C H A P T E R

1

Introduction to SAS/GRAPH Software

Overview

SAS/GRAPH software is the information and presentation graphics component of the SAS System. As a data analysis and presentation tool, SAS/GRAPH software

□ analyzes your data and visually represents the relationship between data values as two- and three-dimensional graphs, including charts, plots, and maps.

□ enhances the appearance of your output by allowing you to select text fonts, colors, patterns, and line styles, and control the size and position of many graphics elements.

□ creates presentation graphics. SAS/GRAPH software can create text slides, display several graphs at one time, combine graphs and text in one display, and create automated presentations.

□ generates a variety of graphics output that you can display on your screen or in a Web browser, store in catalogs, review, or send to a hardcopy graphics output device such as a laser printer, plotter, or slide camera.

□ provides utility procedures and statements to manage the output.

This chapter describes the graphs that are produced by SAS/GRAPH software and explains some of the parts and features of SAS/GRAPH programs.

Generating Graphs

SAS/GRAPH software produces many kinds of charts, plots, and maps in both two- and three-dimensional versions. In addition to helping you understand the variety of graphs that are available to you, these descriptions will also help you choose the correct type of graph for your data and point you to the appropriate chapter.

Charts

SAS/GRAPH software uses the GCHART procedure to produce charts that graphically represent the value of a statistic for one or more variables in a SAS data set. See Chapter 13, "The GCHART Procedure," on page 519 for a complete description.
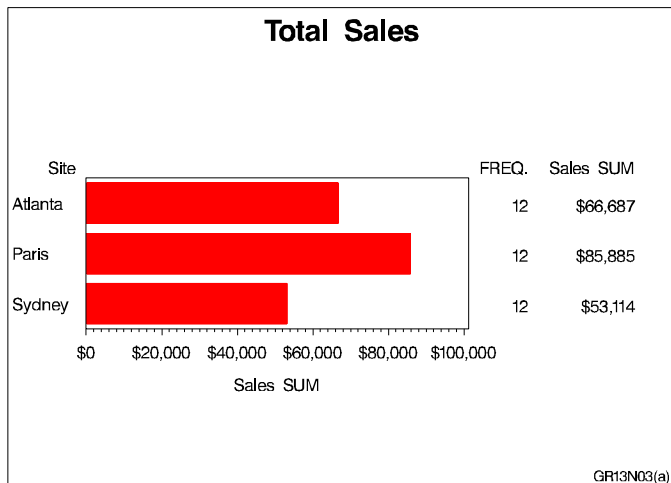
Block charts

Block charts use three-dimensional blocks to graphically represent values of statistics. Block charts are useful for emphasizing relative magnitudes and differences among data values.

Total Sales

BLOCK CHART OF SUM

Atlanta	Paris	Sydney
$66,687	$85,885	$53,114

Site

GR13N01
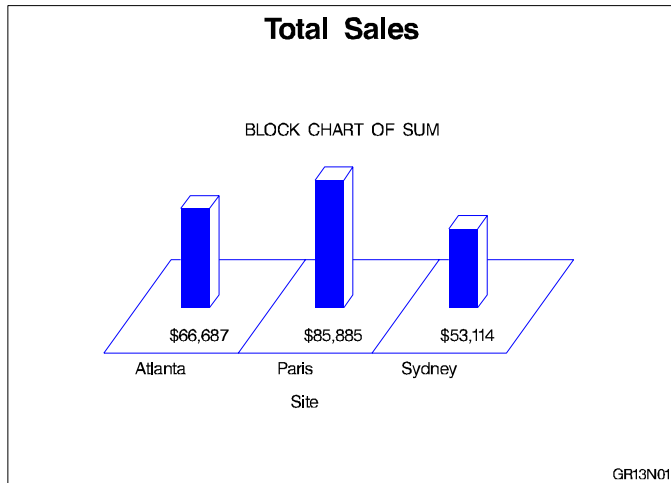
Horizontal bar charts

Horizontal bar charts use horizontal bars to represent statistics based on the values of one or more variables. Horizontal bar charts can generate a table of chart statistics and are useful for displaying exact magnitudes and emphasizing differences.

Total Sales

Site		FREQ.	Sales SUM
Atlanta		12	$66,687
Paris		12	$85,885
Sydney		12	$53,114

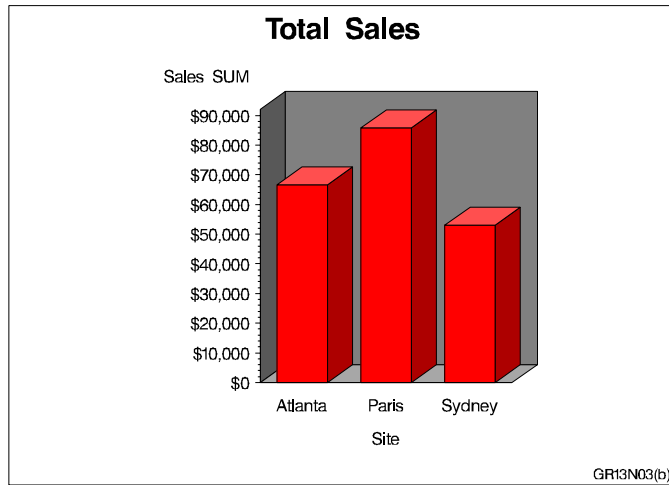$0 $20,000 $40,000 $60,000 $80,000 $100,000

Sales SUM

GR13N03(a)

Vertical bar charts

Vertical bar charts use vertical bars to represent statistics based on the values of one or more variables. Vertical bar charts, which generate only one statistic, are useful for displaying exact magnitudes and emphasizing differences.

