancy levels), health inspection results (such as the average number of and severity of citations for violations to health regulations) and resident characteristics (such as the percent of residents taking antipsychotic medications or reporting severe pain) are presented in summary form, generally broken down by provider characteristics that are of interest to CMS and researchers, such as for-profit vs. not-for-profit nursing homes or by state. Of course, an effective way to present geographic (in this case state-to-state) variation in some measure of interest is with a map, and the Nursing Home Data Compendium includes a lot of US maps – specifically five-color chloropleth maps (sometimes called "heat maps"), in which each state is presented in one of five colors, corresponding to quintiles of the measure being mapped. An example of such a map is shown in **Figure 1**.

The purpose of this paper is to describe in detail not how the figure itself is produced but rather how we generate the figure legends, which differ for each map, based on the range and distribution of the measure being mapped (e.g. percent of residents taking antipsychotic medications as shown in Figure 1). Also, note that the legend includes the number of states in each of the five groups,1 and I describe how that information is added to the legend data. While the basic process is very straightforward, there were a few interesting wrinkles in making the process reproducible, and in "macro-tizing" the code. Also, because I was handing the map-ready data over to another programmer to actually produce the figures, we wanted to standardize the process and the data structure, naming conventions, etc.

OVERVIEW OF THE PROCESS

The basic steps in the process are shown here. The remainder of the paper will describe each of these steps and provide the associated code.

1) Summarize individual-level data to state level, rounding to the desired degree of precision.
2) Use PROC RANK to generate quintiles.
3) Use PROC SUMMARY to obtain the upper and lower bounds of each quintile and the number of states in each range.
4) Prepare the input data set for PROC FORMAT with a DATA step.
5) Use PROC FORMAT to add FORMAT that maps the quintile value to the legend labels to a format library.
6) Generate a permanent data set containing all the information needed to produce the figure and the legend.

Steps 2 through 6 are repeated for each measure (map). So, after the code was working as desired for a single measure, a macro was built that could simply be called once for each measure. This code is also included at the end of the paper.

STEP 1. SUMMARIZE DATA TO STATE LEVEL

All the measures of interest described in this paper begin as binary (i.e. yes/no) indicators at the level of the individual nursing home resident. A federally mandated assessment tool is administered to every resident of every Medicare- or Medicaid certified nursing home on admission and on a quarterly basis, and the indicators are derived from this assessment. The resident data we process are de-identified; however, they include a facility ID that can be linked to the state where the facility is located (as well as other basic facility characteristics such as size and profit status). In order to generate state prevalence estimates for the data compendium, we simply aggregate these 0/1 measures up to the state level. It is, of course, very handy that the mean of a 0/1 measure will give a prevalence at the desired level of aggregation. Though for many applications it may be desirable to process each measure separately, here it was efficient to summarize many measures at once. I used PROC SUMMARY, which is handy here because I can very easily just use the "MEAN=" option on the OUTPUT statement to assign the original variable names to the state averages, and these new variables retain the LABELs (if any of the raw variables). The resulting data set has 51 observations, one for each state plus the District of Columbia. In the DATA step shown below I round each measure

[1] While the number of states in each quintile should be around 10, it does vary slightly due to rounding and because Washington, DC is included (so N=51 "states" for each map. Additionally, we keep the same "bin" thresholds for some longitudinal presentations (e.g. a separate map for 3 successive years), and in this case, if there is a secular trend in the data, the number of states in each "quintile" may vary substantially from one year to the next.

