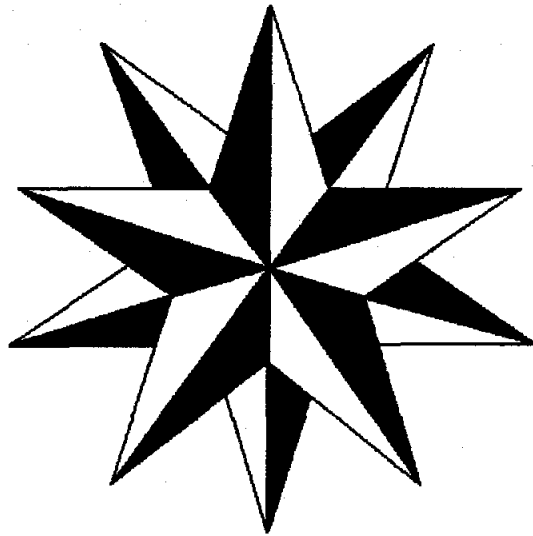


INTELLIGENT PRODUCTION GRAPHIC REPORTING APPLICATIONS



**LeRoy Bessler
Miller Brewing Company**

Intelligent Production Graphic Reporting Applications

LeRoy Bessler, Miller Brewing Company

Abstract

Production graphic reporting applications are those run by a mainframe computer center on a pre-determined schedule, typically getting input data from sources that are computer-updated on a pre-determined schedule.

The formats of the graphs are pre-set. There is no opportunity for ad-hoc manual modification of SAS/GRAPH* programs to update data-date-dependent textual content (e.g., data-date references in titles), custom axis definitions or custom tick-mark assignments which need to change from run to run in a data-sensitive manner (but for which SAS/GRAPH defaults are unacceptable), run-date-dependent selection of input, etc.

Custom macros, as well as standard SAS/GRAPH and SAS* features, can be used to build a self-contained, hands-off graphic reporting machine--which is date-sensitive and data-sensitive, and which respects application-specific graphic design standards.

Why My Concern

I needed to improve a monthly report to executive management. The report included text, tabular listings, and--mostly--graphs. There were several reasons to try to improve it.

Before

The individual graph programs were at risk. Though some used input data that had been computer-updated, many required data to be keyed into the programs. Most required data dates or data-date ranges to be keyed into the graphic software's title statements, and/or into its data-selection logic.

Furthermore, there was a lack of coherence in the report. The diverse pieces were prepared by separate people (the input-data "owners"). This entailed different styles, different formats, different page orientations (portrait vs. landscape), different fonts, different line types and line widths, different personal preferences for graphic software tools, and different hardcopy device types. And some pages were typed!

(Also, with no editorial function designated, there were instances of different nomenclature for the same entity, and could have been instances of the same words used with different definitions implicitly behind them.)

Even report sequence and integrity were at risk, since all the separately-sourced pieces had to be manually collated.

After

I built a hands-off machine that uses automated publishing and graphics intelligence to produce an executive briefing book: structured with exception page, summary page, followed by all the detail graphs, tables, and text. It uses one graphics software (SAS/GRAPH and SAS), and one graphic/typographic printer (an IBM 3820).

Automated publishing--which includes integration of text and graphs, page numbering, figure list, etc.--is accomplished with Document Composition Facility* (DCF). My current desktop for electronic publishing, and any other computer assistance, is an IBM 3090 Model 500J--the mainframe where, at our site, DCF runs. Use of DCF and preparation of graphs in a DCF-compatible format are outside the scope of this paper, but the latter topic has been documented elsewhere.

The role of the former preparers of graphs and tables is to enter data into computer files, using pre-defined formats. Text providers enter it into computer files, which are dynamically imbedded in the report.

There is a single graphics and publishing specialist who adds new pieces to the report from time to time, and who to date has never had to revise the design of the report elements. With one hand in control, consistency is assured and expertise is maximized.

Graphics Intelligence

It is accurate to say that there already is graphics intelligence in SAS/GRAPH default processing. Unfortunately, defaults by definition and design can not address all application-specific needs.

Intelligence can be built into a graphic program, using custom SAS macros, global variables, SAS functions, etc. Intelligence enables the program to change itself in response to changes in the run date, in the data date or data-date range, or in the data range. Whether and how the program changes are a function of application design.

Reusable Building Blocks

The reusable building blocks are SAS macros. There are three varieties (in order of increasing generality):

- (a) application-specific;
- (b) site-specific; and
- (c) general-purpose.

Fire Walls

Any changeable parameters (e.g., the location of a reference line) can be supplied only via control card files, NOT by modifying the hard-coded graphic program.

Data to be depicted is supplied via external files only.

There are no manual date changes. In programs they would compromise program integrity. In control cards they could be forgotten. Data-selection dates are computer-updated.

Years-back ranges (for monthly trend lines) are governed through control cards. Changes to the span of interest are unlikely and infrequent, so that the manual update won't be forgotten if, in fact, it's ever needed.

Powerful Packaging: Standard Graphic Templates

Reliance on a few site-standard and application-standard graphic templates (not to be confused with SAS/GRAPH's TEMPLATES for PROC GREPLAY) facilitates a consistent, parallel presentation.

The presentation formats become familiar. There is no need for page-to-page "adjustment of the perception/interpretation mechanism". The reader/viewer knows what to expect and where to find things quickly. As the use of the templates proliferates through ad-hoc or other production applications, adjustment becomes unnecessary even from report to report.

Among the side benefits of acceptance of standard templates is reduction of the time and effort required to develop new, and to enhance existing, graphic applications.

Data Currency and Integrity

By, after automatic extraction from the data, displaying the data date or the data-date range in a title, a means is provided for the reader/viewer to quickly ascertain the currency and completeness of the data.

Scheduled production jobs assure currency of some of the computer-updated data. Some data providers may be responsible for running personally-controlled jobs to provide other of the computer-updated data. Some data providers must manually key data. In every case, the data date is required on each record.

In an environment where every data record is dated, plan ahead. Even if only last month's data must be graphed as a bar chart, be sure to keep all prior months' data. Anticipate the almost inevitable future request for trend analysis.

Appropriate Business Uses of Graphs

The two best business uses of graphs are:

- to visually display the relative size of measurements, rather than only tabulating them
- to visually display temporal trends

I will concentrate on examples of each of these.

The Perils of Pie Charts

Pie charts are extremely popular for presentations.

However, though presentation graphs can be created ad-hoc, "hands on", and iteratively, production graphs must get it right the first time, every time.

Consider the pie chart in Figure 1. It is supposed to display the relative size (and list the magnitude) of the population of the twelve countries in the European Community.