Pie charts, 3-D Pie charts, and Donut charts

Pie charts, 3-D Pie charts, and Donut charts use the angle of pie slices to graphically represent the value of a statistic for a data range. Pie charts are useful for examining how the values of a variable contribute to the whole and for comparing the values of several variables.
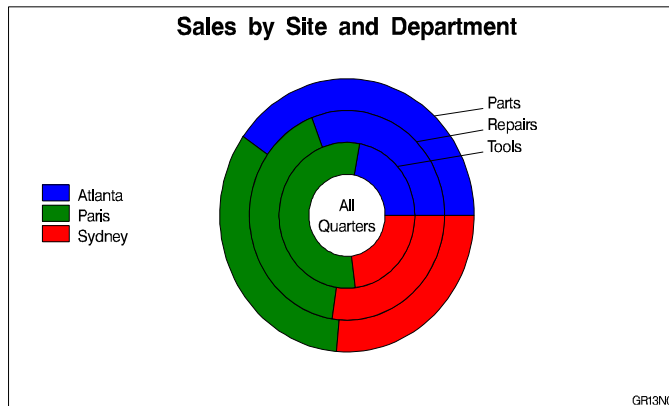
Star charts

Star charts use the length of spines to graphically represent the value of a statistic for a data range. Star charts are useful for analyzing where data are out of balance.

Rejected Parts

Apr
70

May   Mar
64   61

Jun   Feb
63   66

Jul   Jan
72   53

Aug   Dec
59   60

Sep   Nov
48   59

Oct
37

GR13N14(a)
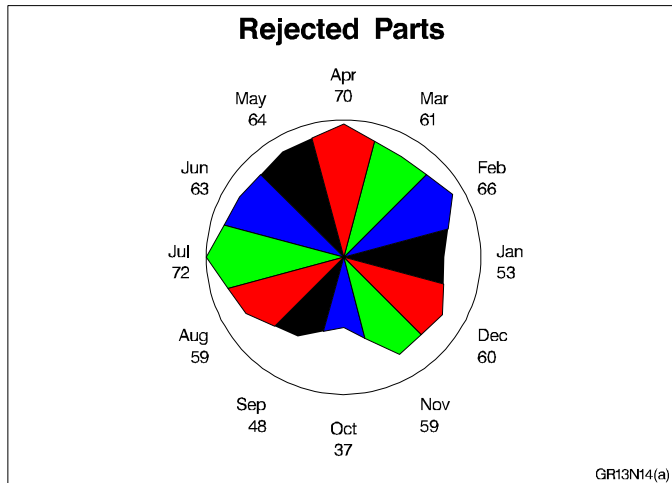
Two-Dimensional Plots

SAS/GRAPH software uses the GPLOT procedure to produce two-dimensional graphs that plot one or more dependent variables against an independent variable within a set of coordinate axes. GPLOT can display the data points as individual symbols (as in a scatter plot), or use interpolation methods specified by the SYMBOL statement to join the points, request spline interpolation or regression analysis, produce various high-low plots, or generate several other types of plots.
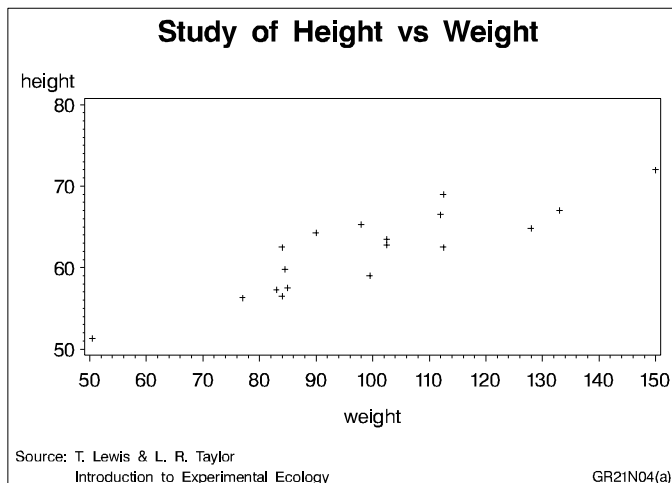
GPLOT can also display data as bubble plots in which circles of different sizes represent the values of a third variable.

Plots are useful for demonstrating the relationship between two or more variables and frequently compare trends or data values or depict movements of data values over time.

See Chapter 21, "The GPLOT Procedure," on page 801 for a complete description.

