Paper 072-2009

Let Me Look At It! Graphic Presentation of Any Numeric Variable

Anastasiya Osborne, Farm Service Agency (USDA), Washington, DC

ABSTRACT

Have you ever been asked to produce a high quality, management-friendly report in record time? Have you ever spent time typing ranges for PROC FORMAT to apply in tables or maps? During Congressional hearings, U.S. Department of Agriculture (USDA) often gets urgent requests to graphically represent politically-sensitive data. This paper presents a SAS® macro that was developed to allow flexibility in choosing a dataset, a variable in question, and a number of groups for statistical analysis. The macro then produces the results in an Excel spreadsheet, and an ODS output. It also automatically creates a format for the variable that can be used in PROC GMAP to produce an impressive map. The macro reduces programming time by eliminating time-consuming tasks to analyze the variable and manually type ranges for PROC FORMAT.

INTRODUCTION

Being a member of the Economic and Policy Analysis Staff at the Farm Service Agency (FSA), USDA requires stamina and creativity. A stream of urgent requests to produce ad-hoc reports with statistical analysis of data can come at any time. The effort in creating these reports can be time-consuming and inefficient, especially when analysis of unfamiliar data is needed within a short period of time, as, for example, during Congressional deliberations. This is when SAS MACRO facilities can be handy. MACRO saves time and automates a tedious mistake-prone process of typing format ranges, so that the mind of the analyst is freed to tackle more complicated issues. This automated approach to analyze the variable, create a user-defined format, and map data drastically reduces staff time to produce a report.

WHY AUTOMATE THE MAP-MAKING PROCESS?

Our office prepares analytical reports for FSA and USDA senior management, and sometimes for Congressional staff. Automating the report writing process as much as possible helps to provide consistent data analysis, include frequently requested statistics, present data in a standard format, and reduce turn-around time of multiple and often similar requests. One conventional statistical technique for creating useful maps is to divide the range of the variable into several groups - for example, 5, 7, 10, and so on. The number of level has visual impact, and the experience has been that assigning 8 groups is enough for a map.

The analyst would decide on the number of groups, and SAS would create those groups with similar frequencies. It is useful to increase or reduce the number of groups, when working with unfamiliar data, to see whether the consequent map makes the pattern more visible. If the range is too large, as in the following example with U.S. corn production by county, it is hard to divide the range (e.g. from 600 bushels to 60,4 million) into equal parts and create a meaningful map. However, mapping frequencies, rather than evenly distributed ranges, creates a much more user-friendly product.

The macro that generates these reports uses three distinct SAS programming techniques: 1) generate ranges for each variable based on a user-requested number of frequency groups; 2) assign ranges for use in PROC FORMAT 3) use the new format in SAS/GRAPH procedures (e.g. PROC GMAP). The resulting code will automatically generate a colorful standard map showing a user-requested number of groups, for any variable. The code also generates an Excel file for documentation purposes.

GENERATING RANGES

First, we need to look at the data. To create “best” ranges for use in PROC FORMAT, first use PROC FREQ to examine the data we intend to map.

```
PROC FREQ data = &DS noprint ;
   tables &VAR/missing norow nocol nopercent
      out  = freq_&VAR ;
RUN ;
```

A careful look at the frequency dataset will guide us in choosing the number of groups we will use. However, the frequency dataset can be long (albeit shorter than the original dataset), and it can be difficult to analyze it manually.

USING PROC RANK/SORT/SUMMARY/EXPORT TO GROUP AND DOCUMENT THE DATA

Then we assign a macro parameter – how many groups are needed. The usual convention is to group data into 100 - for percentiles, 10 - for deciles, and 4 - for quartiles. For my work, we usually find that 5 to 10 groups are about right for a readable map. If the range is very large (see the following map for U.S. corn production) and “lumpy” (i.e. there are large gaps in the data or clusters around certain values), we may need more groups to illustrate the data properly. However, too many groups distract the decision maker and tire the eye, so we use 8 groups now, with occasional division of the highest group into 3 level (total 10) when data warrants it.

For example, **8 groups** would be assigned using the following code:

```
PROC RANK groups= 8 data=s_&DS
   out= rank_prepare TIES = HIGH ;
   var &VAR;
   ranks rank_&VAR ;
RUN ;
```

In PROC RANK, tied values always get the same rank. In analysis of loan rates, a major project in our group, this happens frequently. The option TIES = HIGH specifies that the ties get the highest possible rank.