If you have, say, 5 or 6 entities whose pie slices are approximately the same size, all the slices will be apparent, and SAS/GRAPH will have no problem putting down all the slice labels, with adequate white space between them. For Figure 1, this is clearly not the case.

For some applications, lumping small slices into an "Other" category is unacceptable. For a production "hands-off" application to have a slice alternately appear and disappear from month to month, depending on size, is irritating, if not unacceptable. Furthermore, if you put so many slices into "Other" that it becomes non-negligible, there will be questions as to its detailed content.

Finally, with the vagaries of SAS/GRAPH's pie-chart text handling, sizes and positions of slices may be such that some labels are poorly positioned (and very likely to have some adjacent pairs squashed together in an ugly manner if you request labels for PERCENT in addition to SLICE and VALUE), if they don't disappear completely.

So, Consider the Bread Chart

The solution--if you want a visual display of the relative size, the value, and the percent of whole, with reliably positioned and never suppressed text, and in a quickly digested format--is the Bread Chart. See Figure 3. The Bread Chart is like a very long (what we in America call) French bread cut into chunks of different lengths, which are then laid parallel, and largest to smallest, rather than end to end. (In view of my choice of solid, rather than empty, fill, it looks more like black bread than French bread.)

Figures 2 and 4 show the SAS/GRAPH programs used to produce the unsuccessful pie chart in Figure 1 and the successful bread chart in Figure 3, respectively.

All the macros used in the bread chart program are listed in the Appendix. The potential benefit of the TXTPRMS macro invoked by HBAR1D, and of the seemingly complicating macros for TITLES and FOOTNOTES, is that they make it easy to define an application-default (not a SAS-default) font, height, etc. for text parts of graphs. Obviously, these text-oriented macros are not essential to the success of the bread chart concept. They only make it easier to achieve consistent application-specific customization of text from graph to graph.

There are two reasons for (through automation) building the percent-of-whole as part of the bar label, rather than asking the SAS/GRAPH HBAR function to compute and list the percent along the right margin (via options PERCENT and FREQ=response-variable-name): (1) SAS/GRAPH inflicts two decimal positions whether or not you want them; and (2) you have no control over height, font, and upper-case-vs.-mixed-case for the PERCENT column heading. One rarely needs two decimal positions on a percent to make a business decision. All-upper-case is an ugly hold-over from the bad old days of early data processing, and not representative of standard business communication style. (As a matter of fact, paragraphs in all upper case type are harder to read.)

The bread chart answers that all too common question, "How is the loaf divided up?"

Trend Lines

Trend lines are a common application. Here I show only the case of monthly graphs and a single trend line. Other periodicities are handled analogously. Multiple trend lines on a graph is a minor, straightforward extension.

The features and benefits of the YMPLOT1 macro are best presented by simply referring you to the graphic examples in Figures 5 and 7, their associated SAS/GRAPH programs in Figures 6 and 8, and the underlying macros listed in the Appendix. Many of the macros carry comments on the MACRO statement. Each macro is discussed in text at the bottom of its exhibit.

For explanation of macro syntax, global variables, SYMPUT, etc., I refer the reader to SAS-Institute-provided documentation. The principles and techniques used in my macros are by no means arcane.

I will close with comments on some aspects of the design of YMPLOT1.

Specifying TSTRTMM = 1 and TSTOPMM = 12 allows one to retain, in the example of Figure 8, a January 1988 to December 1990 time axis throughout 1990, as progressively more data becomes available. The framework of the graph remains constant month to month. Alternatively, if one omits both TSTRTMM and TSTOPMM, but maintains the YRSFILE control card value at, in this example, 2, the time axis remains 25 months wide every month, ending at the current report-month.

Fixing VMIN and VMAX likewise assures that the framework of the graph remains constant month to month. Moreover, VMIN = 0 diminishes the likelihood of undue concern about what could, in fact, really be minor fluctuations month to month. In this example, where the units of the response variable are percent, fixing the axis maximum at the unequivocally maximum possible response is an eminently reasonable standardization.

The benefit of the January reference lines coupled with the markers for the data points is obvious. The picture would be more balanced if the first January reference line were omitted; and there should be no ambiguity caused by its omission. I guess that's a possible future enhancement. Also, I look forward to using the less distracting option of dotted lines in Version 6 of SAS/GRAPH.

Notice

SAS/GRAPH and SAS are registered trademarks of SAS Institute Inc., Cary, NC, USA. DCF (Document Composition Facility) is a product of IBM Corp.

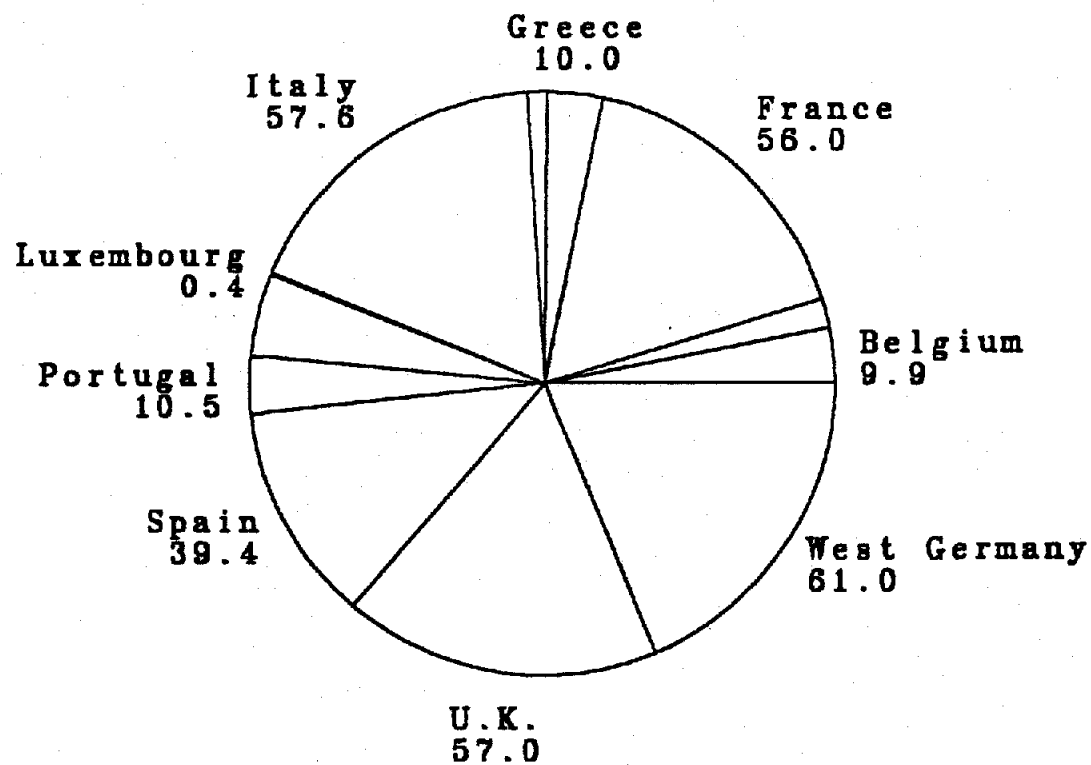
Products are specifically qualified as Release 5.18 of SAS/GRAPH and SAS.