to the nearest thousandth (or tenth of one percent); this is enough precision to distinguish substantive differences among the states and will reduce confusion later about which quintile a state should go in.

```
* STEP 1: Summarize 0/1 resident indicators to state level;
PROC SUMMARY DATA = residents NWAY ;
CLASS state ;
VAR Age_ge65 Age_ge85 Age_ge95 Minority ADLimp0 ADLimp_ge4 Falls_any
   Falls_inj_any Presulc Restrnt_any Incontinent FeedingTube WeightLoss
   Antipsych;
OUTPUT OUT = statesum MEAN=;
RUN;

* Round before getting ranges;
DATA statesum_all ;
 SET statesum (DROP = _:);

 ARRAY unr{14} Age_ge65 Age_ge85 Age_ge95 Minority ADLimp0 ADLimp_ge4 Falls_any
   Falls_inj_any Presulc Restrnt_any Incontinent FeedingTube WeightLoss
   Antipsych;
 ARRAY rnd{14} Age_ge65R Age_ge85R Age_ge95R MinorityR ADLimp0R ADLimp_ge4R
   Falls_anyR Falls_inj_anyR PresulcR Restrnt_anyR IncontinentR FeedingTubeR
   WeightLossR AntipsychR;
 DO i = 1 TO 14;
    RND{i} = ROUND(unr{i},0.001) ;
 END;
 DROP i;
RUN;
```

A partial print of the *Statesum_All* data set is shown in ***Output 1***.

STATE	PresUlc	PresulcR	Restrnt_any	Restrnt_anyR	Antipsych	AntipsychR
AK	0.065292	0.065	0.008591	0.009	0.14433	0.144
AL	0.048739	0.049	0.012544	0.013	0.26161	0.262
AR	0.044360	0.044	0.020304	0.020	0.27412	0.274
AZ	0.059019	0.059	0.007005	0.007	0.21618	0.216
CA	0.063499	0.063	0.029076	0.029	0.23185	0.232
CO	0.039487	0.039	0.013688	0.014	0.21196	0.212
CT	0.039788	0.040	0.013591	0.014	0.25625	0.256
DC	0.072027	0.072	0.008765	0.009	0.22637	0.226
DE	0.050199	0.050	0.007272	0.007	0.21904	0.219
FL	0.060228	0.060	0.019478	0.019	0.23289	0.233

Output 1. Print selected summary data, rounded and unrounded for 10 states

STEP 2: USE PROC RANK TO GENERATE QUINTILES

For the next several steps, it is simplest to process one measure at a time. Here I'm going to provide the code for a single measure, and at the end of the paper will provide the macros that get called once for each measure. (Note the use of a consistent naming convention for the variables and the output data set that will simplify implementing a macro later). PROC RANK with the GROUPS=5 option will assign each state to a quintile for each measure.

```
* STEP 2: Generate quintiles;
PROC RANK DATA = statesum_all  GROUPS=5
          OUT = stateranks_Antipsych (KEEP = state Antipsych:);
VAR AntipsychR;
RANKS Antipsych_5 ;
RUN;
```

A partial print of the *StateRanks_Antipsych* data set is shown in ***Output 2***.

STATE	Antipsych	AntipsychR	Antipsych_5
AK	0.14433	0.144	0
AL	0.26161	0.262	3
AR	0.27412	0.274	4
AZ	0.21618	0.216	1
CA	0.23185	0.232	2
CO	0.21196	0.212	1
CT	0.25625	0.256	3
DC	0.22637	0.226	2
DE	0.21904	0.219	2
FL	0.23289	0.233	2

Output 2. Print the *StateRanks_Antipsych* data set for 10 states

STEP 3: USE PROC SUMMARY TO OBTAIN DATA FOR MAP LEGEND

While PROC RANK very conveniently assigns states to quintiles of the measure, it does not readily tell us what the cut-points or thresholds are between one quintile and the next. We need this information for two purposes. First, we want to put it into the legend on our map. Second, I want to be able to assign future data to categories (or "bins") based on the same ranges; for example, in the next edition of the data compendium, we may want to use the same categories so that we could see which states have changed "quintiles"2. I also want to know how many states are in each bin because we want to include that in the legend as well. PROC SUMMARY is again going to help me get this information – here I use the rank (quintile) variable as my CLASS variable to obtain the MIN and MAX of the prevalence for each level as well as the number in each bin.

```
* STEP 3: Get Data needed for map legend;
PROC SUMMARY DATA = stateranks_Antipsych NWAY ;
CLASS Antipsych_5;
VAR AntipsychR ;
OUTPUT OUT = range_Antipsych (DROP = _TYPE_
        RENAME = (_FREQ_= n_in_bin Antipsych_5=bin))
        MIN=minAntipsych MAX=maxAntipsych;
RUN;
```

The resulting data set (*range_Antipsych*) has just five observations (one per bin). ***Output 3*** has a complete listing.