Then, the rank-prepare dataset needs to be sorted.

```
PROC SORT data = rank_prepare ;
   by rank_&VAR ;
RUN ;
```

Then we use PROC SUMMARY to find the minimum and maximum of each group:

```
PROC SUMMARY data=rank_prepare n max ;
   var &VAR;
   by Rank_&VAR ;
   output out= minmax&VAR
   min =min&VAR._byrank
   max =max&VAR._byrank
      ;
RUN ;
```

We export results to a user-friendly Excel file for documentation.

```
/* NOTE - use your preferred location */
PROC EXPORT data = group&VAR
   outfile = "H:\SAS programs for 2008\&dataset &lryear groups for
             &VAR..xls"
   dbms = excel2000
   replace ;
RUN ;
```

Historically, the next step here has been to use a CALL SYMPUT to create the format. However, later on another way was developed to accomplish the same goal. The following section describes the historic way.

USING CALL SYMPUT TO ASSIGN RANGE VALUES TO MACRO VARIABLES

With CALL SYMPUT, we create a macro for the min and max range of each group. The macro variables will have the same name as the dataset variables, and have an additional suffix for the rank number.

```
DATA DOIT ;
  set group&VAR ;
  suffix = normal_rank&VAR ; /* We need a suffix to create macro
            variables out of each min and max value for each rank */

  array xxx(*) _numeric_ ; /* "*" is needed to count the number of
                  variables */
  do i = 1 to dim(xxx);
      call symput(cats(VNAME(xxx[i]), suffix), xxx[i]) ;
      /* CALL SYMPUT is a DATA step subroutine the assigns a value
         to the macro variable. VNAME assigns the name of the
         variable A as the value of the variable B. CATS is a
         concatenation function to add a suffix to the name supplied
         by VNAME function. */
  end ;
RUN ;
```

This is how the log looks, for 8 groups.

```
MPRINT(MKFMT):   VALUE OAPCOR2003_2007f 600 - 6866 = "600 to 6,866" 7000
- 28467 = "7,000 to 28,467" 28730 - 98000 = "28,730 to 98,000" 98333 -
342667 = "98,333 to 342,667" 343600 - 1097000 = "343,600 to 1,097,000"
1099333 - 3519333 = "1,099,333 to 3,519,333" 3522000 - 11176667 =
"3,522,000 to 11,176,667" 11183333 - 60399967 =
"11,183,333 to 60,399,967" ;
NOTE: The previous statement has been deleted.
MPRINT(MKFMT):   RUN ;
```

You then have to copy this text and paste it into the body of the SAS program.

USING CNTLIN OPTION TO CREATE A FORMAT

This is a way to accomplish the same goal without having to copy and paste a part of the log manually into the SAS program, although in the previous method it is easier to spot problems with the underlying data (e.g., not enough data to make the desired number of groups). Some crops, such as corn, soybeans, and oats, are produced in many states, and can be easily shown in 10-12 groups. The minor crops, for example crambe and mustard, can have only 4-6 levels, consisting of one value rather than a range.

Use the **CNTLIN** option in PROC FORMAT to create a format from a dataset automatically. We need a dataset with the variables START (values), LABEL (descriptions for those values), and FMTNAME (how we want to name this format). FMTNAME is a character variable and requires quotes.

```
DATA
   fmtdata_&VAR ( keep = fmtname start end label)
   group&VAR    ( keep = normal_rank&VAR start end _freq_
                 rename = (start=min&VAR end=max&VAR)
                 );
       length normal_rank&VAR start end 8. fmtname $ 32
               label $ 32 ;
  retain fmtname "&VAR.f" ; /* This will be repeated in each row */
   set minmax&VAR end = EOF ;
   by rank_&VAR ;


    normal_rank&VAR = rank_&VAR + 1; /* We see that the ranks start at
                       zero. To create a "normal rank", I add 1. */

   if normal_rank&VAR = . then delete ;
   start = round(min&VAR.by_rank, 0.01) ;
   end = round(max&VAR.by_rank, 0.01) ;
   label = catx( " " , put(start, comma14.0 ) , "to" , put(end,
           comma14.0) ) ;
   if eof then
     call symputx ( "last_rank' , normal_rank&var ) ;
RUN ;
```

To use the **CNTLIN** option in PROC FORMAT, comment out the previous DATA **DOIT** step and run the following lines calling PROC FORMAT with the CNTLIN option.

```
PROC FORMAT CNTLIN = fmtdata_&VAR ;
RUN ;

PROC FORMAT fmtlib ;
RUN ;  /* For printing the contents of a format catalog */
```

PROC FORMAT USING THE MACRO

Now lower and upper limits of each group become numbered macro variables. The number of ranks is the value for the macro variable &last_rank.

```
%macro MKFMT ( ) ;
   %local i ;

   PROC FORMAT;
      VALUE &VAR.f
        %do i = 1 %to &last_rank ;
           &&min&VAR&i - &&max&VAR&i    = "&&min&VAR&i to &&max&VAR&i"
        %end ;
      ;
   RUN ;

%mend  MKFMT ;

options mprint ;
%MKFMT()
```

