

Working Smart: SAS Graphics vs. Harvard Graphics

**Gregory Kovich
Inland Steel Flat Products Company**

This paper details the use of version 6.07 Base SAS Language, SAS Macro Language, and SAS/Graph using annotation, template, and replay in an IBM 3090 MVS batch environment.

The Problem

Each month Harvard Graphics, a user-friendly PC based graphics package, was employed to create and print 72 graphs (24 pages). This packet of graphs, which represents the performance of the External Processing & Distribution Department, was distributed to the officers of the Inland Steel Flat Products Company.

Harvard Graphics was the software of choice due to the high quality output it provides, the ability to mix bars and symbols on the same chart, the relative ease of annotating text and values, the ability to display multiple graphs per page, and the small skill set needed to execute this task.

Start to finish, the process of preparing and printing the packet averaged about 14 hours, approximately one man month annually! The steps to generate this report were as follows:

- Gather and arrange the data
- Enter the data into Harvard Graphics
- Annotate text onto the graphs detailing average and planned values for a quarter
- Print the output

Given the repetitive nature of creating the graphs (only the data values were different not the procedure) the SAS

System appeared to be the ideal solution needed to regain the lost productivity.

Overview of the Solution

Three points needed to be covered before source code programming could begin. First, the information had to be accessible on the Mainframe (SAS database or flat file). Second, the output had to go to a printer, not a plotter (ZETA or Hewlett Packard). Finally, the knowledge to annotate bars and text on a GPLOT had to be obtained.

Details of the Solution

To accomplish this task the following techniques needed to be implemented and integrated:

- Parameter Macros
- Annotation
- Graphics
- Template/Replay
- Output to a Hewlett Packard Laserjet

Parameter Macros

The repetitive nature of the graph creation immediately signalled that SAS Macro Language would need to be employed.

Since not every graph uses the same information, values needed to be passed to the macro in order to achieve "custom graphs". A parameter macro allows one to define as many macro variables as needed. These macro variables, which can contain

text, numbers, and even keyword statements (ex. TITLE, FOOTNOTE, etc.) are then passed to the macro allowing custom processing.

Macro language also allows nesting of macros; simply, the call of a macro within a macro. This allows greater control over the program while also providing modularity. In this example, parameter macros GRAPH2 and GRAPH4 were created as "parent" macros calling, or nesting, the ANNOTE, GRAPHIT2 or GRAPHIT4 macros.

Whenever coding with SAS Macro, it is a good practice to select the system options MLOGIC, MACROGEN, and SYMBOLGEN. MLOGIC provides both, information to the SAS Log as to when a macro has been invoked, and how the SAS System is responding to any macro condition statements. The options MACROGEN and SYMBOLGEN, when used in tandem, provide a powerful debugging tool by displaying the values SAS recognizes for macro variables and then substituting that value into the code.

Example Macro 1.1 displays the "parent" macro GRAPH2.

Example Macro 1.2 displays the "parent" macro GRAPH4.

Example Macro 1.3 displays a portion of the SAS Log during the call of GRAPH4.

Annotation

The audience for these graphs desired average values to be displayed from both the previous year and selected quarters. To display these bars throughout the first quarter the 'X' axis contains eleven tick marks. The first five are bars, representing

the average value for the previous year and past four quarters, the other six are symbols representing actual values of the next six months. (example: in January 1993, the 'X' axis displays bars for: 1992, 92Q1, 92Q2, 92Q3, and 92Q4, a symbol for J, and blank symbols for the remaining five months F, M, A, M, J). When the first month of a new quarter is plotted the bar representing the previous year's quarter are dropped (example: in April 1993, 92Q1 is dropped). Each completed actual quarter necessitates the dropping of the corresponding history quarter until only the bar representing the history year average are displayed beside twelve symbols.

This "rolling quarter" concept is facilitated through the use of conditional macro code logic (%IF,%THEN,%DO) which evaluates on the macro variable &CMONTH.