[2] The implementation of this second purpose is beyond the scope of this paper, and it involves some additional wrinkles (e.g. what if new data falls outside the range of the old data), but knowing I was going to want to do this later guided some of the decisions for the current purpose – generating the data for the figure legends.

bin	n_in_bin	min Antipsych	max Antipsych
0	10	0.123	0.203
1	10	0.210	0.217
2	11	0.219	0.233
3	10	0.236	0.267
4	10	0.269	0.331

Output 3. Ranges for quintiles for the Antipsychotic measure and number of states in each group

STEP 4. PREPARE THE RANGE AND COUNT DATA SET FOR PROC FORMAT

I want to construct a FORMAT library to store all the legend labels. There are a few things I need to do to get the rangeAntipsych data set ready to be used as my CNTLIN= data set with PROC FORMAT. The DATA step code is shown below. Here is what is going on:

a) The RETAIN statement assigns a format name (Antipsychf) and specifies that I will be creating a numeric format. PROC FORMAT expects these variables to have the names FMTNAME and TYPE. Though it is trivial for a 5-observation data set, this method is slightly more efficient than assignment statements.
b) The required START and END values (which specify the range for the values to be formatted) both get the value of the variable BIN (0 to 4).
c) Note in **Output 3** that there are gaps between the end of one range and the beginning of the next range. This wouldn't look right on our legend, nor would it work correctly when new data (which might well have values falling in these gaps) gets filtered with this format. So, beginning with bin 1, I want the starting value for the bin to be 0.1 greater than the ending value for the previous bin. I use the LAG function to make this happen.
d) MINVAL will hold the lower bound of each range and MAXVAL the upper bound. I multiply by 100 and round to the nearest tenth so that the values will be in the format xx.x (like a percent, which is the way people are used to seeing prevalence estimates).
e) Here I assign the values to the required LABEL variable, using concatenation functions (CATS and CAT) to format them precisely as desired.

```
* STEP 4: Prepare data as input to PROC FORMAT;
DATA cntl_Antipsych ;
 SET range_Antipsych ;

 RETAIN FMTNAME 'Antipsychf'          /* STEP a */
        TYPE    'N'  ;
 START = bin;                         /* STEP b */
 END = bin;

lastmaxAntipsych = LAG(maxAntipsych);        /* STEP c */

* numeric start and end points for each of the bins ;       /* STEP d */
IF bin = 0 THEN minval = ROUND(100*minAntipsych,0.1);
ELSE minval = ROUND(100*lastmaxAntipsych + 0.1,0.1);
maxval = ROUND(100*maxAntipsych,0.1) ;

* convert above to labels for legend, adding # of states;   /* STEP e */
LENGTH bincount $4 LABEL $18 ;
space = ' ';
bincount = CATS('(',PUT(n_in_bin,2.),')');
LABEL = CAT(PUT(minval,4.1),' to ',PUT(maxval,4.1),space,space,bincount) ;

DROP space bincount ;
RUN;
```

The resulting data set (*cntl_Antipsych*) is listed in ***Output 4***. The only variables that will be used by PROC FORMAT are FMTNAME, TYPE, START, END, and LABEL, but the others are shown for clarity. Note how the ranges have been extended to eliminate gaps in between. (As an aside, if we were going to use this data to create a format to assign new data to bins, we would use MINVAL and MAXVAL as our START and END and BIN as our LABEL.)