This is the result of **CNTLIN** option, an fmtdata_&var dataset.

```
----------------------------------------------------------------------------
|                  FORMAT NAME: OAPCOR2003_2007F LENGTH: 24                 |
|   MIN LENGTH:   1  MAX LENGTH:  40  DEFAULT LENGTH  24  FUZZ: STD         |
|--------------------------------------------------------------------------|
|START           |END             |LABEL  (VER. 9.2     20FEB2009:14:37:48)|
|----------------+----------------+----------------------------------------|
|             600|            6866|600 to 6,866                            |
|            7000|           28467|7,000 to 28,467                         |
|           28730|           98000|28,730 to 98,000                        |
|           98333|          342667|98,333 to 342,667                       |
|          343600|         1097000|343,600 to 1,097,000                    |
|         1099333|         3519333|1,099,333 to 3,519,333                  |
|         3522000|        11176667|3,522,000 to 11,176,667                 |
|        11183333|        60399967|11,183,333 to 60,399,967                |
----------------------------------------------------------------------------
```

One way or the other, we are now ready to map the data.

USING DEFINED FORMAT IN SAS/GRAPH

Before using the defined format in PROC GMAP, it is good practice to set GOPTIONS for an appealing look of the map. Here is the recommended page setup, including legend options:

```
   GOPTIONS RESET = GOPTIONS /*global*/

            GUNIT = pct /* Specifies the unit as percent, unless explicitly
                        specified in another SAS statement. */

            border      /* Printed map is framed */
            cback = white
            ROTATE = landscape  /* The way it's printed on a page */
            GSFMODE = append /* This is to control whether maps replace one
                  another, or append to the bottom of the specified file */
            Ctext = black
            Ftext = swiss
            Htext = 9 PT ;
            TITLE2 f = swissb ; /* 'Times-Bold'; */

 legend1    mode = RESERVE /*Better than SHARE – no map crowding occurs when
                      the legend is partly covered by the map.*/
            shape = bar(1.4, 1.6) /* Shape of the shaded bars. Units
                                specified in GUNIT option. */
            position = (top) /*(bottom)*/ /* POSITION moves the legend above
                      the map. Defaults: outside, center. */
            value = (font = swiss)
            label = ('') /* LABEL used to suppress the legend label. Default:
                        The name of the response variable.*/
            frame  /* Box drawn around the legend */
            cshadow = gray9a /* This adds a drop shadow to the box around the
                          legend*/
            cborder = gray
            cframe = white;
```

I also developed a color scheme. There is SAS documentation about the colors, but it takes some time to develop a scheme, as the colors look different on different printers.

The color scheme depends on what variable is being depicted. For example, maps showing changes in payments range in color from light yellow to dark green. A scheme for changes in loan rates comparative to the last year takes gray as the “No change” color, green-blue-black for negative changes, and yellow-pink-red for positive changes. Below is a color scheme for 10 strictly positive levels, such as crop production, loan rates, and number of users:

```
/* Color patterns for positive levels (ranges), updated in October 2008. */
pattern1  v = msolid c =LemonChiffon;
pattern2  v = msolid c = cxffcc00;         /* Gold  */
pattern3  v = msolid c = orange;
pattern4  v = msolid c = cxd9892b;         /* Brilliant  orange */
pattern5  v = msolid c = vpag;             /* Very pale green */
 pattern6  v = msolid c = YellowGreen ;     /* Lighter green */
 pattern7  v = msolid c = MediumSeaGreen ; /* Darker  green */
pattern8  v = msolid c = brown;
 pattern9  v = msolid c = CX800000 ;        /* Maroon  */
 pattern10 v = msolid c = CX33070F ;        /* “Vivid red” that shows as
                                               almost black  */
```

Please look at the appendix for a thorough explanation (with map examples) of the color scheme for changes.

MAP MAKING

Now we use the created format to generate a map of the variable. I have a master mapping program where I use %INCLUDE and call the macro %ASSIGN to create formats. There are better ways to do this, such as using the AUTOCALL macro facility, which permits faster updates and better SAS program consistency (see Jensen). %INCLUDE is what I am using here.

This macro %ASSIGN is run in another macro, %***GET_FORMATS,*** if we need to map many crops at a time.

```
    %macro GET_FORMATS (DS = , VAR = , GROUP = ,
                        CR = , CROP = , unit =  ) ;
    … /*create a subset if the master database, sort it by the variable */

    %INCLUDE "H:\SAS papers - mine\assign groups June 2 2008 new.sas";
    %ASSIGN (DS = &CR._formap, VAR = &CR.pro&lryear, GROUP = &GROUP,
             CR = &CR, CROP = C&CROP, unit = &UNIT ) ;

    %mend GET_FORMATS ;

    %GET_FORMATS (DS = COR_formap, VAR = CORpro&lryear, GROUP = 8,
                  CR = COR, CROP = Corn, unit = Production (bushels));
```

In the “Using Proc Rank/Sort/Summary/Export to Group the Data” section, I explained what is going on behind the scenes when %***GET_FORMATS*** is run.