The SAS code included in this paper was tested, and I believe it to be reliable. However, for publication, most of the code was rewritten and reorganized for conciseness and clarity. So it doesn't have the long track record and wide range of exercise that the production version has. In any case, it can only be presented on an "as is" basis. Any code you adopt for your own use should be tested by you, and you must assume responsibility for the consequences of its use.

Author

Dr. LeRoy Bessler
Miller Brewing Company
P. O. Box 482
Milwaukee, Wisconsin 53201-0482
414-931-2773

Population in the European Community
By Country, 1989
(in Millions)



Source: "The World Factbook 1989"

Figure 1. Unsuccessful Pie Chart: Should have twelve labels. See Figure 2 for blameless program.

```

/* put device-specific GOPTIONS here */

GOPTIONS SIMFONT=TRIPLEXU; /* These GOPTIONS control FONT assignment */
GOPTIONS NOCHARACTERS;    /* for all text which either has F=NONE */
GOPTIONS NOCELL;          /* (by default or explicitly), or cannot */
GOPTIONS HPOS=55;         /* have the F= parameter used for it. */
                          /* The latter is true for parts of PIE. */
                          /* The value 55 here is device-specific. */
                          /*******/

OPTIONS DQUOTE;
DATA;
INFILE INDATA;
INPUT @1 COUNTRY $12.
      @18 POP      4.1;

RUN;
PROC GCHART;
PIE COUNTRY / SUMVAR      = POP
               DISCRETE
               NOHEADING
               OTHER      = 0
               SLICE      = OUTSIDE
               VALUE      = OUTSIDE;

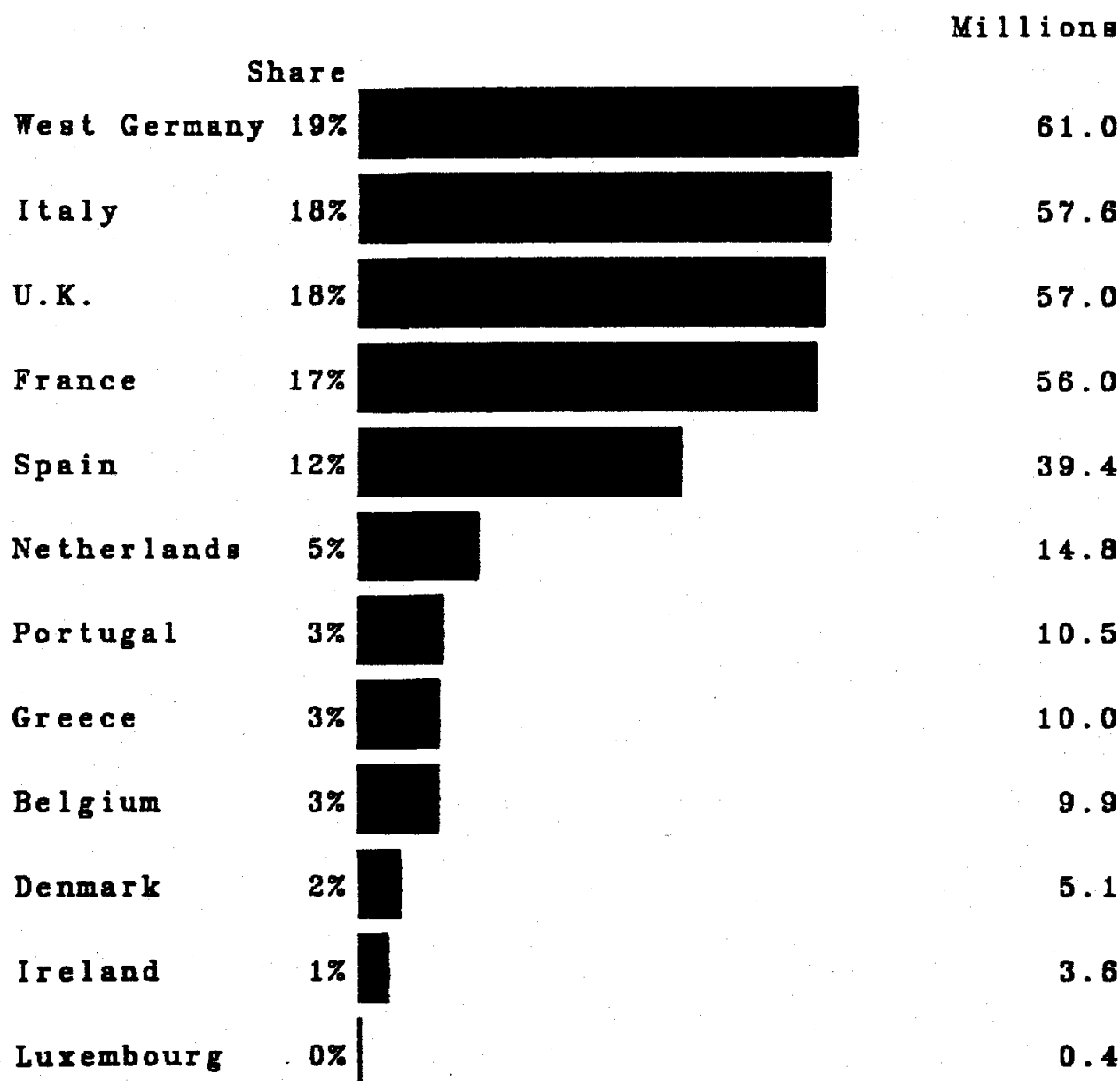
TITLE1 H=1.5 F=TRIPLEX
       'Population in the European Community';
TITLE2 H=1.5 F=TRIPLEX
       'By Country, 1989';
TITLE4 H=1.5 F=TRIPLEX
       '(in Millions)';
FOOTNOTE1 H=1 F=TRIPLEX
          'Source: "The World Factbook 1989"';

RUN;

```

Figure 2. SAS/GRAPH Program for Unsuccessful Pie Chart: See Figure 1 for graph.