Example Annotate 1.1 displays the source code needed to draw the bars.

Example Annotate 1.2 displays the SAS Log for a run during the month of May.

Annotating text onto the GPLOT is extremely straightforward. Since average values for the quarter are not meaningful until the second month of the quarter, the annotate datasets use the WHERE option to subset observations.

Example Annotate 1.3 displays the source code needed to annotate labels in a GPLOT.

Remember, the data is already summarized: one value per month, quarter, year.

Graphics

The graphics macros GRAPHIT2 and GRAPHIT4 address two issues. First, based on the run month, the 'X' axis needs to be defined. Second, the minimum and maximum 'Y' axis values might need to be adjusted. (These values are already "seeded" in the parameter statements of GRAPH2 and GRAPH4).

The remaining code in the macro addresses the actual GPLOT itself. The output graphics catalog is identified, axis statements are defined, GPLOT options are specified and titles are selected.

Example Graphics 1.1 displays the source code needed to accomplish these steps.

Template/Replay

The use of PROC GREPLAY and TEMPLATE provides the ability to select named graphs from a catalog and place them into a custom defined template.

Three template macros are defined, one each to display; two, three, and four graphs per page, respectively. The objectives of these macros are as follows. First, to select the SAS GOPTIONS needed to support output to the device. Second, to define the template to which the graphic images will be displayed and the manner of their display.

SAS GOPTIONS provide control over the type of output, device, orientation, graphics stream file length, and resolution (vpos, hpos).

Templates can be defined either by utilizing the default SAS templates located in

the SASHELP.TEMPLT catalog or by customizing your own through the PROC GREPLAY template definition option.

The powerful combination of GREPLAY (graphics replay) and TREPLAY (template replay) allows the capability to recall graphs from catalogs by their name, and display them in your template, printed to your device in the manner that you specify.

One item which is very important but often overlooked is the GPROTOCOL SAS uses to print to your device. The next section describes three methods on how to print to a HP Laserjet; however, depending on your method, the GPROTOCOL is an extremely important selection.

Example Template 1.1 displays the source code to generate a template utilizing the default SAS template definitions.

Example Template 1.2 displays the source code for a customized template.

Output to HP Laserjet

Three methods are proven successful at Inland Steel to allow mainframe graphics to print to Hewlett Packard Laserjet printers.

The first method is considered the easiest, but contains the most steps. This method involves sending your GREPLAY output to a PDS member, downloading that member to a file on your PC, and copying it to the printer. It works, but it's ugly.

Example Output 1.1 displays an example of how this has been done.

The second method directs the output to a SNA/RJE LAN printer. The software

which controls these printers must be adjusted to accept "punched" output. The GPROTOCOL, SASGPASC, sends the graphics stream file in ASCII format to the Laserjet. GSFLN must be set to 64 in order for this method to work.

Example Output 1.2 details the JCL and GOPTIONS needed to get direct output to an enhanced SNA/RJE printer.

The last method is the most expensive, due to the cost of a protocol converter which converts the output from the mainframe into something recognizable by the Laserjet. At Inland Steel, the i-Data Protocol Converter (about \$900), with GPROTOCOL set to SASGPNET and GSFLN set to 128, accomplishes the printing to a non-SNA/RJE printer. There is no difference sending this output to the printer in the JCL. The format cards look exactly like you are sending text output to a printer.

Conclusion

Through the use of SAS Macro language, repetitive graphics procedures were easily reproduced.

Through the use of the Annotate facility available to SAS/GRAPH bars and text were produced and included with GPLOT output.

Through the use of PROC GREPLAY and template definitions, multiple graphs per page were defined and output directed to Laserjet printers.

Finally, through the great flexibility of the SAS System, three options for Mainframe to Laserjet printing were discussed.

In all, the SAS System turned a fourteen hour Harvard Graphics ordeal into a single submit!