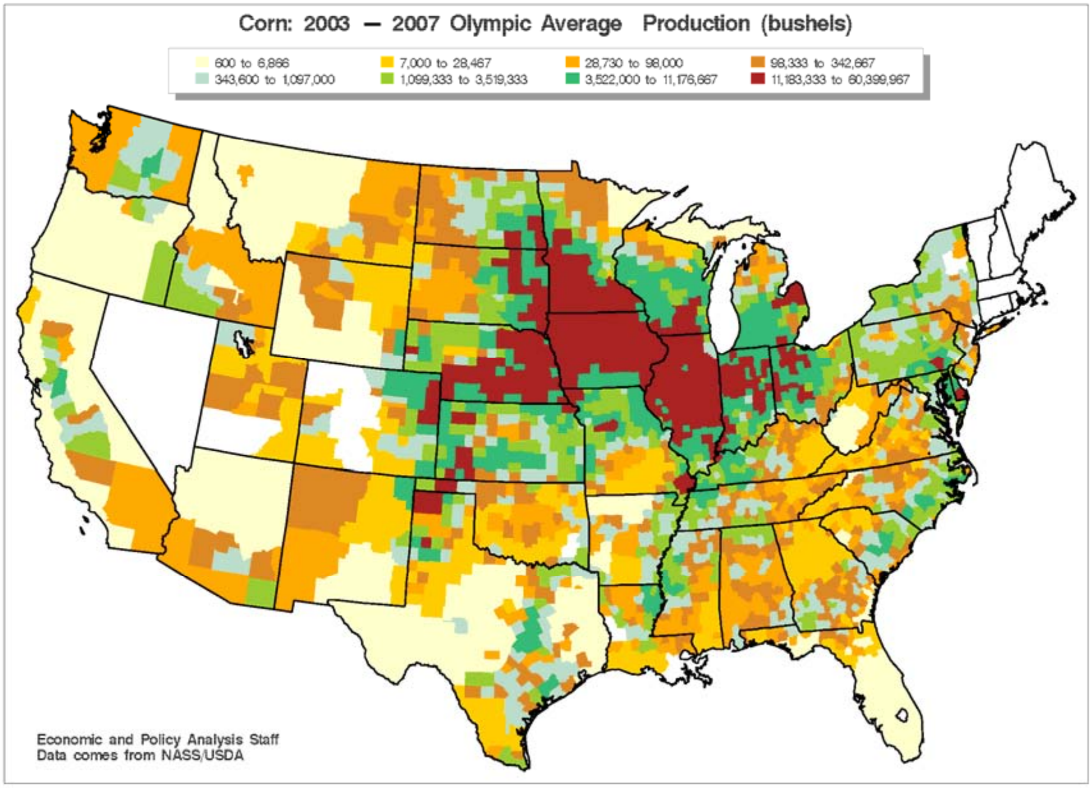
Now we are all set up for mapping. My group prefers to have a PDF file as the output on a shared drive where all maps for a given project are located:

```
/*    OUTPUT */
 filename OUT "H:\31_Maps\Output of maps lr2009\&VAR 2003-2007 OAP.pdf";

 ODS listing close;
 ODS PDF file= OUT ;

title1 h= 15 PT f = swissb "&VAR: 2003-2007 Olympic Average Production
                            &unit";
```

```
PROC GMAP data = &VAR._formap map = maps.uscounty;

   id state county; /* These are the variables common to the "maps" dataset
                  and the current dataset we are trying to map. ID statement
                  declares the variables that match response data set
                  observations to map dataset areas.
                  Second ID means that the map will display the boundaries of
                  all state map areas, but only those county map areas that
                  are requested. */

    choro &VAR    /*This is the requested variable we have been trying to map
                all this time. CHORO statement instructs PROC GMAP to create a
                choropleth map displaying values of the variable */
      annotate = HOME.slines  /* This way we don't see the separations
                              between counties - we see a group of counties
                              with the same color as one block */
      cempty = white /* Specifies that empty areas are to be outlined in
                        white color */
      coutline = same
      discrete
      legend = legend1;

      note height = 10 pt f = swissb move = (3,5)  color = black
      'Economic and Policy Analysis Staff, FSA/USDA';
      note height = 10 pt f = swissb move = (3,3)  color = black
      ‘Data comes from NASS/USDA';

     format &VAR   &VARF. ; /* Here is our format. */

   RUN ;
   QUIT ;

   title; /* TITLE and FOOTNOTE statements are global and can appear anywhere
         within SAS code. They are placed in their own areas within the
         graphics output. They decrease the area map. No overwriting of the
         map occurs.*/

   GOPTIONS RESET = pattern; /* To cancel all current PATTERN statements, use
          the RESET = option in GOOPTIONS statement */


  ODS PDF CLOSE;
```

HOW IT LOOKS

You can see the brown “Corn Belt” – the area in the Midwest of the United States where corn is the predominant crop. The data is publicly available and comes from the National Agricultural Statistics Service (NASS). Iowa, Illinois, Indiana, and Ohio grow about 50% of U.S. corn.