Two-dimensional scatter plots

Two-dimensional scatter plots show the relationship of one variable to another, often revealing concentrations or trends in the data. Typically, each variable value on the horizontal axis can have any number of corresponding values on the vertical axis.

Study of Height vs Weight

height
80

70

60

50

50 60 70 80 90 100 110 120 130 140 150

weight

Source: T. Lewis & L. R. Taylor
Introduction to Experimental Ecology   GR21N04(a)

Simple line plots

Simple line plots show the relationship of one variable to another, often as movements or trends in the data over a period of time. Typically, each variable value on the horizontal axis has only one corresponding value on the vertical axis. The line connecting data points can be smoothed using a variety of interpolation methods, including the Lagrange and the cubic spline interpolation methods.
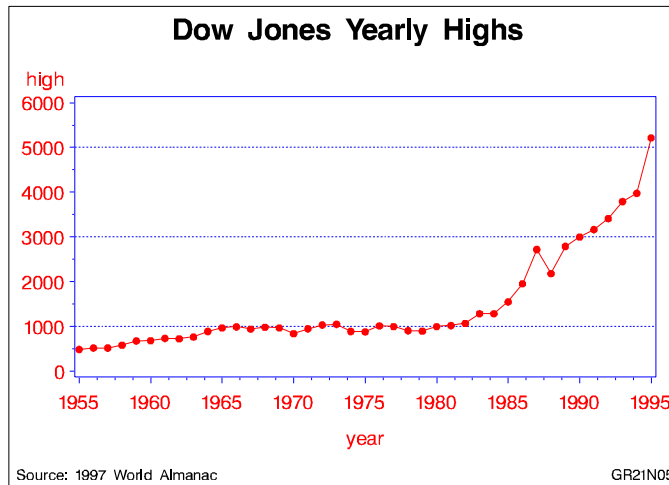
Regression plots

Regression plots specify that the plot is a regression analysis. You can specify one of three types of regression equation – linear, quadratic, or cubic – and optionally display confidence limits for mean predicted values or individual predicted values.
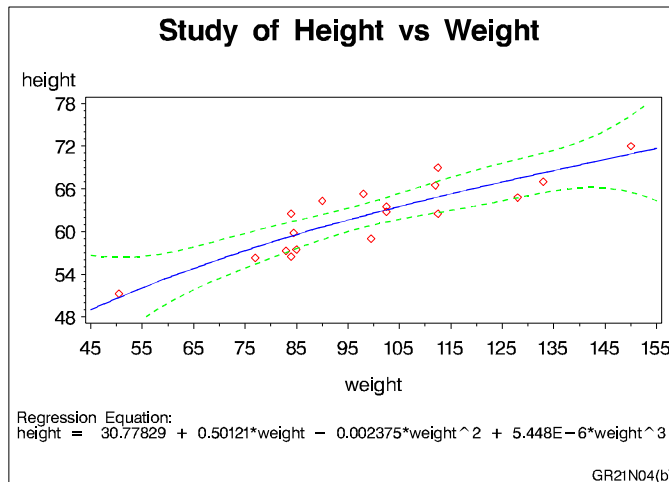
High-low plots

High-low plots show how several values of one variable relate to one value of another variable. Typically, each variable value on the horizontal axis has several corresponding values on the vertical axis. High-low plots include box, needle, and stock market plots.

Comparison of Grades by Section

(figure: box plot showing grade comparison by section for Monday, Wednesday, and Friday)

GR08N12(b)
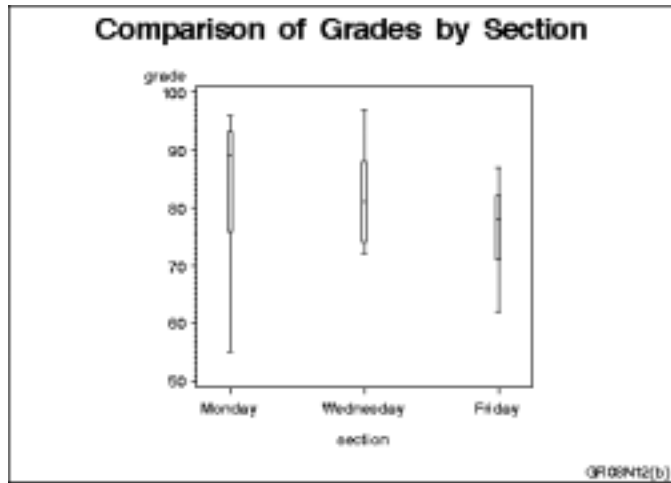
Bubble plots

Bubble plots show the relative magnitude of one variable in relation to two other variables. The values of two variables determine the position of the bubble on the plot, and the value of a third variable determines the size of the bubble.
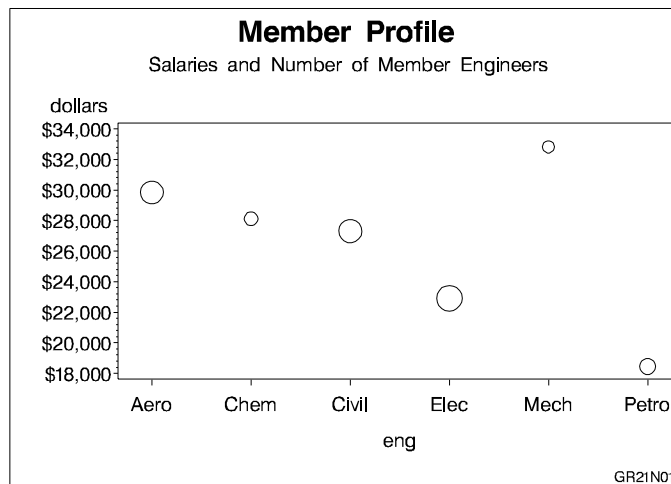
Member Profile
Salaries and Number of Member Engineers