Population in the European Community By Country, 1989



Source: "The World Factbook 1989"

Figure 3. Bread Chart: Shares ranked by size; names, percents, and values shown. See Figure 4 for program.

```

/* put device-specific GOPTIONS here */

GOPTIONS SIMFONT=TRIPLEXU; /* *****/
GOPTIONS NOCHARACTERS; /* These GOPTIONS control FONT assignment */
GOPTIONS NOCELL; /* for all text which either has F=NONE */
GOPTIONS HPOS=55; /* (by default or explicitly), or cannot */
/* have the F= parameter used for it. */
/* The latter is true for parts of HBAR. */
/* The value 55 here is device specific. */
/* *****/

OPTIONS DQUOTE;
DATA TOPCT;
INFILE INDATA;
INPUT @1 COUNTRY $12.
      @18 POP 4.1;
RUN;
%APNDPCT(DATA=TOPCT,
        MIDPOINT=COUNTRY,
        RESPONSE=POP,
        OUT=TOCHART,
        OUTNAME=COUNTRY,
        OUTVALUE=POP);
%HBAR1D(DATA=TOCHART,
        MIDPOINT=COUNTRY,
        RESPONSE=POP,
        MPTLABEL='Share',
        RSPLABEL='Millions',
        HTTL=1.5,
        FTTLFT=TRIPLEX);
%TTL(TTLN0=1,
     TEXT='Population in the European Community');
%TTL(TTLN0=2,
     TEXT='By Country, 1989');
%FOOT(FOOTN0=1,
     TEXT='Source: "The World Factbook 1989"');
RUN;

```

Figure 4. SAS/GRAPH Program for Bread Chart: See Figure 3 for graph.

Demand on Facility, in Percent of Capacity
By Month, May 1989 to May 1990

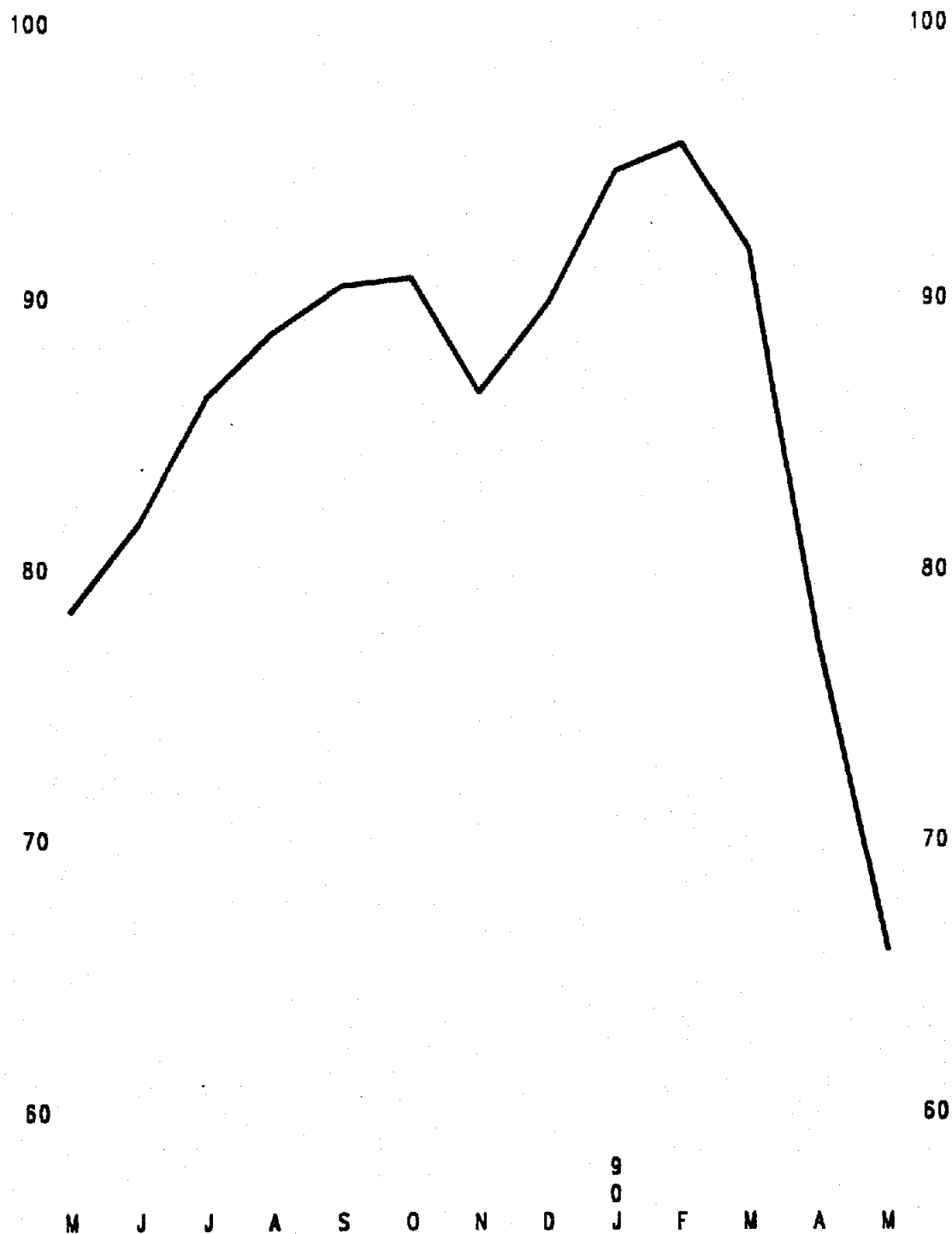


Figure 5. Plot by Year-Month, Using YMPLOT1 Macro with Defaults: See Figure 6 for program.