Corn: 2003 — 2007 Olympic Average Production (bushels)

Economic and Policy Analysis Staff
Data comes from NASS/USDA

CONCLUSION

This paper shows how any numeric data can be presented graphically, by using an automatically created format and a well thought-out color scheme. Automation of the graphics helps reduce typos, allows automatic verification, and dramatically speeds up the report turn-around time. It also conveys results in an intuitive, appealing, convincing way, and makes the decision makers happy.

REFERENCES

Bilenas, Jonas V. “I Can Do That With PROC FORMAT.” Available at:
http://www2.sas.com/proceedings/forum2008/174-2008.pdf

Jensen, Karl, and Greathouse, Matt “The Autocall Macro Facility in the SAS for Window Environment.” Available at:
http://www2.sas.com/proceedings/sugi25/25/cc/25p075.pdf

Murphy, William C. “Changing Data Set Variables into Macro Variables.” Available at:
http://www2.sas.com/proceedings/forum2007/050-2007.pdf

Scerbo, Marge “Get your hands dirty using the FORMAT procedure.” Available at:
http://www2.sas.com/proceedings/forum2007/189-2008.pdf

APPENDIX: COLOR SCHEME FOR CHANGES

The easiest way to communicate the results of your analysis to non-analysts (management, Congress) is to use an intuitive color scheme that everyone understands. For example, colors can convey statistical information if they range from cold to hot. Negative values can be dark (“cold”), and positive values can be lighter (“hot”). “No change” should be depicted with gray, a neutral color.

My tried-and-true approach to depicting CHANGES in values rather than values themselves resulted in the following master color scheme of changes. A SAS paper TS-688, http://support.sas.com/techsup/technote/ts688/ts688.html, is very helpful in making a color scheme for unusual situations.
Another good source is the ColorBrewer website, http://www.personal.psu.edu/cab38/.

```
/* These need to be updated with each map depending on the number
   of negative, zero, and positive ranges.  Zero should ALWAYS be gray!
   After picking colors, renumber the patterns from 1 to the last.
   Delete all the patterns after the map is produced. */

pattern1  v = msolid  c = bib;              /* Brilliant blue */
pattern2  v = msolid  c = cx7674d9;         /* Blue */
pattern3  v = msolid  c = vpav ;            /* Very pale blue */
pattern4  v = msolid  c = MediumSeaGreen ; /* Darker  green */
pattern5  v = msolid  c = YellowGreen ;     /* Lighter green */
pattern6  v = msolid  c = vpag;             /* Very light green */

pattern7  v = msolid  c = ltgray;           /* The center of the color scheme
                                                   – “NO CHANGE” */
pattern8  v = msolid  c = LemonChiffon;
pattern9 v = msolid   c = CXFFB6C1 ;        /* Light pink */
pattern10 v = msolid  c = CXFF69B4;         /* Hot pink */
pattern11 v = msolid  c = CXFF1493;         /* Deep pink */
pattern12 v = msolid  c = red;
pattern13 v = msolid  c = CXDC143C;         /* Crimson */
pattern14 v = msolid  c = CXB22222 ;        /* Fire Brick */
pattern15 v = msolid  c = CX99293D  ;       /* DEPK – “deep pink”, but dark */
```

Based on this master scheme, an analyst is free to judge how significant the changes are, how lopsided they are (e.g., there are few negative ranges comparative to positive ones, but the negative values are unexpectedly low), and use the color patterns accordingly.

The idea to develop such a color scheme came to me after hearing management complain that the old maps were hard to understand. The new color scheme was more intuitive, and received positive feedback from every client. An example how the change in a color scheme would drastically increase understanding of the data is shown on the next page.

The map at the top of the next page (Fig. A1) was produced using the old official color scheme at EPAS. The scheme was developed many years ago. Each map was produced using this scheme, regardless whether the data showed the levels or the changes in levels. Also, the format ranges were assigned manually, which sometimes led to a repeat of the ranges from the last time the program was updated.

The map on the bottom (Fig. A2) shows the new color scheme and the custom ranges assigned automatically using the macro described in this paper.

Now you can finally see that corn loan rates are higher than sorghum rates in almost all U.S. counties, with rare exceptions in the north of the Midwest. Hopefully, this example shows how important the color scheme is in communicating data patterns to decision makers. Colors based on common sense and intuitively appealing will be remembered. The colors in the above color scheme are not random – they bear meaning. They create visual impact.