(figure: bubble plot of dollars vs eng for Aero, Chem, Civil, Elec, Mech, Petro)

GR21N01

Three-Dimensional Plots

SAS/GRAPH software uses the G3D procedure to produce three-dimensional surface and scatter plots that examine the relationship among three variables. Variable values are plotted on a set of three coordinate axes.

See Chapter 29, "The G3D Procedure," on page 975 for a complete description.

Surface plots

Surface plots are three-dimensional plots that display the relationship of three variables as a continuous surface. Surface plots examine the three-dimensional shape of data.

Surface Plot of HAT Data Set

GR29N01
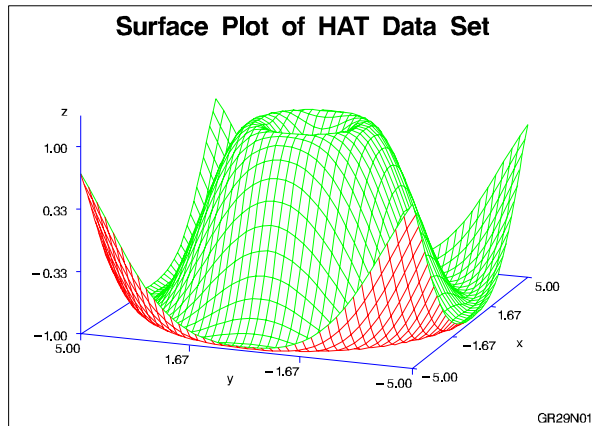
Scatter plots

Scatter plots enable you to examine three-dimensional data points instead of surfaces and to classify your data using size, color, shape, or a combination of these features.
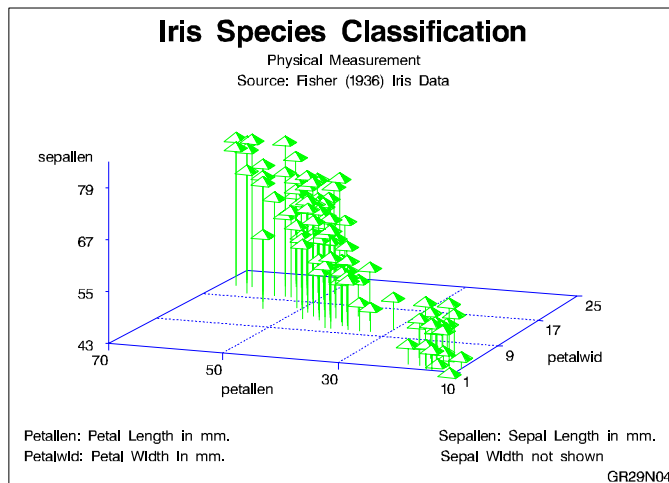
Iris Species Classification

Physical Measurement
Source: Fisher (1936) Iris Data

Petallen: Petal Length in mm.
Petalwld: Petal Width In mm.

Sepallen: Sepal Length in mm.
Sepal Width not shown

GR29N04

Contour plots

SAS/GRAPH software uses the GCONTOUR procedure to examine three-dimensional data in two dimensions. Lines or areas in a contour plot represent levels of magnitude (z) corresponding to a position on a plane (x,y).

See Chapter 14, "The GCONTOUR Procedure," on page 625 for a complete description.

Contour plots are two-dimensional plots that show three-dimensional relationships. These plots use contour lines or patterns to represent levels of magnitude of a contour variable plotted on the horizontal and vertical axes.

Clay Content at Site A
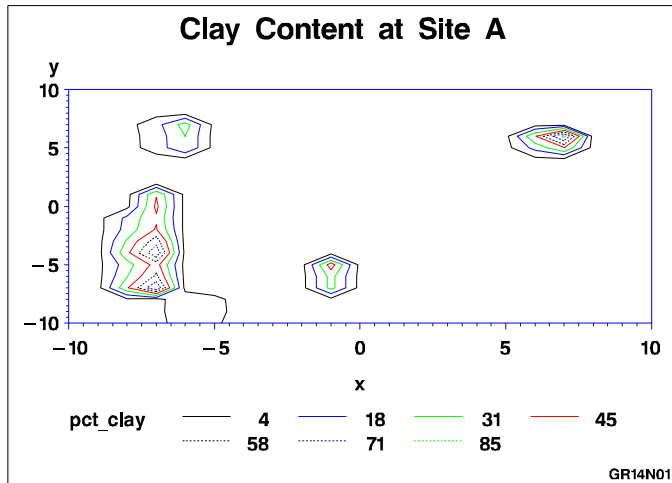
When you need to interpolate or smooth data values that are used by the G3D and GCONTOUR procedures, use the G3GRID procedure. The G3GRID procedure does not produce graphics output but processes existing data sets to create data sets that the G3D or GCONTOUR procedure can use to produce three-dimensional surface or contour plots. See Chapter 30, "The G3GRID Procedure," on page 1007 for a complete description.