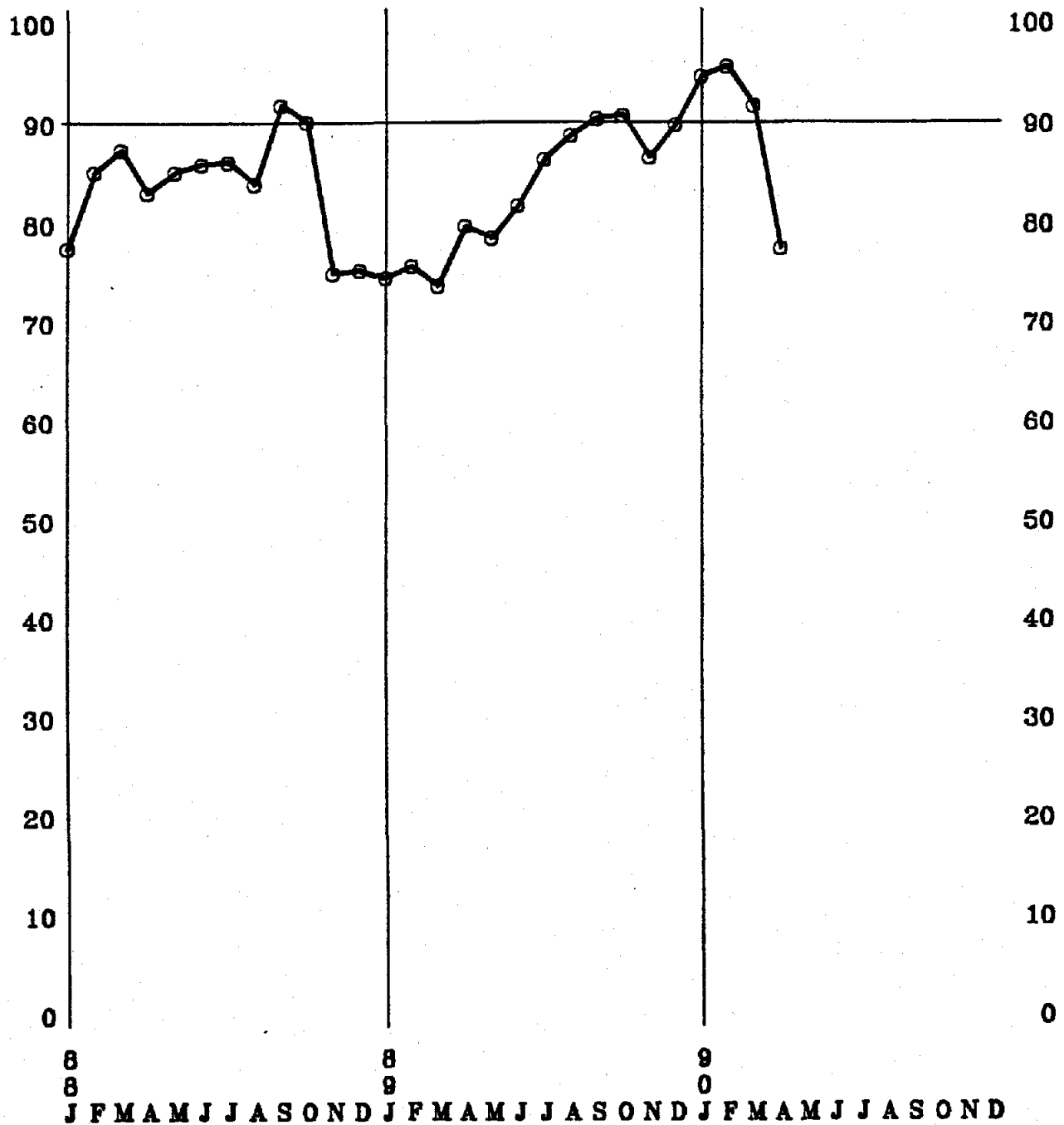
```

/* put GOPTIONS here */
OPTIONS DQUOTE;
DATA TOSELECT;
INFILE INDATA;
INPUT @1 YYMM      $4.
      @18 MEASURE  5.1;
RUN;
%YMPLLOT1(TICKSTEP=10,
          DATA=TOSELECT,
          VAR=MEASURE);
%TTL(TTLNO=1,
     TEXT='Demand on Facility, in Percent of Capacity');
%TTLBYMO(TTLNO=2);
RUN;

```

Figure 6. SAS/GRAPH Program for Plot with Macro Defaults: See Figure 5 for graph.

Demand on Facility, in Percent of Capacity By Month, January 1988 to April 1990



90% is threshold for adverse impact on service

Figure 7. Plot by Year-Month, Using YMPLOT1 Macro with Many Options: See Figure 8 for program.

```

/* put GOPTIONS here */
OPTIONS DQUOTE;
DATA TOSELECT;
INFILE INDATA;
INPUT @1 YYMM      $4.
      @18 MEASURE  5.1;
RUN;
%MPLOT1(TICKSTEP=10,
        DATA=TOSELECT,
        VAR=MEASURE,
        RDATAFILE=RPTYMM,
        YRSFILE=YRSBACK,
        REFFILE=REFLINE,
        SUMFILE=SUMMARY,
        JANREFS=YES,
        MARKER=YES,
        DSTRMM=1,
        TSTRMM=1,
        TSTOPMM=12,
        VMIN=0,
        HTTL=1.5,
        FTTLFT=TRIPLEX,
        FTXT=TRIPLEX);
%TTL(TTLNO=1,
      TEXT='Demand on Facility, in Percent of Capacity');
%TTLBYMO(TTLNO=2);
%FOOTREF(FOOTNO=1,
          TEXT='% is threshold for adverse impact on service');
RUN;

```

Figure 8. SAS/GRAPH Program for Plot with Use of Many Options: See Figure 7 for graph.

Appendix - Macros for Intelligent Graphic Reporting Applications

```
%MACRO APNDPCT
  (DATA=,          /* input file          */
   MIDPOINT=,      /* midpoint variable    */
   RESPONSE=,      /* response variable    */
   OUT=,           /* output file          */
   OUTNAME=,       /* output midpoint variable name */
   OUTVALUE=); /* output response variable name */

DATA;
SET &DATA;
FORMAT NAMSHORT $12.;
NAMSHORT = &MIDPOINT;
KEEP NAMSHORT &RESPONSE;
RUN;
PROC SORT OUT=SORTED; BY NAMSHORT;
RUN;
PROC FREQ DATA=SORTED;
TABLE NAMSHORT / OUT=PERCENTS NOPRINT;
WEIGHT &RESPONSE;
RUN;
DATA;
SET PERCENTS;
FORMAT PCT 2.;
PCT=ROUND(PERCENT,1);
FORMAT &MIDPOINT $16.;
IF PCT < 10 THEN
  &MIDPOINT = NAMSHORT || ' ' || LEFT(PCT) || '%';
ELSE
IF PCT < 100 THEN
  &MIDPOINT = NAMSHORT || ' ' || LEFT(PCT) || '%';
ELSE
  &MIDPOINT = NAMSHORT || LEFT(PCT) || '%';
RUN;
PROC SORT OUT=PCTS; BY NAMSHORT;
RUN;
DATA &OUT;
MERGE SORTED PCTS; BY NAMSHORT;
KEEP &MIDPOINT &RESPONSE;
RENAME &MIDPOINT=&OUTNAME &RESPONSE=&OUTVALUE;
RUN;
%MEND;
```

