An Introduction to SAS-Lecture 6

Konstantinos Fokianos
University of Cyprus

PROC FORMAT

In this example, you have collected questionnaire data and you want to create a nice report showing the true description of the data than the numeric codes. The data look like this

Variable	Description	Codes	Columns	Type
ID	Subject ID		1-2	Character
GENDER	Subject Gender	1=MALE 2=FEMALE BLANK=missing All other=miscoded	4	Numeric
RACE	Subject Race	C=Caucasian A=African American H=Hispanic N=Native American All other=missing or miscoded	6	Character
AGE	Subject Age		8-9	Numeric
SATISFY	Satisfied with job	Likert 5-point scale	11	Character
TIME	Do you have enough leisure time?	Likert 5-point scale	13	Character

PROC FORMAT

Scales which express a range of attitudes using number such as 1 to 5 are routinely used and referred as Likert scales after a psychometrician who was the first introduced them to measure attitudes. So 1=strongly disagree, 2=disagree, 3=no opinion, 4=agree and 5=strongly agree.
We also want to create a new variable called `AGEGROUP` which groups age into 20 year intervals up to age 60, and includes all ages above 60 in a single group.
We need to create the formats that we want to use. This example actually demonstrates a lot of basic characteristics of `PROC FORMAT`.

PROC FORMAT

```
PROC FORMAT;
    VALUE GENDER 1 = 'Male'
                 2 = 'Female'
                 . = 'Missing'
             OTHER = 'Miscoded';
    VALUE $RACE  'C' = 'Caucasian'
                 'A' = 'African American'
                 'H' = 'Hispanic'
                 'N' = 'Native American'
               OTHER = 'Other'
                 ' ' = 'Missing';
   VALUE $LIKERT '1' = 'Str dis'
                 '2' = 'Disagree'
                 '3' = 'No opinion'
                 '4' = 'Agree'
                 '5' = 'Str agree'
               OTHER = ' ';
   VALUE AGEGROUP LOW-<20 = '< 20'
                  20-<40  = '20 to <40'
                  40-<60  = '40 to <60'
                  60-HIGH = '60+';
 DATA QUESTION;
   INPUT  ID      $ 1-2
          GENDER    4
          RACE    $ 6
          AGE       8-9
          SATISFY $ 11
          TIME    $ 13;
   FORMAT GENDER GENDER.
          RACE   $RACE.
          SATISFY TIME $LIKERT.;
   AGEGROUP = PUT (AGE,AGEGROUP.);
```

PROC FORMAT

- The first format GENDER associates the text strings 'Male' and 'Female' with the numeric data values 1 and 2, respectively. Missing values (.) are represented as 'Missing' and all other values as miscoded.
- The next format is $ RACE, and this is a character format. We begin with the dollar sign to indicate that this is a character format. Similarly, we create the other formats.
- We use the formats in the data step. The DATA step reads in the raw data and permanently makes the format assignment: format GENDER to variable GENDER, format $RACE to RACE and so on.
- You use the AGEGROUP. format with a PUT function to create a character variable (AGEGROUP) based on numeric values of AGE.

PROC FORMAT

Formats can also be used to aggregate data values for group processing and presentation in procedures that operate on groups of values such PROC FREQ, a procedure that counts the number (frequency) of occurrences for each formatted value of the output. This process of labeling or grouping data values using a format usually has no adverse side effects; however unexpected results can occur in PROC FREQ, specifically when formatting a variable that contain missing values.

PROC FORMAT

```
PROC FORMAT;
  VALUE BADFMT  1 = 'ONE'
                2 = 'TWO'
            OTHER = 'MISCODED';
RUN;

DATA TEST;
INPUT X  Y;
DATALINES;
1 1
2 2
5 5
3 .
;

PROC FREQ DATA=TEST;
   TABLES X Y;
   FORMAT X Y BADFMT.;
RUN;
```

PROC FORMAT

- Notice that the output shows the category MISCODED with a frequency count of 2 for the variable X.
- However for the variable Y things are quite different. Since missing values do not have their own separate range in the BADFMT. format, they immediately fall into the OTHER category. The inclusion of missing values in this OTHER category declares the entire category as missing.
- When a range of values, or a group of discrete values, are grouped together in a single format, all the values in the range or series are represented by the lowest value in the range or series. Since missing values is the lowest value possible, their inclusion makes all values to be declared as missing.

PROC GPLOT

Our goal now is to put SAS for us to create high-level graphics. The following instructions start at the bottom - where SAS makes the decisions as to the look of your graphics - and work their way gradually toward the top by adding more user control. `PROC GPLOT` is used throughout, though the same approach can be taken with any of the other SAS/GRAPH procedures.