Maps

SAS/GRAPH software uses the GMAP procedure to produce two- and three-dimensional maps that can show an area or represent values of response variables for subareas.

SAS/GRAPH software includes data sets to produce geographic maps. In addition, you can create your own map data sets.

See Chapter 19, "The GMAP Procedure," on page 731 for a complete description.

Block maps

Block maps are three-dimensional maps that represent data values as blocks of varying height rising from the middle of the map areas.

Hazardous Waste Site Installations (1997)
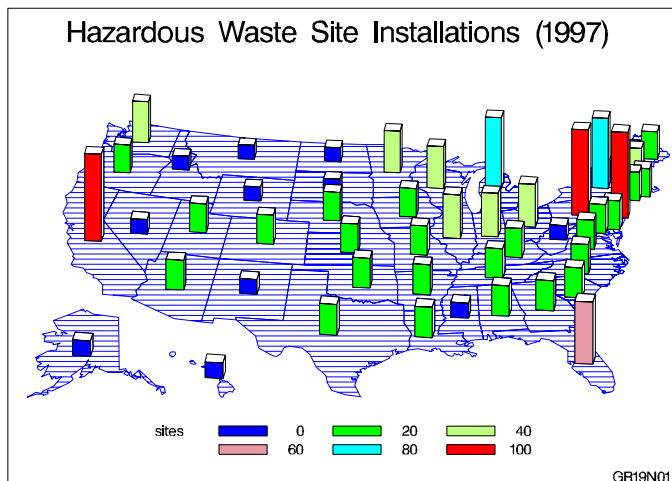
Choropleth maps

Choropleth maps are two-dimensional maps that display data values by filling map areas with combinations of patterns and color that represent the data values.
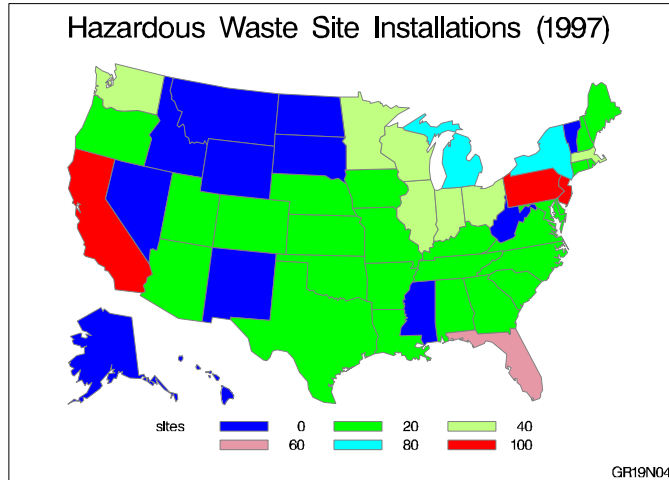
Prism maps

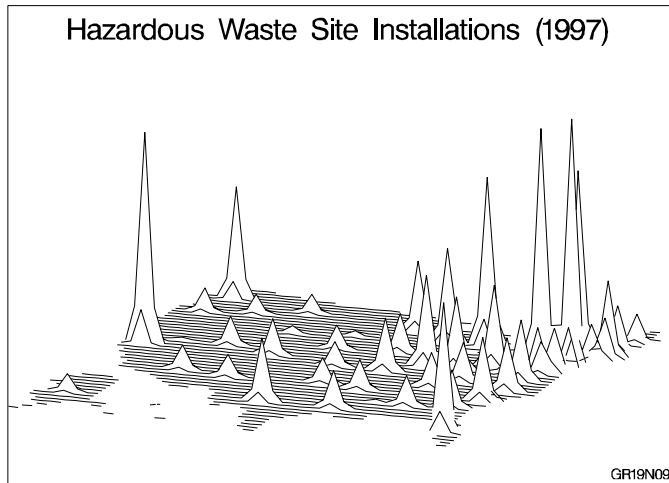Prism maps are three-dimensional maps that display data by raising the map areas and filling them with combinations of patterns and colors.
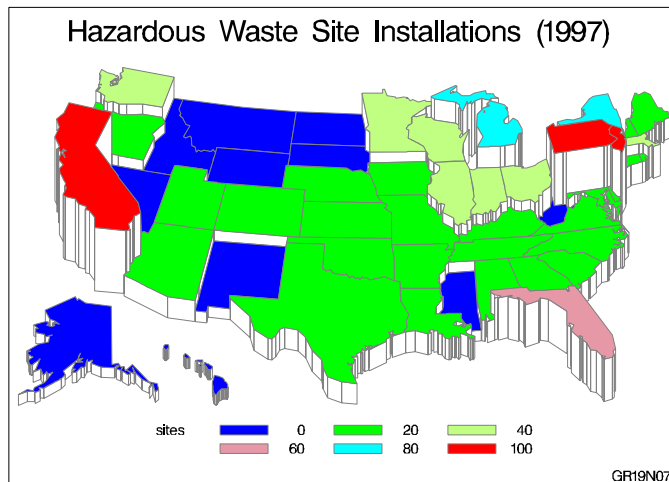
Surface maps

Surface maps are three-dimensional maps that represent data values as spikes of varying heights.