APNDPCT: Determine percent-of-total for the response variable for each midpoint, and append that percent at the end of the midpoint description.

```

%MACRO DATRANG
  (DATA=,DSTRTM=.,TSTRTM=.,TSTOPMM=.); /* %rptdate must precede */
%GLOBAL DSTRTY; /* data start year */
%GLOBAL DSTRTM; /* data start month */
%GLOBAL DSTRTYM; /* data start year-month */
%GLOBAL DSTOPY; /* data stop year */
%GLOBAL DSTOPM; /* data stop month */
%GLOBAL DSTOPYM; /* data stop year-month */
%GLOBAL TSTRTY; /* tick start year */
%GLOBAL TSTRTM; /* tick start month */
%GLOBAL TSTOPY; /* tick stop year */
%GLOBAL TSTOPM; /* tick stop month */
DATA _NULL_;
%IF &DATA ^= . %THEN %DO;
  INFILE &DATA;
  INPUT YRSBACK 2.;
  DSTRTY = &RPTY - YRSBACK;
%END;
%ELSE %DO;
  DSTRTY = &RPTY - 1;
%END;
CALL SYMPUT('DSTRTY',LEFT(DSTRTY));
CALL SYMPUT('TSTRTY',LEFT(DSTRTY));
%IF &DSTRTM ^= . THEN
  DSTRTM = &DSTRTM;
ELSE
  DSTRTM = &RPTMM;
CALL SYMPUT('DSTRTM',LEFT(DSTRTM));
%IF &TSTRTM ^= . AND &TSTRTM -> DSTRTM THEN
  CALL SYMPUT('TSTRTM',&TSTRTM);
ELSE
  CALL SYMPUT('TSTRTM',LEFT(DSTRTM));
CALL SYMPUT('DSTRTYM',LEFT((DSTRTY * 100) + DSTRTM));
CALL SYMPUT('DSTOPY',&RPTY);
CALL SYMPUT('DSTOPM',&RPTMM);
CALL SYMPUT('DSTOPYM',&RPTYMM);
CALL SYMPUT('TSTOPY',&RPTY);
%IF &TSTOPMM ^= . AND &TSTOPMM -< &RPTMM THEN
  CALL SYMPUT('TSTOPM',&TSTOPMM);
ELSE
  CALL SYMPUT('TSTOPM',&RPTMM);
RUN;
%MEND;

```

DATRANG: Determine start and stop values for year-month tick marks, and override start value, if any, for year-month input data selection. Assign those values to global variables. Use retrieved report-year-month global variables as appropriate.

```

%MACRO FOOT(FOOTNO=,TEXT=);
FOOTNOTE&FOOTNO
  H=&HFOOT F=&FTIFO &TEXT;
%MEND;

```

FOOT: Define FOOTNOTE number and text. Retrieve global variables for HEIGHT and FONT.

```

%MACRO FOOTREF(FOOTNO=,TEXT=);
FOOTNOTE&FOOTNO
  H=&HFOOT F=&FTIFO "&REFVAL"&TEXT;
%MEND;

```

FOOTREF: Define FOOTNOTE number and description of reference value. Retrieve global variables for HEIGHT, FONT, and reference value to be included in text.

```

%MACRO GETJREF(DOGET=);
%IF &DOGET=YES %THEN %DO;
DATA _NULL_;
%GLOBAL LASTJREF;
%GLOBAL %VARLIST(PREFIX=JREF,VARCOUNT=11); /* max is 10 yrs, Jan-Jan */
%IF &TSTRTM > 1 %THEN
  %LET BEGINYY = &TSTRTY + 1;
%ELSE
  %LET BEGINYY = &TSTRTY;
I = 0;
DO Y = &BEGINYY TO &TSTOPY;
  I = I + 1;
  CALL SYMPUT ('LASTJREF',I);
  CALL SYMPUT ('JREF'||LEFT(I),LEFT((Y * 100) + 1));
END;
RUN;
%END;
%MEND;

```

GETJREF: Assign global variables for January year-month values to use for vertical reference lines.

```

%MACRO GETREF(DATA=);
%IF &DATA != ' ' %THEN %DO;
DATA _NULL_;
INFILE &DATA;
INPUT REFVAR;
%GLOBAL REFVAL;
CALL SYMPUT('REFVAL',LEFT(REFVAR));
RUN;
%END;
%MEND;

```

GETREF: Access file, if any, to assign global variable for value to use for response reference line.

```

%MACRO GETRTKS;
DATA _NULL_;
%GLOBAL LASTRTK;
%GLOBAL %VARLIST(PREFIX=RTK,VARCOUNT=100) ;
I = 0;
DO J = &AXISTART TO &AXISEND BY &AXISSTEP;
  I = I + 1;
  CALL SYMPUT ('LASTRTK',I);
  CALL SYMPUT ('RTK'||LEFT(I),LEFT(J));
END;
RUN;
%MEND;

```

GETRTKS: Assign global variables to be used later for the VALUE parameter specifications for the response axis. (To control this, retrieve global variables for start, end, and increment of response axis.)

```

%MACRO HBARID
    (DATA=,          /* input file */
    MIDPOINT=,       /* name of midpoint variable */
    RESPONSE=,       /* name of response variable */
    MPTLABEL=' ',   /* 16-char label of MIDPOINTS */
    RSPLABEL=' ',   /* 16-char label of RESPONSES */
    FILL=SOLID,      /* fill for bars - EMPTY/SOLID */
    HTTL=1,          /* height for all %ttl__ macros */
    FTTLFT=NONE,     /* font for all %ttl__ & %foot__ macros */
    HFT=1);          /* height for all %foot__ macros */
%TXTPRMS(HTTL=&HTTL,FTTLFT=&FTTLFT,HTXT=1,FTXT=NONE,HFT=&HFT);
PROC GCHART DATA=&DATA;
%IF &FILL = SOLID %THEN %DO;
    PATTERN1 V=SOLID C=BLACK;
%END;
%ELSE %DO;
    PATTERN1 V=EMPTY C=BLACK;
%END;
HBAR &MIDPOINT /
    SUM
    SUMVAR = &RESPONSE
    MAXIS = AXIS1
    RAXIS = AXIS2
    DESCENDING
    DISCRETE;
AXIS1 MAJOR = NONE
    MINOR = NONE
    STYLE = 0;
AXIS2 LABEL = NONE
    VALUE = NONE
    MAJOR = NONE
    MINOR = NONE
    STYLE = 0;
LABEL &MIDPOINT = &MPTLABEL;
LABEL &RESPONSE = &RSPLABEL;
%MEND;

```

HBARID: Create a horizontal bar chart, with midpoint values ordered by descending sequence of value of response variable. Options are explained in comments on MACRO statement.

```
%MACRO MAKLIST(ITEM=);
%DO I = 1 %TO &&LAST&ITEM;
  "&&&ITEM&I"
%END;
%MEND;
```

MAKLIST: Retrieve the global variables to develop the list of values to be assigned to a parameter.

```
%MACRO MARKER(MARKER=);
DATA _NULL_;
%GLOBAL V;
%IF &MARKER=YES %THEN %DO;
  CALL SYMPUT('V','');
%END;
%ELSE %DO;
  CALL SYMPUT('V','NONE');
%END;
RUN;
%MEND;
```