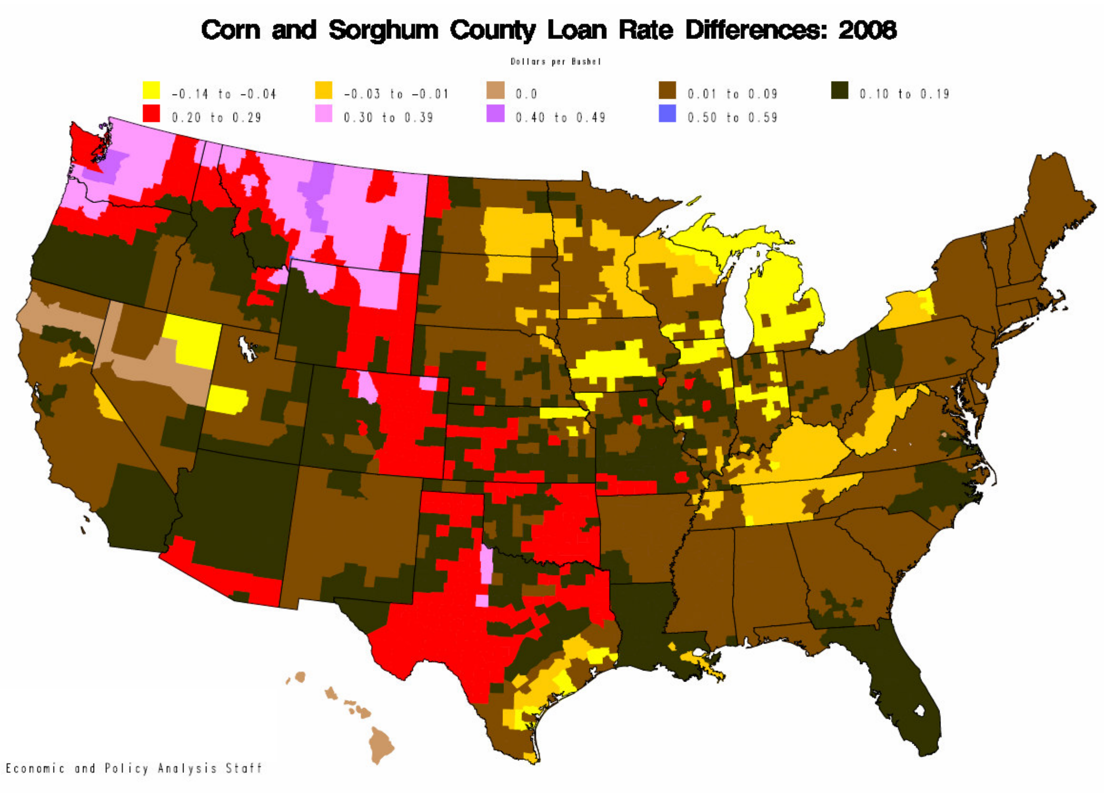


Fig. A1. Map of the difference between corn and sorghum loan rates in 2008, using **old** color scheme.