PROC GPLOT

One important topic that seems to be discussed a lot over the last years is the amount of money spent on health care. The datasets to be used in this presentation all deal with health care expenditures both in the United States and in several other countries. The following program produces a plot that shows the total national health expenditure in the United States from 1970 through 1990. Plot 1 is *minimalist* SAS/GRAPH; in some sense it is like saying 'here are my data, show me a picture. The variables are as follows:

```
year      YEAR
nh_tot    NATIONAL TOTAL
fg_tot    FEDERAL GOVERNMENT
```

PROC GPLOT

```
data t112;
input year nh_tot fg_tot @@;
cards;
1970 74.4 17.7 1971 82.3 20.4
1972 92.3 22.9 1973 102.5 25.2
1974 116.1 30.5 1975 132.9 36.4
1976 152.2 42.9 1977 172.0 47.6
1978 193.7 54.3 1979 217.2 61.4
1980 250.1 72.0 1981 290.2 84.0
1982 326.1 93.3 1983 358.6 103.2
1984 389.6 112.6 1985 422.6 123.6
1986 454.8 133.1 1987 494.1 144.0
1988 546.0 156.7 1989 602.8 175.0
1990 666.2 195.4
;
run;

proc gplot data=t112;
plot nh_tot*year;
run;
```

PROC GPLOT

The results of this job (and each subsequent job) are shown at the graphics window. `SAS/GRAPH` has made a few decisions for yoy; among them are

- the scales on both the X and Y axes
- the font to use for text (plus other text attributes - e.g., size, and color if you have an output device that supports color)
- the symbol to use for points on the graph (and not to connect them).

Notice that the variable used first in the `PLOT` statement ends up as the Y-variable (vertical axis).

PROC GPLOT

Now we will start taking control of the graphics output.
The first change to be made is to the SYMBOL that's used to show the points on the graph.
The SYMBOL statement allows you not only to select a symbol to be plotted, but also to state whether or not to join the points, and if they are joined, how they're joined. We'll stick to basics and select a 'dot' for the points, and merely 'join' them in a 'connect the dots' fashion

PROC GPLOT

To change the plotting symbol and join the points we use

```
symbol1 value=dot interpol=join;
```

If we want to add a title (to describe the plot) and a footnote (to tell people where the data come from) we use

```
title1
"NATIONAL HEALTH CARE EXPENDITURES: 1970-1990";
footnote1
"Source: Health-United States-1990";
```

PROC GPLOT

SAS has made several decisions for producing this graph. Note that fonts have been chosen by the program, and both the title and the footnote have been centered on the graph.
To choose a font for the text in the chart we can either use a GOPTIONS statement and declare a font to be used for all text in the chart, or we can specify a font every time we tell SAS about text to be placed on the chart. We will use the GOPTIONS method will be used, and a text height for all text in the chart will be specified.

```
goptions ftext=swiss htext=2 gunit=pct;
```

PROC GPLOT

Lets look closer at the options that we are using.

- The height of text (HTEXT) is specified as 2. The main problem is what is "2" inches, centimeters, etc.? The answer is that, if no UNITS are specified, then they are character cells. SAS/GRAPH divides each output device it can use (printers, plotters, terminals, etc.) into a grid. The grid composed of character cells.
- The number of these cells varies from device to device. Before this goes too far, the easiest way to insure that graphics you design on one device (e.g., the monitor on your PC) will look similar if you change output devices (e.g., to a laser printer) is to specify that all UNITS used in your SAS/GRAPH job will be a percentage of the screen, paper, etc. - i.e., GUNIT=PCT (e.g., if your paper is 8 inches high, HTEXT=2 means text will be .16 inches high, 2% of 8 inches).

PROC GPLOT

The GOPTIONS has changed all of the text height - the title should stand out, so we will override the goptions htext=2 with a text height specified for the title. The footnote will also be moved to the lower right (right justified) using the 'JUSTIFY=' option (also available for titles, but let's leave the title centered).

```
title1 h=4
"NATIONAL HEALTH CARE EXPENDITURES: 1970-1990";
footnote1 justify=right
"Source: Health-United States-1990";
```

PROC GPLOT

Now that the line, title, and footnote look OK, it's time to work on the appearance of the X and Y axes. We can control just about any aspect of the axes. The first thing we'll do is change the labels. The X-axis label of 'YEAR' is fine, but the Y-axis might look better if it were labeled 'AMOUNT (IN BILLIONS)'.

```
axis1 label=(angle=90 rotate=0
"AMOUNT (IN BILLIONS)");
proc gplot data=t112;
plot nh_tot*year/vaxis=axis1;
run
```

PROC GPLOT

Simply adding an AXIS statement to our SAS code will not change either the X or Y axis. The PROC must be told to use the AXIS definition. In this case, for the vertical (or Y) axis. We've changed the label text and its angle.

- The ANGLE option determines the angle of the baseline on which the text rests.
- The ROTATE option determines the angle of the text relative to the baseline.