Hazardous Waste Site Installations (1997)

GR19N09

SAS/GRAPH software also provides several utility procedures for handling map data. The GPROJECT procedure lets you choose how geographic maps are projected. This is particularly important for large areas because producing a map of any large area on the Earth involves distorting some areas in the process of projecting the spherical surface of the Earth onto a flat plane. You can use the procedure to select the projection method that least distorts your map.

Map areas are constructed of joined data points. Each data point represents an observation in a SAS data set. For large maps, the amount of data can be prohibitively expensive (in terms of computing resources or time to process); the GREDUCE procedure enables you to reduce the number of points in the data set. The GREMOVE procedure enables you to remove boundary lines within a map.

Creating Text Slide and Presentation Graphics

You can use SAS/GRAPH software to create slide presentations of your graphs. With SAS/GRAPH you can

- □ create text slides with the GSLIDE and GPRINT procedures
- □ combine several graphs into one output with the GREPLAY procedure
- □ automatically or manually replay your graphs and text slides with the GREPLAY procedure.

Text Slides

Use the GSLIDE procedure to create text slides in which you can specify a variety of colors, fonts, sizes, angles, overlays, and other modifications as well as drawing lines and boxes on the output.

See Chapter 27, "The GSLIDE Procedure," on page 959 for a complete description.

Text slides display text as graphics output. Text slides can be used as title slides for presentations, or to produce certificates, signs, or other display text.
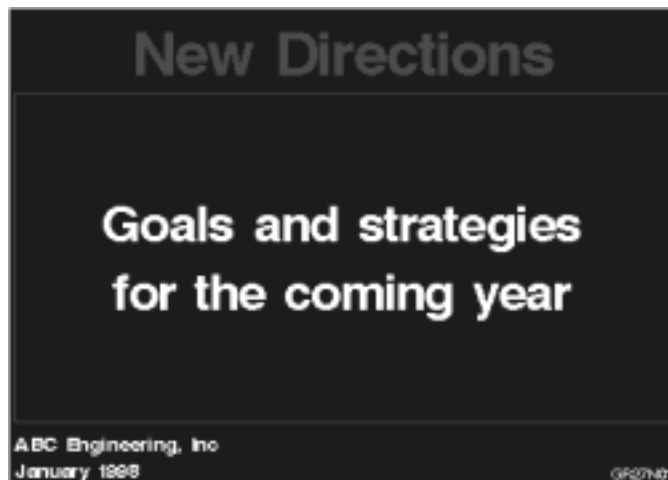
Use the GPRINT procedure to display as a graphic SAS procedure output that has been saved in a text file. With GPRINT, you bring the text file into SAS/GRAPH and then add titles, notes, and footnotes, and select colors for the output.

See Chapter 22, "The GPRINT Procedure," on page 859 for a complete description.

Combining Output into One Slide

Use the GREPLAY procedure to combine several graphs into a single output. You can create special effects by overlaying or rotating the graphs at any angle.

Templated graphs display two or more graphs or text slides as one output by replaying stored graphs into a template or framework. Like graphs and text slides, templated graphs can be ordered in groups and stored in catalogs for replay as part of a presentation.

Figure 1.1   Templated graphs
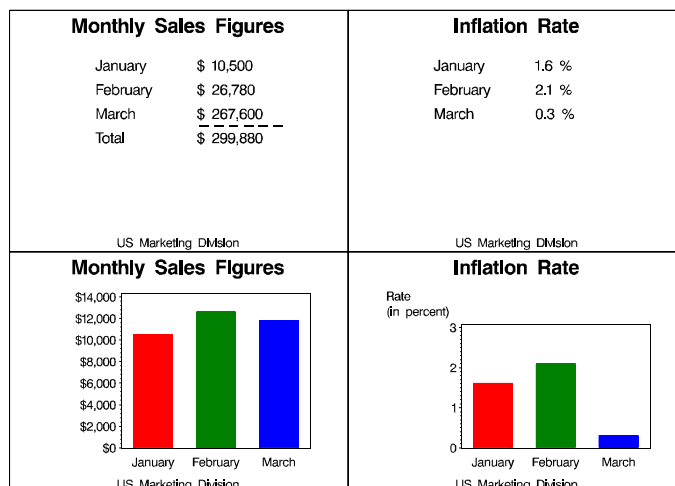
In addition, you can use the GREPLAY procedure to create an automated or user-controlled presentation of graphics output. The GREPLAY procedure enables you to name, arrange, and customize the presentation of graphs that are stored in a catalog.