This paper presents just one solution available to the SAS user community. The code demonstrated here is not warranted either explicitly or implicitly.

Notice of Trademark:

Registered Trademarks of SAS Institute Inc. (Cary, NC) include but are not limited to the following: *SAS*, and *SAS/GRAPH*.

Laserjet is a registered trademark of Hewlett Packard Inc.

Harvard Graphics is a registered trademark of Software Publishers Corporation Inc.

```
%macro graph2(no,sort,act,by,name,min,max,text1,text2,t1,t2,t3,foot,r,f1,f2);
```

```
data this;
set all(where=(graphno=&no and grphsort=&sort));
c=&act; act=act*c; ytd=ytd*c;
qtr1=round(qtr1*c,&r);
qtr2=round(qtr2*c,&r);
qtr3=round(qtr3*c,&r);
qtr4=round(qtr4*c,&r);
avg1=round(avg1*c,&r);
avg2=round(avg2*c,&r);
avg3=round(avg3*c,&r);
avg4=round(avg4*c,&r);

data _null_;
set this;
if month=2 then do;
call symput('qtr1',trim(left(qtr1)));
end;
if month=3 then do;
call symput('qtr2',trim(left(qtr2)));
end;

data one;
length function style $ 8 text $15;
retain function 'label' style 'simplex' xsys ysys '2' position '7'
size 2.50 hsys '3';
set this(where=(month=3));
x=month;
text="A = &qtr1";
y=qtr1;

%annotate

symbol1 h=2.0 c=black line=1 w=4 v=triangle i=join ;
symbol2 h=2.0 c=black line=2 w=2 v= i=stepjr;
symbol3 h=2.0 c=black line=1 w=4 v= i=stepjr;

%graphit2
%mend graph2;
```

Example Macro 1.1 - Code for the GRAPH2 Macro.

```
%macro graph4(no,sort,act,by,name,min,max,text1,text2,t1,foot,r,f1,f2);
```

```
data this;
set all(where=(graphno=&no and grphsort=&sort));
c=&act; act=act*c; ytd=ytd*c;
qtr1=round(qtr1*c,&r);
qtr2=round(qtr2*c,&r);
qtr3=round(qtr3*c,&r);
qtr4=round(qtr4*c,&r);
```

```
***** IDENTICAL CODE AS MACRO GRAPH2 *****
```

```
%graphit4
```

```
%mend graph4;
```

Example Macro 1.2 - Partial listing of Macro GRAPH4, Note Parameter change of T2 and T3 compared to Macro GRAPH2.

NOTE: Copyright(c) 1989 by SAS Institute Inc., Cary, NC USA.
 NOTE: SAS (r) Proprietary Software Release 6.07 TS305
 Licensed to INLAND STEEL COMPANY, Site 000223001.

NOTE: Running on IBM Model 3090 Serial Number 371738,
 IBM Model 3090 Serial Number 471738,
 IBM Model 3090 Serial Number 571738.