PROC GPLOT

We will now change another attribute of the axes, i.e., the number of tick marks and number of labels printed for axis values.

```
axis1 label=(angle=90 rotate=0
"AMOUNT (IN BILLIONS)")
minor=(n=3);
axis2 order=(1970 to 1990 by 5)
minor=(n=4);
proc gplot data=t112;
plot nh_tot*year/vaxis=axis1
                 haxis=axis2;
run;
```

On the vertical axis, we add a minor tick mark at every 25 billion dollars. On the horizontal axis, we change the number of years that are printed on the chart - every fifth year is labeled - the intervening years are noted by minor tick marks.

PROC GPLOT

Now we'll add some color. Color is another attribute that can be controlled either within the GOPTIONS statement or within each statement (titles, footnotes, axes, etc.).

```
goptions ftext=swiss htext=2 gunit=pct
ctext=green csymbol=yellow;
axis1 label=(angle=90 rotate=0
"AMOUNT (IN BILLIONS)")
minor=(n=3)
color=yellow;
axis2 order=(1970 to 1990 by 5)
minor=(n=4)
color=yellow;
```

We have used the 'CTEXT=' and 'CSYMBOL=' options with the GOPTIONS statement to set the text and symbol colors to be used in the entire chart. You can see that the symbol color affects both the dots and the line connecting the dots since both are chosen in the SYMBOL statement. What happened to the color of the axis labels and labels for axis values? They are now yellow - the CAXIS= option in an AXIS statement affects both the axis and any text associated with the axis.

PROC GPLOT

Let's make the axes and tick marks yellow, but leave all text associated with the axes green.

```
axis1 label=(angle=90 rotate=0
"AMOUNT (IN BILLIONS)")
minor=(n=3) ;
axis2 order=(1970 to 1990 by 5)
minor=(n=4);
symbol1 value=dot h=2.5 interpol=join;
proc gplot data=t112;
plot nh_tot*year/vaxis=axis1
                 haxis=axis2
                 caxis=yellow;
run;
```

Another way to change the color of both axes is to use the CAXIS= option within the PROC. This color selection only changes the color of the axes themselves and leaves the text green (as chosen with the CTEXT= option in the GOPTIONS statement). The H= SYMBOL option uses the same units as text.

PROC GPLOT

In addition to total national health care expenditures, the dataset also contains information as to how much money the federal government has spent on health care from 1970 through 1990. It would be interesting to see how the total amount spent on health has changed relative to government expenditures. We will leave all our options (symbols, colors, axes, etc.) unchanged and add another line to the chart.

PROC GPLOT

```
proc gplot data=t112;
plot nh_tot*year
     fg_tot*year /overlay;
run;
```

We can add another line merely by adding another set of 'Y*X' variables to the plot statement, and by adding the 'OVERLAY' option. Just as when we started building the chart, SAS/GRAPH has chosen its own symbol for the second line, and has chosen not to join the points. We need another SYMBOL statement to control the appearance of the second line

PROC GPLOT

```
symbol1 value=dot h=2.5 interpol=join line=1;
symbol2 value=dot h=2.5 interpol=join line=3;
```

There are a number of different ways to differentiate the lines on a plot.
One way is to change the symbol VALUE - there are dots, triangles, squares, etc.
Another way is to change the color (not an option is your printer only allows black-and-white).
A third way is to change the type of line used to connect the points by using the 'LINE=' option.
SAS/GRAPH provides forty-six different line types. For this example, lines 1 and 3 (solid and dashed) are chosen . We should probably change the title, but even if we change the title, something else is missing. It would be nice to have a LEGEND to tell those looking at the chart what each line represents.

PROC GPLOT

```
goptions reset=symbol;
symbol1 value=dot h=2.5 interpol=join;
symbol2 value=square h=2.5 interpol=join;
title1 h=4
"HEALTH CARE EXPENDITURES IN THE UNITED STATES:
1970-1990";
proc gplot data=t112;
plot nh_tot*year
fg_tot*year/vaxis=axis1
            haxis=axis2
            caxis=yellow
             overlay
legend;
```

PROC GPLOT

The label on the legend - 'PLOT' - is not very descriptive. Neither are the labels - 'NH_TOT' and 'FG_TOT' - on the two lines. Also notice that we have changed the SYMBOLS. We are no longer using line type to differentiate the data, but the symbol itself (a dot versus a square). You probably also noticed that there is a 'GOPTIONS RESET=SYMBOL' statement in the code. If this was not included, SAS/GRAPH would have remembered that the line type for the second symbol was dashed. SYMBOL statements are GLOBAL, i.e., they retain their values for the duration of a SAS session until you change them with another SYMBOL statement or reset them within a GOPTIONS statement. Since we had set the line type and do not have a 'LINE=' option in the new SYMBOL statement, the line would have remained dashed for SYMBOL2. SYMBOLS are not the only GLOBAL statements - TITLES, FOOTNOTES, LEGENDS, AXES are all GLOBAL.