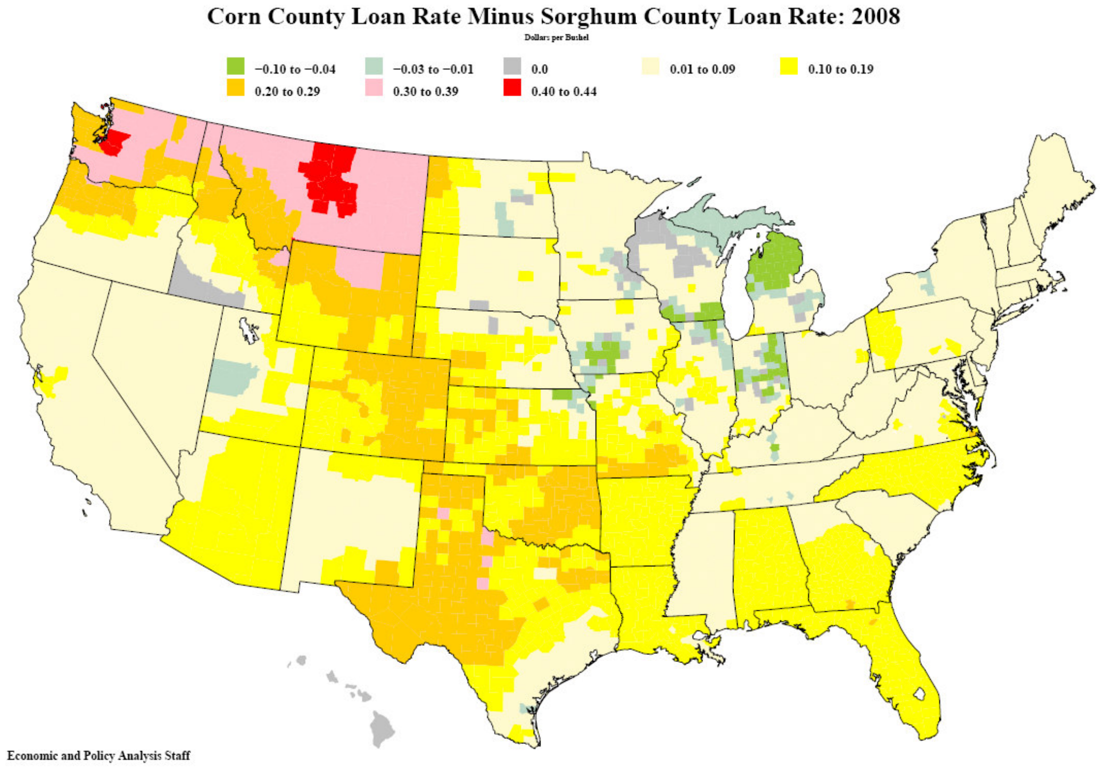


Fig. A2. Map of the difference between corn and sorghum loan rates in 2008, using **new** color scheme.

Below is another example of the drastic improvement in visibility, when the new color scheme is applied in a map of old official data. Analysis becomes easy, and the new algorithm prevents reversing of the variables.

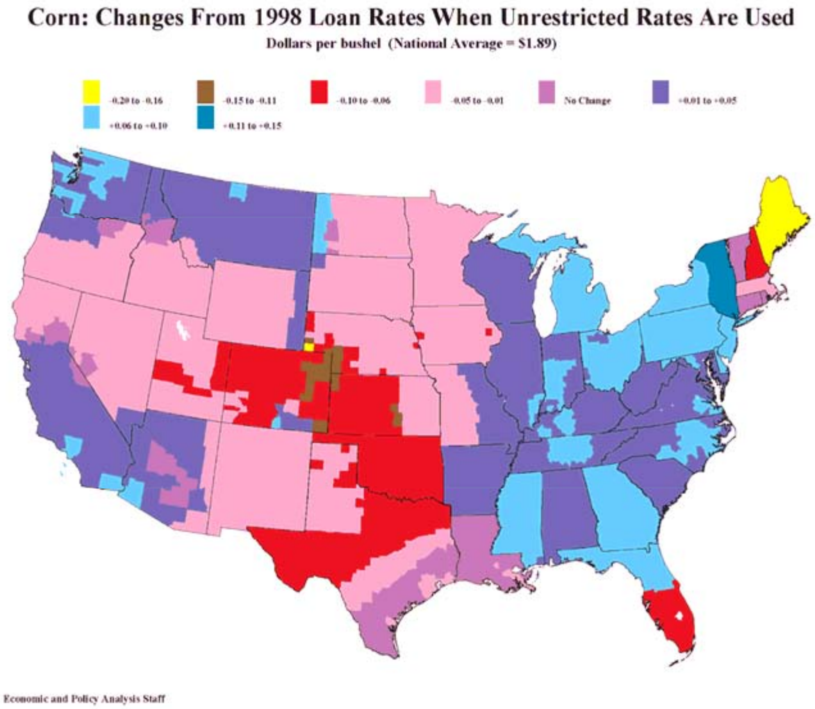

Corn: Changes From 1998 Loan Rates When Unrestricted Rates Are Used
Dollars per bushel (National Average = $1.89)
-0.20 to -0.16
-0.15 to -0.11
-0.10 to -0.06
-0.05 to -0.01
No Change
+0.01 to +0.05
+0.06 to +0.10
+0.11 to +0.15
Economic and Policy Analysis Staff

Fig. A3. Map of the changes in corn loan rates between 1998 and 1999, using **old** color scheme.

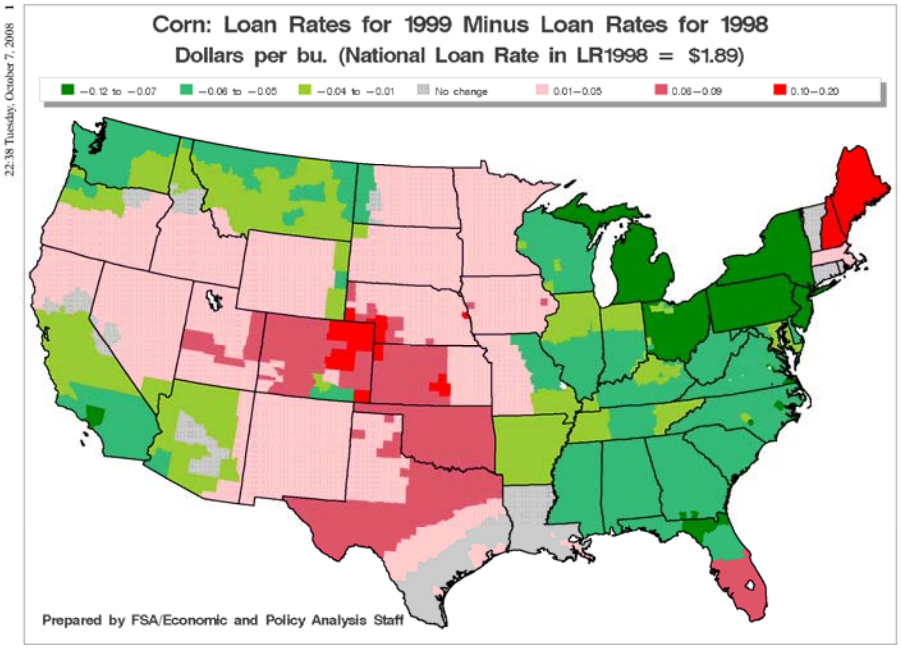
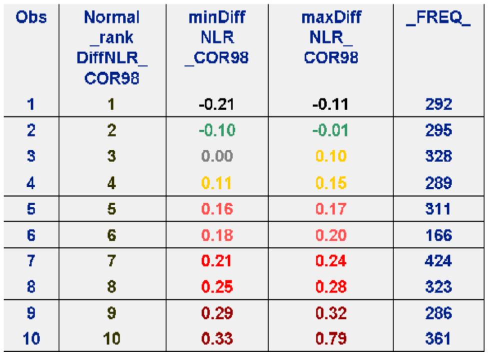