MARKER: If a plot-point marker was asked for, assign the standard marker to a global variable. Else, assign no marker as the default.

```
%MACRO PLTILIN(DATA=,PLOTVVAR=,PLOTTHVAR=,VAXISNO=,HAXISNO=);
SYMBOL1 I=JOIN V=&V L=1 W=4 C=BLACK;
PROC GPLOT DATA=&DATA;
PLOT &PLOTVVAR*&PLOTTHVAR /
  HAXIS=AXIS&HAXISNO
  VAXIS=AXIS&VAXISNO
%IF &PLOTTHVAR = YMM AND &JANREFS = YES %THEN %DO;
  HREF = %MAKLIST(ITEM=JREF)
  LHREF = 1
%END;
%IF &REFFILE != ' ' %THEN %DO;
  VREF = &REFVAL
  LVREF = 1
%END;
}
SYMBOL2 V=NONE C=BLACK;
PLOT2 &PLOTVVAR*&PLOTTHVAR=2 / VAXIS=AXIS&VAXISNO;
%MEND;
```

PLTILIN: Do a single-line plot, using line width 4. Provide tick marks at the right side. Include a reference line for the response variable, if a reference value is provided. If the midpoint variable is year-month, provide reference lines at each January, if requested.

```
%MACRO RAXIS(RAXISNO=);
AXIS&RAXISNO LABEL = NONE
  MAJOR = NONE
  MINOR = NONE
  STYLE = 0
  ORDER = &AXISSTART TO &AXISEND BY &AXISSTEP
  VALUE = ( H=&HTEX F=&FTEX %MAKLIST(ITEM=RTK) );
%MEND;
```

RAXIS: Retrieve global variables to provide the response-axis definition.

```

%MACRO RPTDATE(DATA=);
%GLOBAL RPTYMM;
%GLOBAL RPTY;
%GLOBAL RPTMM;
%IF &DATA ^= ' ' %THEN %DO;
  DATA _NULL_;
  INFILE &DATA;
  INPUT @01 RPTYMM $4.
        @01 RPTY $2.
        @03 RPTMM $2.;
  CALL SYMPUT('RPTYMM',RPTYMM);
  CALL SYMPUT('RPTY',RPTY);
  CALL SYMPUT('RPTMM',RPTMM);
  RUN;
%END;
%ELSE %DO;
  DATA _NULL_;
  YR=YEAR(TODAY());
  MO=MONTH(TODAY());
  IF MO > 1 THEN DO;
    SELECTMM = MO - 1;
    SELECTYY = YR - 1900;
  END;
  ELSE DO;
    SELECTMM = 12;
    SELECTYY = YR - 1901;
  END;
  SLCTYYMM = (SELECTYY * 100) + SELECTMM;
  CALL SYMPUT('RPTYMM',SLCTYYMM);
  CALL SYMPUT('RPTY',SELECTYY);
  CALL SYMPUT('RPTMM',SELECTMM);
  RUN;
%END;
%MEND;

```

RPTDATE: Assign global variables for the report date--as year-month, year, and month. Use input file, if any. Else, assume month previous to current month is desired.

```

%MACRO RXSPRMS(DATA=,VAR=,STEP=,AXISMIN=.,AXISMAX=.);
%GLOBAL AXISTART;
%GLOBAL AXISEND;
%GLOBAL AXISSTEP;
PROC MEANS DATA=&DATA NOPRINT MIN MAX;
VAR &VAR;
OUTPUT OUT=MINMAX MIN=MINVAR MAX=MAXVAR;
RUN;
DATA _NULL_;
SET MINMAX;
CALL SYMPUT('AXISSTEP',&STEP);
FORMAT AXISTART 10.;
AXISTART = MINVAR - MOD(MINVAR,&STEP);
IF (&AXISMIN ^= . AND AXISTART >= &AXISMIN) THEN AXISTART = &AXISMIN;
CALL SYMPUT('AXISTART',AXISTART);
FORMAT AXISEND 10.;
AXISEND = MAXVAR + &STEP - MOD(MAXVAR,&STEP);
IF (&AXISMAX ^= . AND AXISEND <= &AXISMAX) THEN AXISEND = &AXISMAX;
CALL SYMPUT('AXISEND',AXISEND);
RUN;
%MEND;

```

RXSPRMS: Assign global variables for start, end, and increment of response axis. Invoker must supply increment. Invoker may specify start and/or end. Else, axis start or end is determined based on actual data range, and is placed at closest point that increment permits.

```

%MACRO SAVLAST(DATA=,VAR=,OUT=);
%IF &OUT = ' ' %THEN %DO;
DATA _NULL_;
SET &DATA END=LAST;
RETAIN VALUE 'N. A.';
IF YYMM = &RPTYMM THEN VALUE = &VAR;
IF LAST;
FILE &OUT;
PUT @1 VALUE;
RUN;
%END;
%MEND;

```

SAVLAST: Not for graph. Used to save number for all-graphs summary of report month elsewhere.

```

%MACRO TTL(TTLNO=,TEXT=);
TITLE&TTLNO H=&HTITLE F=&FTIFO &TEXT;
%MEND;

```

TTL: Define TITLE number and text. Retrieve global variables for HEIGHT and FONT.

```

%MACRO TTLBYMO(TTLNO=);
TITLE&TTLNO H=&HTITLE F=&FTIFO
  "By Month, &FIRSTMO &FIRSTYR to &LASTMO &LASTYR";
%MEND;

```

TTLBYMO: Define TITLE number. Retrieve global variables for HEIGHT, FONT, and text (dates).

```

%MACRO TTLDATS(DATA=);
%GLOBAL FIRSTMO;
%GLOBAL FIRSTYR;
%GLOBAL LASTMO;
%GLOBAL LASTYR;
PROC SORT DATA=&DATA OUT=SORTED; BY YYMM;
DATA _NULL_;
SET SORTED;
IF _N_ = 1;
FORMAT FIRSTMM 2.;
FIRSTMM = SUBSTR(YYMM,3,2);
FIRSTMO = PUT(FIRSTMM,MONTHUL.);
CALL SYMPUT('FIRSTMO',TRIM(FIRSTMO));
FIRSTYR = SUBSTR(YYMM,1,2) + 1900;
CALL SYMPUT('FIRSTYR',LEFT(FIRSTYR));
RUN;
PROC SORT DATA=&DATA OUT=SORTED; BY DESCENDING YYMM;
DATA _NULL_;
SET SORTED;
IF _N_ = 1;
FORMAT LASTMM 2.;
LASTMM = SUBSTR(YYMM,3,2);
LASTMO = PUT(LASTMM,MONTHUL.);
CALL SYMPUT('LASTMO',TRIM(LASTMO));
LASTYR = SUBSTR(YYMM,1,2) + 1900;
CALL SYMPUT('LASTYR',LEFT(LASTYR));
RUN;
%MEND;