See Chapter 26, "The GREPLAY Procedure," on page 919 for a complete description.

Enhancing Graphics Output (graphs and text slides)

SAS/GRAPH Statements

You also can use *global statements* and *graphics options* in SAS/GRAPH programs. With global statements, you can add titles and footnotes and control the appearance of axes, symbols, patterns, and legends. With graphics options, you can control the appearance of graphics elements by specifying default colors, fill patterns, fonts, text height, and so on.

The Annotate Facility

The Annotate facility enables you to program graphics by using certain variables in SAS data sets. It is often used to add text or special elements to the graphics output of other procedures, although it also can be used to construct custom graphics output. Text and graphics can be placed at coordinates derived from input data, as well as coordinates expressed as explicit locations on the display.

Figure 1.2   Annotated graphs
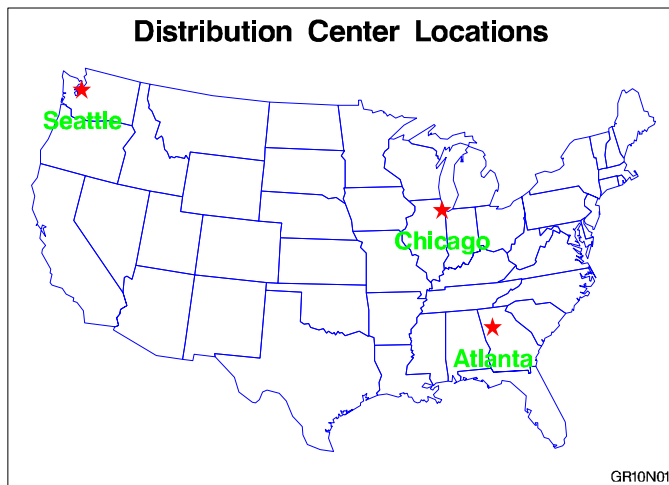
Creating Custom Graphics

The Annotate facility can also be used to generate custom graphics without using any of the SAS/GRAPH graphing procedures.

The DATA Step Graphics Interface

The DATA Step Graphics Interface provides functions and calls that produce graphics output from the DATA step, rather than from a procedure. The functions and calls are similar in form to those specified by the ISO Graphic Kernal Standard (GKS); however, the interface is not an implementation of the GKS. The form is similar enough

that many GKS-compliant programs may be converted easily to run as SAS/GRAPH programs.

Graph-N-Go

To generate presentation graphs without writing any SAS/GRAPH code, you can use Graph-N-Go (not available on mainframes). You can start Graph-N-Go in several ways:

☐ from the menus in any SAS window, select

| Solutions | ► | Reporting | ► | Graph-N-Go |

☐ submit either of the following from the SAS command line:

```
gng

graphngo
```

☐ use an Explorer window to directly open a GFORM entry. Double-click (or right-click and choose Open) on a GFORM entry to start a Graph-N-Go session using that entry.

Information on using the application is in Graph–N-Go help, which you can access from the application's main window in either of two ways:

☐ select

| Help | ► | Using This Window |

☐ press F1 (this may not work in some operating environments).

You can also get help for the application by submitting the following command from the SAS command line:

```
help gng
```

About this Book

This book provides reference information for all facilities, procedures, statements, and options that can be used with SAS/GRAPH software. This chapter describes what you need to know to use SAS/GRAPH software, and what conventions are used in text and example code. To gain full benefit from using this book, you should familiarize yourself with the information presented in this chapter, and refer to it as needed.

Audience

This book is written for users who are experienced in using SAS System software. You should understand the concepts of programming in the SAS language, and you should have an idea of the tasks you want to perform with SAS/GRAPH software.

Prerequisites

The following table summarizes the SAS System concepts that you need to understand in order to use SAS/GRAPH software:

To learn how to	Refer to
invoke the SAS System at your site	instructions provided by the SAS Software Consultant at your site
use base SAS software	*SAS Language Reference: Concepts* or *SAS Language Reference: Dictionary*
use the DATA step to create and manipulate SAS data sets	
use the SAS Text Editor to enter and edit text	
allocate SAS data libraries and assign librefs	documentation for using the SAS System under the operating system for the hardware at your site
create external files and assign filerefs	
manipulate SAS data sets using SAS procedures	*SAS Procedures Guide*

Conventions Used in This Book

This section explains the conventions this book uses for text, SAS language syntax, and file and library references. The book uses the following terms in discussing syntax:

keyword   is a literal that is a primary part of the SAS language. (A literal must be spelled exactly as shown, although it can be entered in uppercase or lowercase letters.) Keywords in this book are procedure names, statement names, macro names, routine names, and function names.

argument   is an element that follows a keyword. It is either literal, or it is user-supplied. It has a built-in value (for example, NODISPLAY), or it has a value assigned to it (for example, COLOR=*text-color*).