Welcome to the SAS System, Release 6.07.

```
2300      %graph4(24,1,1,5,pg1a,60,90,Number of,Suppliers,&tot,
MLOGIC(GRAPH4): Beginning execution.
SYMBOLGEN: Macro variable TOT resolves to title f=centb h=3.00 c=black j=c 'Total';
SYMBOLGEN: Some characters in the above value which were subject to macro quoting have been unquoted for printing.
2301      Source: OP Acct Inventory,1,&fnull1,&supp2);
SYMBOLGEN: Macro variable FNULL1 resolves to footnote f=centb h=3.10 c=black j=c '';
SYMBOLGEN: Some characters in the above value which were subject to macro quoting have been unquoted for printing.
SYMBOLGEN: Macro variable SUPP2 resolves to footnote2 f=complex h=1.50 c=black j=| 'Note: Combination
      Sites default to PROCESSOR';
SYMBOLGEN: Some characters in the above value which were subject to macro quoting have been unquoted for printing.
MLOGIC(GRAPH4): Parameter NO has value 24
MLOGIC(GRAPH4): Parameter SORT has value 1
MLOGIC(GRAPH4): Parameter ACT has value 1
MLOGIC(GRAPH4): Parameter BY has value 5
MLOGIC(GRAPH4): Parameter NAME has value pg1a
MLOGIC(GRAPH4): Parameter MIN has value 60
MLOGIC(GRAPH4): Parameter MAX has value 90
MLOGIC(GRAPH4): Parameter TEXT1 has value Number of
MLOGIC(GRAPH4): Parameter TEXT2 has value Suppliers
MLOGIC(GRAPH4): Parameter T1 has value title f=centb h=3.00 c=black j=c 'Total'
MLOGIC(GRAPH4): Parameter FOOT has value Source: OP Acct Inventory
MLOGIC(GRAPH4): Parameter R has value 1
MLOGIC(GRAPH4): Parameter F1 has value footnote f=centb h=3.10 c=black j=c ''
MLOGIC(GRAPH4): Parameter F2 has value footnote2 f=complex h=1.50 c=black j=| 'Note: Combination
      Sites default to PROCESSOR'
```

```
MACROGEN(GRAPH4): DATA THIS;
SYMBOLGEN: Macro variable NO resolves to 24
SYMBOLGEN: Macro variable SORT resolves to 1
MACROGEN(GRAPH4): SET ALL(WHERE=(GRAPHNO=24 AND GRPHSORT=1));
SYMBOLGEN: Macro variable ACT resolves to 1
MACROGEN(GRAPH4): C=1;
MACROGEN(GRAPH4): ACT=ACT*C;
MACROGEN(GRAPH4): YTD=YTD*C;
SYMBOLGEN: Macro variable R resolves to 1
MACROGEN(GRAPH4): QTR1=ROUND(QTR1*C,1);
SYMBOLGEN: Macro variable R resolves to 1
MACROGEN(GRAPH4): QTR2=ROUND(QTR2*C,1);
SYMBOLGEN: Macro variable R resolves to 1
MACROGEN(GRAPH4): QTR3=ROUND(QTR3*C,1);
SYMBOLGEN: Macro variable R resolves to 1
MACROGEN(GRAPH4): QTR4=ROUND(QTR4*C,1);
SYMBOLGEN: Macro variable R resolves to 1
```

NOTE: The data set WORK.THIS has 8 observations and 14 variables.
 NOTE: The DATA statement used 0.10 CPU seconds and 1963K.

```
MACROGEN(GRAPH4): DATA _NULL_;
MACROGEN(GRAPH4): SET THIS;
MACROGEN(GRAPH4): IF MONTH=2 THEN DO;
MACROGEN(GRAPH4): CALL SYMPUT('qtr1',TRIM(LEFT(QTR1)));
MACROGEN(GRAPH4): END;
MACROGEN(GRAPH4): IF MONTH=5 THEN DO;
MACROGEN(GRAPH4): CALL SYMPUT('qtr2',TRIM(LEFT(QTR2)));
MACROGEN(GRAPH4): END;
```

NOTE: The DATA statement used 0.09 CPU seconds and 1976K.

Example Macro 1.3 - Partial Listing of the SAS Log. Invocation of MACRO GRAPH4.

```

%macro annote;
  %if &cmonth=Feb or &cmonth=Mar or &cmonth=Apr %then %do;

data nine ;
  set this(where=(month=-4));
  length function style $ 8 xsys ysys $ 1;
  function= 'move';
  xsys='2';
  ysys='2';
  x=month-.25;
  y=&min;          output;
  function= 'bar';
  xsys='2';
  ysys='2';
  x=month+.25;
  y=yttd;
  line=0;
  style='r2';      output;