bin	_bin	minAnt_psych	maxAnt_psych	fmtname	type	start	end	lastmaxAnt_psych	min_val	max_val	label	(label)
0	10	0.123	0.203	Antipsychf	N	0	0	.	12.3	20.3	12.3 to 20.3	(10)
1	10	0.210	0.217	Antipsychf	N	1	1	0.203	20.4	21.7	20.4 to 21.7	(10)
2	11	0.219	0.233	Antipsychf	N	2	2	0.217	21.8	23.3	21.8 to 23.3	(11)
3	10	0.236	0.267	Antipsychf	N	3	3	0.233	23.4	26.7	23.4 to 26.7	(10)
4	10	0.269	0.331	Antipsychf	N	4	4	0.267	26.8	33.1	26.8 to 33.1	(10)

Output 4. CNTL data set for construction of FORMAT to assign legend labels.

STEP 5. ADD FORMAT MAPPING QUINTILE VALUES TO LEGEND LABELS TO LIBRARY

The next step is extremely straightforward – just use PROC FORMAT to read the CNTL dataset built in the last step to generate a FORMAT and add it to a format library specific to the annual data compendium. The PROC CATALOG step that allows us to give a description to the format (which will show up in the format catalog's metadata) is a nice touch.

```
* STEP 5: Add format for this measure to format library;
PROC FORMAT LIBRARY = LIBRARY.Compendium2011Res CNTLIN=cntl_Antipsych  ;
RUN;

PROC CATALOG CATALOG = LIBRARY.Compendium2011Res ;
MODIFY Antipsychf.FORMAT (DESCRIPTION = "Legend labels for map [Antipsych]") ;
CONTENTS ;
RUN ;
QUIT;
```

The CONTENTS statement off PROC CATALOG will produce the listing shown in ***Output 5***. Obviously this will be more useful when the catalog contains a lot of formats.

```
Contents of Catalog LIBRARY.COMPENDIUM2011RES

#  Name         Type    Create Date   Modified Date   Description
-------------------------------------------------------------------------------
1  ANTIPSYCHF   FORMAT   07/21/2014    07/21/2014   Legend labels for map [Antipsych]
```

Output 5. Metadata (output from PROC CATALOG) for the legend label format(s).

STEP 6. GENERATE DATA SET WITH ALL DATA NEEDED FOR THE FIGURE & LEGEND

As I mentioned earlier, the map figures themselves are produced by another programmer who knows way more than I do about using SAS® to generate figures (see Hadden, 2014) but I supply her with the summarized data, ready to map with little or no additional manipulation. We agreed that I would produce a little data set for each map. While a DATA step could certainly be used here, I chose PROC SQL primarily because it allows be to easily specify the order of the variables on the resulting data set , and I wanted this to be consistent. For the maps, we need states identified by their FIPS code, and the STFIPS function translates the two-letter postal abbreviation into the FIPS code. Similarly, the STNAMEL function converts the postal abbreviation to the full name of the state (in title case). We use the format we created in the last step to associate the correct legend label with each state. Note the OPTIONS FMTSEARCH statement at the top – this may be critical to ensure that the correct version of the format is used, if, for example, the ranges and/or number of states in each category change from one year to the next. Finally, the quintile variable itself (*Antipsych_5*) is the value that is actually "mapped" for each state.

```
* STEP 6.  Write data for one map to permanent file;

* make sure to use the correct format library ;
OPTIONS FMTSEARCH = (library.Compendium2011Res);

PROC SQL;
CREATE TABLE out.Antipsych (LABEL = "Data for Map of % of Residents Taking
   Antipsychotics [var=Antipsych]") AS
  SELECT
     state          AS state_abbr  LABEL = "State (postal abbreviation)"
   , STFIPS(state)  AS state_fips  LABEL = "State (FIPS code)"
   , STNAMEL(state) AS statename   LABEL = "State Name"
   , AntipsychR     AS Antipsych   LABEL = "% of Residents Taking Antipsychotics
   (rounded)"
   , Antipsych_5
       LABEL = "Quintile for % of Residents Taking Antipsychotics [Antipsych]
   (0=low,4=high)"
   , PUT(Antipsych_5,Antipsychf.) AS legend_label
       LABEL = "Range for Bin (# of States)"
FROM stateranks_Antipsych
ORDER BY state ;
QUIT;
```