PROC GCHART

With the GCHART procedure, you can not only generate BAR CHARTS, but you can also draw PIE and BLOCK charts as well. We will use the SALES data to illustrate the main concepts.

```
PROC GCHART DATA=SALES;
   TITLE 'Vertical Bar Chart';
   VBAR REGION;
RUN;
```

This is a frequency chart and it is the simplest type of chart that you can produce. In this type of chart, the bars represent the number of observations that have the values displayed in the X axis.
The VBAR statement specifies the variable for which you want frequency counts. In this case REGION is a character variable which has the values NORTH, SOUTH EAST and WEST.

You can display the information in a horizontal bar chart by replacing the statement VBAR with HBAR in the program.

```
PROC GCHART DATA=SALES;
   TITLE 'Horizontal Bar Chart';
   HBAR REGION;
RUN;
```

By default, HBAR charts include the same frequency information as calculated by PROC FREQ.

You can avoid all these statistics by using

```
HBAR REGION / NOSTAT;
```

Suppose now that you want the height (VBAR) or the length (HBAR) of the bars to represent the frequency in percent of the total observations rather than counts. Then you employ the TYPE=PERCENT as follows.


```
PROC GCHART DATA=SALES;
   TITLE 'Vertical Bar Chart Showing Percents';
   VBAR REGION / TYPE = PERCENT;
RUN;
```

Now, the horizontal axis reports percentages instead of counts.

PROC GCHART

You can also use continuous numeric variable with the `VBAR` and `HBAR` statements. For instance, suppose that you want the frequency distribution of the variable `PRICE`:

```
PROC GCHART DATA=SALES;
   TITLE 'Vertical Bar Chart for a Continuous Variable';
   VBAR PRICE;
RUN;
```

PROC GCHART

- When you use the PROC `GCHART` to produce frequencies of percentages for a continuous numeric variable, the procedure creates sub-ranges of values of the variables to use for grouping variables. You can either let SAS to do this for or you can specify the subranges.
- The values on the X axis represent the midpoints of the interval. For instance, the bar with the label 9 shows all observations with `PRICE` between 8 and 10.

PROC GCHART

If you want to select the X axis intervals yourself, then you have two options:

- You can use the `MIDPOINTS` statement;
- You can use the `DISCRETE` statement.

PROC GCHART

```
PROC GCHART DATA=SALES;
   TITLE 'Vertical Bar Chart Demonstrating MIDPOINTS Option';
   VBAR PRICE / MIDPOINTS = 8 to 20 by 4;
RUN;
```

You can also specify midpoints by several other ways:

```
MIDPOINTS = lower_bound TO upper_bound BY interval;
```

```
VBAR PRICE / MIDPOINTS=0,5,10, 20;
```

PROC GCHART

If you have a variable that takes discrete values (1,2,3 and so on) you can use the DISCRETE statement to let SAS know:
For instance

```
PROC CHART DATA=SALES;
   TITLE 'Vertical Chart With DISCRETE Option';
   VBAR DAY / DISCRETE;
RUN;
```

PROC GCHART

Suppose now that you want to create a chart such that the total (`SUM`) of the variable `QUANTITY` for each region of the country is displayed. Then we use the `SUMVAR=` option to select the numeric variable on which to compute the `SUM` or the `MEAN` , and the `TYPE=` to indicate the statistic.

PROC GCHART

```
PROC GCHART DATA=SALES;
   TITLE 'Adding options SUMVAR= and TYPE=';
   VBAR REGION / SUMVAR=QUANTITY TYPE=SUM;
RUN;
```

In the resulting plot, the bars represent the sum of the variable `QUANTITY` for each of the regions. If you had done this using the `HBAR` statement, you would have requested the actual sums to print along the bars; if you user `TYPE=MEAN`, then you would have obtained the mean statistic.

PROC GCHART

You can represent two nested categories on the X axis. Suppose you want separate frequencies of items for each region of the country.

```
PROC GCHART DATA=SALES;
   TITLE 'Vertical Bar Chart With GROUP= Option';
   VBAR ITEM / GROUP=REGION;
RUN;
```

Suppose that you want to examine the sales frequencies by `REGION` and also want to examine which items contributed to these sales.

```
PROC GCHART DATA=SALES;
   TITLE 'Vertical Bar Chart With SUBGROUP= Option';
   VBAR REGION / SUBGROUP=ITEM;
RUN;
```

PROC GCHART

Finally, we show how to create a pie chart:

```
PROC GCHART DATA=SALES;
   TITLE 'A PIE Chart';
   PIE REGION;
RUN;
```