```

TTLDATS: Assign global variables for two month-years to define the data-date range for use in a title. Prepare the month names with a site-specific mixed-case format.

```

%MACRO TXTPRMS
  (HTTL=1,      /* height for all %ttl__ macros      */
   FTTLFT=NONE, /* font for all %ttl__ & %foot__ macros      */
   HTXT=1,     /* height for other text if H can be specified */
   FTXT=NONE,  /* font for other text if F can be specified */
   HFT=1);    /* height for all %foot__ macros      */
DATA _NULL_;
%GLOBAL HTITLE;
CALL SYMPUT('HTITLE',LEFT(&HTTL));
%GLOBAL FTIFO;
CALL SYMPUT('FTIFO',"&FTTLFT");
%GLOBAL HTEX;
CALL SYMPUT('HTEX',LEFT(&HTXT));
%GLOBAL FTEX;
CALL SYMPUT('FTEX',"&FTXT");
%GLOBAL HFOOT;
CALL SYMPUT('HFOOT',LEFT(&HFT));
RUN;
%MEND;

```

TXTPRMS: Options are explained in comments on MACRO statement.

```

%MACRO VARLIST(PREFIX=,VARCOUNT=);
%DO I = 1 %TO &VARCOUNT;
  &PREFIX&I
%END;
%MEND;

```

VARLIST: Assign global variables for a list of values to be used later in specification of a parameter.

```

%MACRO YMHAXIS(YMHAXSNO=);
AXIS&YMHAXSNO LABEL = NONE
  MAJOR = NONE
  MINOR = NONE
  STYLE = 0
  ORDER = %MAKLIST(ITEM=YYMM)
  VALUE = ( A=-90 R=90 H=&HTEX F=&FTEX %MAKLIST(ITEM=HTK) );
%MEND;

```

YMHAXIS: Retrieve global variables to provide the year-month axis definition. The axis must be horizontal--though axis definition is incapable of forcing that--for the tick marks to be useful, given their design (see an example for results).

```

%MACRO YMHTKS;
DATA _NULL_;
%GLOBAL LASTHTK ;
%GLOBAL LASTYYMM;
%GLOBAL %VARLIST(PREFIX=HTK,VARCOUNT=121) ; /* max is 10 yrs, Jan-Jan */
%GLOBAL %VARLIST(PREFIX=YYMM,VARCOUNT=121); /* max is 10 yrs, Jan-Jan */
I = 0;
DO Y = &TSTRTY TO &TSTOPY;
  IF Y = &TSTRTY THEN
    DO M = &TSTRTM TO 12;
      %YMHTMAK;
    END;
  ELSE
    IF Y = &TSTOPY THEN
      DO M = 1 TO &TSTOPM;
        %YMHTMAK;
      END;
    ELSE
      DO M = 1 TO 12;
        %YMHTMAK;
      END;
    END;
  END;
END;
RUN;
%MEND;

```

YMHTKS: Assign global variables to be used for the VALUE and ORDER parameter specifications for the year-month axis. (Retrieve global variables for tick-mark start and stop in order to set bounds.)

```

%MACRO YMHTMAK;
I = I + 1;
CALL SYMPUT ('LASTHTK',I);
CALL SYMPUT ('LASTYYMM',I);
MO = PUT(M,MOINITL.);
IF M = 1 THEN YYM = PUT((LEFT(Y) || MO),$CHAR3.);
ELSE YYM = PUT((' ' || MO),$CHAR3.);
CALL SYMPUT ('HTK' || LEFT(I),YYM);
YYMM = (Y * 100) + M;
CALL SYMPUT ('YYMM' || LEFT(I),LEFT(YYMM));
%MEND;

```

YMHTMAK: Used by the YMHTKS macro. Uses a site-specific format for initial character of month-name.

```

%MACRO YMPLLOT1
  (TICKSTEP=, /* increment between vertical tick marks */
  DATA=, /* input data */
  VAR=, /* variable to be plotted vs. year-month */
  RDATAFILE=' ', /* ddname of file to specify last year-month
  YRSFILE=' ', /* ddname of file to specify number of years
  REFFILE=' ', /* ddname of file to specify reference line
  SUMFILE=' ', /* ddname of file to output value for last
  JANREFS=NO, /* reference lines at each January? YES/NO
  MARKER=NO, /* circle with dot at each data point? YES/NO
  DSTRTMM=., /* override for month number of first year-month
  TSTRTMM=., /* override to extend year-month axis
  TSTOPMM=., /* override to extend year-month axis
  VMIN=., /* override to start vertical axis below
  VMAX=., /* override to end vertical axis above
  HTTL=1, /* height for all %ttl__ macros
  FTTLFT=NONE, /* font for all %ttl__ & %foot__ macros
  HTXT=1, /* height for other text if H can be specified
  FTXT=NONE, /* font for other text if F can be specified
  HFT=1); /* height for all %foot__ macros

%RPTDATE(DATA=&RDATAFILE);
%DATRANG
  (DATA=&YRSFILE,DSTRTMM=&DSTRTMM,TSTRTMM=&TSTRTMM,TSTOPMM=&TSTOPMM);
%GETJREF(DOGET=&JANREFS);
%YMHTKS;
DATA TOPLOT;
SET &DATA;
IF YYMM >= &DSTRTYM;
IF YYMM <= &DSTOPYM;
RUN;
%SAVLAST(DATA=TOPLOT,VAR=&VAR,OUT=&SUMFILE);
%RXSPRMS(DATA=TOPLOT,VAR=&VAR,
  AXISMIN=&VMIN,AXISMAX=&VMAX,STEP=&TICKSTEP);
%GETRTKS;
%TTLDATS(DATA=TOPLOT);
%GETREF(DATA=&REFFILE);
%MARKER(MARKER=&MARKER);
%TXTPRMS(HTTL=&HTTL,FTTLFT=&FTTLFT,
  HTXT=&HTXT,FTXT=&FTXT,HFT=&HFT);
%PLT1LIN(DATA=TOPLOT,PLOTVVAR=&VAR,PLOTHVAR=YYMM,
  VAXISNO=1,HAXISNO=2);
%RAXIS(RAXISNO=1);
%YMHAXIS(YMHAXISNO=2);
%MEND;

```

YMPLLOT1: Plot one variable vs. year-month. Options are explained in comments on MACRO statement.