The resulting data set has one observation for each state (plus DC); a listing of the first 10 observations is shown in ***Output 6***.

state_ abbr	st_fips	statename	Antipsych	Antipsych_5	legend_label
AK	2	Alaska	0.144	0	12.3 to 20.3 (10)
AL	1	Alabama	0.262	3	23.4 to 26.7 (10)
AR	5	Arkansas	0.274	4	26.8 to 33.1 (10)
AZ	4	Arizona	0.216	1	20.4 to 21.7 (10)
CA	6	California	0.232	2	21.8 to 23.3 (11)
CO	8	Colorado	0.212	1	20.4 to 21.7 (10)
CT	9	Connecticut	0.256	3	23.4 to 26.7 (10)
DC	11	District of Columbia	0.226	2	21.8 to 23.3 (11)
DE	10	Delaware	0.219	2	21.8 to 23.3 (11)
FL	12	Florida	0.233	2	21.8 to 23.3 (11)

Output 6. Partial listing of final data set for mapping for one measure

BUILD MACRO TO AUTOMATE PROCESS FOR MANY MEASURES

Once I was certain that the process was working exactly as I wanted for a single measure, I converted the above pieces (Steps 2 through 6) into a macro that could be called once for each measure. By careful naming of the variables, I really need only a single parameter for the macro (the measure name), though I add a label for good measure. The full code is shown below with a few example macro calls.

```
%MACRO mklegend(inds=statesum_all,invar=Antipsych,labl=) ;

%LET labl=%STR(&labl.);

* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
* STEP 2: Assign the Quintiles        * ;
* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
PROC RANK DATA = &inds.  GROUPS=5
          OUT = stateranks_&invar. (KEEP = state &invar.:);
VAR &invar.R;
RANKS &invar._5 ;
RUN;

* Test print (file has one row per state);
PROC PRINT DATA = stateranks_&invar. (OBS = 10) ;
ID state ;
RUN;

* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
* STEP 3: Get Data needed for map legend  * ;
* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
PROC SUMMARY DATA = stateranks_&invar. NWAY ;
CLASS &invar._5;
VAR &invar.R ;
OUTPUT OUT = range_&invar. (DROP = _TYPE_
        RENAME = (_FREQ_= n_in_bin &invar._5=bin))
        MIN=min&invar. MAX=max&invar.;
RUN;

* Test print (file has one row per quintile (aka bin));
PROC PRINT DATA = range_Antipsych ;
ID bin ;
VAR n_in_bin minAntipsych maxAntipsych;
RUN;
```

```
* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
* STEP 4: Prepare data as input to PROC FORMAT   * ;
* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
DATA cntl_&invar. ;
 SET range_&invar. ;

 RETAIN FMTNAME "&invar.f"         /* STEP a */
        TYPE    'N'   ;
 START = bin;                        /* STEP b */
 END = bin;

lastmax&invar. = LAG(max&invar.);        /* STEP c */

* numeric start and end points for each of the bins ;       /* STEP d */
IF bin = 0 THEN minval = ROUND(100*min&invar.,0.1);
ELSE minval = ROUND(100*lastmax&invar. + 0.1,0.1);
maxval = ROUND(100*max&invar.,0.1) ;

* convert above to labels for legend, adding # of states;   /* STEP e */
LENGTH bincount $4 LABEL $18 ;
space = ' ';
bincount = CATS('(',PUT(n_in_bin,2.),')');
LABEL = CAT(PUT(minval,4.1),' to ',PUT(maxval,4.1),space,space,bincount) ;

DROP space bincount ;
RUN;


* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
* STEP 5: Add format for this measure to format library  * ;
* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
PROC FORMAT LIBRARY = LIBRARY.Compendium2011Res CNTLIN=cntl_&invar.  ;
RUN;

PROC CATALOG CATALOG = LIBRARY.Compendium2011Res ;
MODIFY &invar.f.FORMAT (DESCRIPTION = "Legend labels for map [&invar.]") ;
CONTENTS ;
RUN ;
QUIT;

* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;
* STEP 6.  Write data for one map to permanent file      * ;
* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ * ;

* make sure to use the correct format library ;
OPTIONS FMTSEARCH = (library.Compendium2011Res);

PROC SQL;
CREATE TABLE out.&invar. (LABEL = "Data for Map of &labl. [var=&invar.]") AS
  SELECT
     state          AS state_abbr  LABEL = "State (postal abbreviation)"
   , STFIPS(state)  AS state_fips  LABEL = "State (FIPS code)"
   , STNAMEL(state) AS statename   LABEL = "State Name"
   , &invar.R      AS &invar.   LABEL = "&labl. (rounded)"
   , &invar._5
       LABEL = "Quintile for &labl. [&invar.] (0=low,4=high)"
   , PUT(&invar._5,&invar.f.) AS legend_label
       LABEL = "Range for Bin (# of States)"
FROM stateranks_&invar.
ORDER BY state ;
QUIT;
```

```
* Testing ;
PROC PRINT DATA = out.&invar. (obs=10);
ID state_abbr ;
run;

PROC CONTENTS DATA = out.&invar. ;
RUN;

%MEND mklegend ;

* A few example calls to the macro;
%mklegend(invar=Age_ge95,labl=% of Residents Age 95 and older)
%mklegend(invar=ADLimp0,labl==% of Residents with 0 ADL impairments)
%mklegend(invar=Falls_any,labl==% of Residents with Any Recent Falls)
%mklegend(invar=Antipsych,labl==% of Residents Taking Antipsychotic Medication)
```