```

Example Annotate 1.1 - Partial listing of ANNOTE macro.

```

MLOGIC(ANNOTE): Beginning execution.
SYMBOLGEN: Macro variable CMONTH resolves to May
SYMBOLGEN: Macro variable CMONTH resolves to May
SYMBOLGEN: Macro variable CMONTH resolves to May
MLOGIC(ANNOTE): %IF condition &cmonth=Feb or &cmonth=Mar or &cmonth=Apr is FALSE
SYMBOLGEN: Macro variable CMONTH resolves to May
SYMBOLGEN: Macro variable CMONTH resolves to May
SYMBOLGEN: Macro variable CMONTH resolves to May
MLOGIC(ANNOTE): %IF condition &cmonth=May or &cmonth=Jun or &cmonth=Jul is TRUE
NOTE: The DATA statement used 0.07 CPU seconds and 1976K.

```

```

MACROGEN(ANNOTE): DATA NINE ;
MACROGEN(ANNOTE): SET THIS(WHERE=(MONTH=-3));
MACROGEN(ANNOTE): LENGTH FUNCTION STYLE $ 8 XSYS YSYS $ 1;
MACROGEN(ANNOTE): ;
MACROGEN(ANNOTE): FUNCTION= 'move';
MACROGEN(ANNOTE): XSYS='2';
MACROGEN(ANNOTE): YSYS='2';
MACROGEN(ANNOTE): X=MONTH-.25;
SYMBOLGEN: Macro variable MIN resolves to 60
MACROGEN(ANNOTE): Y=60;
MACROGEN(ANNOTE): OUTPUT;

```

```

NOTE: The data set WORK.NINE has 2 observations and 21 variables.
NOTE: The DATA statement used 0.06 CPU seconds and 1976K.

```

```

MLOGIC(ANNOTE): Ending execution.

```

Example Annotate 1.2 - Partial listing of the SAS Log invoking ANNOTE macro.

```

%else %if &cmmonth=May or &cmmonth=Jun or &cmmonth=Jul %then %do;

    %if &cmmonth=Jun %then %do;
    data _null_;
    set this(where=(month=5));
    %local actual;
    call symput('actual',trim(left(round(act,&r))));

    data current;
    length function style $ 8 text $15;
    retain function 'label' style 'simplex' xsys ysys '2' position '6'
    size 2.50 hsys '3';
    set this(where=(month=5));
    x=month;
    text=" &actual";
    y=act;
    %end;

```

..... Code repeats for &CMONTH variables

Example Annotate 1.3 - Partial listing of ANNOTE macro, demonstrating LABEL annotation.

```

%macro graphit4;
data _null_;
set this end=eof;
retain min &min max &max;
if ((act gt 0) and (act lt min)) then do;
    min=act;
end;
if act gt max then do;
    max=act;
end;
if eof then do;
    %local min max;
    call symput('max',max);
    call symput('min',min);
end;
run;

%if &cmmonth=Feb or &cmmonth=Mar or &cmmonth=Apr %then %do;
proc gplot data=this gout=wmk.disgrf;
axis1 order=(-4 to 6 by 1)
    offset=(4,1)
    value=(f=swiss h=1.5 "&pyear" "&yr2d.Q1" "&yr2d.Q2"
    "&yr2d.Q3" "&yr2d.Q4" 'J' 'F' 'M' 'A' 'M' 'J' )
    color = black
    minor = none
    label = (f=swiss h=2.0 c=black 'Month');

axis2 value = (f=swiss h=1.7 c=black)
    minor = (n=4)
    color = black
    order = &min to &max by &by
label = (f=duplex h=1.7 c=black "&text1" j=r "&text2");

plot act *month = 1
    qtr1 *month = 2
    qtr2 *month = 2
    qtr3 *month = 2
    qtr4 *month = 2 / overlay name="&name" skipmiss frame
    annotate=anno haxis = axis1 vaxis = axis2;
&t1 ; &f1 ; &f2 ;
footnote3 f=complex h=1.5 c=black j=r "&foot"; run;
%end;