22:38 Tuesday, October 7, 2008 1
Corn: Loan Rates for 1999 Minus Loan Rates for 1998
Dollars per bu. (National Loan Rate in LR1998 = $1.89)
−0.12 to −0.07
−0.06 to −0.05
−0.04 to −0.01
No change
0.01−0.05
0.06−0.09
0.10−0.20
Prepared by FSA/Economic and Policy Analysis Staff

Fig. A4. Map of the changes in corn loan rates between 1999 and 1998, using **new** color scheme.

The Fig. A5. depicts the output of the PROC RANK procedure in ODS, described on p.2. Colors are added for better visibility. It shows how the groups are formed. It also shows that the “No change” option has to be created manually, since the automatic procedure will not put “0.00” in a separate group. In Fig. A6., the maps features the legend created from PROC FORMAT



Obs	Normal _rank DiffNLR_ COR98	minDiff NLR _COR98	maxDiff NLR_ COR98	_FREQ_
1	1	-0.21	-0.11	292
2	2	-0.10	-0.01	295
3	3	0.00	0.10	328
4	4	0.11	0.15	289
5	5	0.16	0.17	311
6	6	0.18	0.20	166
7	7	0.21	0.24	424
8	8	0.25	0.28	323
9	9	0.29	0.32	286
10	10	0.33	0.79	361

Fig. A5. Output from the PROC RANK for a map in A6.

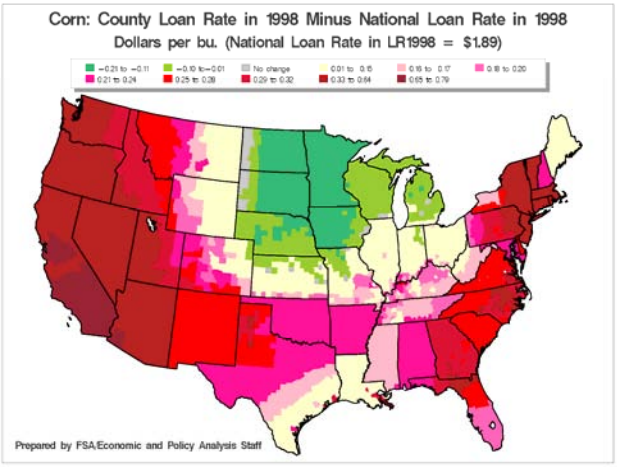

Corn: County Loan Rate in 1998 Minus National Loan Rate in 1998
Dollars per bu. (National Loan Rate in LR1998 = $1.89)
−0.21 to −0.11
−0.10 to −0.01
No change
0.01 to 0.15
0.16 to 0.17
0.18 to 0.20
0.21 to 0.24
0.25 to 0.28
0.29 to 0.32
0.33 to 0.64
0.65 to 0.79
Prepared by FSA/Economic and Policy Analysis Staff

Fig. A6. Map of a difference between a corn loan rate in 1998 and a national loan rate same year.

ACKNOWLEDGMENTS

Special heartfelt thanks to Ian Whitlock for teaching me SAS in 2000 back at Westat, and up to this day; and to Dalia Kahane for moral support in writing this paper. Also, I am full of gratitude to Joy Harwood, the Economic and Policy Analysis Staff director at the Farm Service Agency, USDA; and to Terry Hickenbotham, my supervisor at the Farm Loan Analysis Group, for allowing me to work on and present this paper at the SAS meetings and conferences.

CONTACT INFORMATION

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

Anastasiya Osborne
Farm Service Agency, USDA
1400 Independence Ave, S.W.
Washington, DC 20250-0506
Work Phone: 202-690-0446. E-mail: Anastasiya.Osborne@wdc.usda.gov