CONCLUSION

What started as a fairly basic analytic and data manipulation problem of assigning states to categories of a measure for purposes of mapping became an interesting formatting task because of the need to label the categories in a specific way and the need to make the process reproducible – from measure to measure and year to year. Further, the need to do precisely the same set of manipulations for dozens of measures made the job an obvious candidate for macros. In my view, the real "trick" here is using PROC SUMMARY to reveal (and document) the thresholds between the quintile categories from the data produced by PROC RANK – which assigns the categories for the current data but doesn't "tell" us what those breakpoints are. I hope that there are a few other simple techniques in this process that you'll find useful as well.

There are some obvious possible extensions of the code. With relatively little revision, the macro could be (and has been) extended to process data for multiple years, different numbers of categories for different measures, different levels of precision. We've also extended it to accommodate new data – where we don't want the basic categories to change, but we may need to extend the overall range because of secular trends in the data. Briefly, we can pull the previous year's information out of the formats (using the CNTLOUT option), check if the ranges need to be extended, generate new formats to assign the new data into the (mostly) unchanged bins, and, of course, re-generate the counts of states going into each bin.

REFERENCES

Centers for Medicare and Medicaid Services, "Nursing Home Data Compendium 2013 Edition", Available at *https://www.cms.gov/Medicare/Provider-Enrollment-and-Certification/CertificationandComplianc/downloads/nursinghomedatacompendium_508.pdf*

Hadden, Louise. 2014. " Extreme SAS® Reporting II: Data Compendium and 5 Star Ratings Revisited" *Proceedings of SAS Global Forum 2014*. Available at http://support.sas.com/resources/papers/proceedings14/1342-2014.pdf

ACKNOWLEDGMENTS

The author gratefully acknowledges the contributions of several CMS colleagues, including Edward Mortimore, Jon Friedlander, Dan Andersen and Ian Kramer. Louise Hadden at Abt Associates actually generated the beautiful maps.

CONTACT INFORMATION

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

Name: Christianna Williams
E-mail: Christianna.S.Williams@gmail.com