```

Example Graphics 1.1

```

%MACRO MULT_2(PLATE=,G1=,G2= );
GOPTIONS DEVICE=HPLJGL2 display
    rotate = portrait
    cback = white
    hsize = 7.5
    vsize = 10.0
    hpos = 75
    vpos = 100
    GACCESS = GSASFILE
    GPROTOCOL= SASGPASC
    HANDSHAKE = XONXOFF
    GSFLen = 64;

PROC GREPLAY
IGOUT=wmk.distgrf
TC=sashelp.templt
TEMPLATE=&PLATE
NOFS;
TREPLAY 1: &G1 2: &G2 ;
RUN;
%MEND MULT_2;

```

Example Template 1.1 - TEMPLATE utilizing default SAS library definition.

```

%macro vert_4(g1=,g2=,g3=,g4=,g5=);
goptions device=hpljgl2 display
    rotate = portrait
    cback = white
    hsize = 7.5
    vsize = 10.0
    hpos = 75
    vpos = 100
    gaccess = gsasfile
    gprotocol= SASGPASC
    handshake = xonxoff
    gsflen = 64;

proc greplay nofs;
tc template ;
tdef vert_4
1/ llx=0    ulx=0    lrx=100   urx=100
   lly=75   uly=100  lry=75    ury=100
2/ clip
   llx=0    ulx=0    lrx=50    urx=50
   lly=47.5 uly=95   lry=47.5 ury=95
3/ clip
   llx=50   ulx=50   lrx=100   urx=100
   lly=47.5 uly=95   lry=47.5 ury=95
4/ llx=0    ulx=0    lrx=50    urx=50
   lly=0     uly=47.50 lry=0     ury=47.50
5/ llx=50   ulx=50   lrx=100   urx=100
   lly=0     uly=47.5  lry=0     ury=47.50 ;
igout=wmk.distgrf;
template vert_4;
treplay 1: &g1 2: &g2 3: &g3 4: &g4 5: &g5;
run;
%mend vert_4;

```

Example Template 1.2 - Customized TEMPLATE.

1. GOPTIONS GSFNAME=DDNAME
2. Download DDNAME's file to hard drive of PC
3. Copy filename.enc to printer

Example:

1.

```
//CHART DD DSN=GOTRF14.CHARTS.CHART
//SAS.SYSIN DD *
GOPTIONS GSFNAME=CHART;
```
2. C: > EXTRA receive d:\filename.enc 'GOTRF14.CHARTS.CHART'
3. D: > copy /b filename.enc lpt1

Example Output 1.1 - Procedure to copy Graphics Stream file to the printer.

```
//SNAGRF JOB XXXTSO,'GM KOVICH',MSGLEVEL=(1,1),PRTY=4,
//      MSGCLASS=T,NOTIFY=GOTRF14
//*MAIN CLASS=ASIST,USER=GOTRF14
//*FORMAT PR,DDNAME=FT12F001,DEST=SNA31,FORMS=DLAN
//*FORMAT PU,DDNAME=GSASFILE,DEST=SNA31PUI
//STEP1 EXEC PROC=SAS,REGION=3000K
//SAS.FT11F001 DD SYSOUT=T
//SAS.FT12F001 DD SYSOUT=R
//SAS.GSASFILE DD SYSOUT=R
//GRAPHS DD DSN=GOTRFXX.SAS607.XXXX.DATABASE,DISP=OLD
//SAS.SYSIN DD *
```

options dquote symbolgen macrogen mlogic mprint;

```
GOPTIONS DEVICE=HPLJGL2
ROTATE=LANDSCAPE
CBACK=WHITE
HSIZE=10.0
VSIZE=7.5
HPOS =100
VPOS =75
GACCESS=GSASFILE
GPROTOCOL=SASGPASC
HANDSHAKE=XONXOFF
GSFLEN=64 ;
```

Example Output 1.2 - JCL and GOPTIONS to print to a converted BARR SNA/RJE printer.