                      Arguments that you must use are *required arguments*. Other arguments are *optional arguments*, or simply *options*.

value   is an element that follows an equal sign. It assigns a value to an argument. It may be a literal, or it may be a user-supplied value.

parameter   is a value assigned to an argument that itself takes a value, for example, the COLOR= parameter of the LABEL= option in a LEGEND statement, as shown in the following statement:

```
legend label=(color=blue);
```

Syntax Conventions

Type styles have special meanings when used in the presentation of SAS/GRAPH software syntax in this book. The following list explains the style conventions for the syntax sections:

UPPERCASE   identifies SAS keywords such as the names of statements and procedures (for example, PROC GCHART). Also identifies arguments and values that are literals, (for example, NOLEGEND and LABEL=NONE).

italic   identifies arguments or values that you supply. Items in italic can represent user-supplied values that are either

            □ nonliteral values assigned to an argument (for example, *axis-color* in COLOR=*axis-color*)

□ nonliteral arguments (for example, VBAR *chart-variable*. . . ;).

In addition, an item in italics can be the generic name for a list of arguments or parameters from which the user can choose (for example, *appearance-options*).

The following symbols are used to indicate other syntax conventions:

< > (angle brackets)   identify optional arguments. Any argument not enclosed in angle brackets is required.

| (vertical bar)   indicates that you can choose one value from a group. Values separated by bars are mutually exclusive.

. . . (ellipsis)   indicates that the argument following the ellipsis can be repeated any number of times (*plot-request <. . . plot-request-n>*, for example). If the ellipsis and the following argument are enclosed in angle brackets, they are optional. In SAS/GRAPH software, an ellipsis also indicates a range from which a value is selected (LINE=1 . . . 46, for example).

The following examples illustrate the syntax conventions described in this section. These examples contain selected syntax elements, not complete syntax.

PROC GANNO ANNOTATE=*Annotate-data-set*
    <DATASYS>;

□ PROC GANNO is in uppercase because it is a SAS keyword, the name of a statement. The remaining elements are arguments for the statement.
□ ANNOTATE= is not enclosed in angle brackets because it is a required argument. It is in uppercase to indicate that it is a literal and must be spelled as shown.
□ *Annotate-data-set* is in italic because it is a value that you must supply; in this case, the value must be a data set name.
□ DATASYS is enclosed in angle brackets because it is an optional argument. It is in uppercase to indicate that it is a literal and must be spelled as shown.
□ The ending semicolon (;) is required because it is outside the angle brackets for the option.

SYMBOL <1 . . . 99>
    <COLOR=*symbol-color*>
    <MODE=EXCLUDE|INCLUDE>
    <*appearance-options*>;

□ SYMBOL is in uppercase because it is a SAS keyword, the name of a statement. The numbers 1 . . . 99 are in angle brackets because they are optional. The ellipsis indicates that you choose one from the range of numbers 1 through 99. The remaining elements are arguments for the statement.
□ COLOR= is enclosed in angle brackets because it is an optional argument.
□ *Symbol-color* is in italics because it represents a value that you specify.
□ MODE= is enclosed in angle brackets because it is an optional argument.
□ EXCLUDE and INCLUDE are in uppercase because they are literal values and must be spelled exactly as shown. They are separated by a vertical bar (an *or* bar) because you use one or the other but not both.

□ *Appearance-options* is in italics because it is a generic name for a list of options that can be used in the SYMBOL statement.

HBAR *chart-variable*< . . . *chart-variable-n*>
    </ <PATTERNID=BY | GROUP | MIDPOINT | SUBGROUP>
    <*statistic-options*>>;

□ *Chart-variable* is italic because it is an argument that you supply. It is required because it is not in angle brackets.

□ *Chart-variable-n* is enclosed in angle brackets because additional user-supplied arguments are optional. The ellipsis before the argument indicates that it can be repeated as many times as desired.

□ PATTERNID= is a literal option. The values BY, GROUP, MIDPOINT, and SUBGROUP are literal values that are mutually exclusive. You can use only one, and it must be spelled as shown.

□ *Statistic-options* is in italics because it is the generic name of a list of options that affect the chart statistics.

When you are using an option, a statement, or a procedure whose syntax shows arguments or values in italics, you must supply the argument or value. When the argument or value is a font, color, or variable name, SAS/GRAPH software expects valid font names, color names, and variable names. Consider the following four syntax samples:

FONT=*font*

COLOR=*color*

COLOR=*text-color*

PIE *chart-variable* < . . . *chart-variable-n*>;

□ *Font* must be a valid SAS font name. (See Chapter 6, "SAS/GRAPH Fonts," on page 125 for details.)

□ *Color* and *text-color* must be valid SAS/GRAPH colors. (See Chapter 7, "SAS/GRAPH Colors," on page 139 for details.)

□ *Chart-variable* must be a valid SAS variable name. (See *SAS Language Reference: Dictionary* for details.)

Conventions for Examples and Output

Most of the chapters in this book include examples that illustrate some of the features of a procedure or its statements. Each example contains

□ a description of the highlights of the example

□ the program statements that produce the output

□ the actual output from the example

   □ an explanation of the features of the example.

The output that is shown for the examples in this book was generated in an HP-UX operating environment, using the default XCOLOR display device. If you are using a different operating environment or display device, you may need to make some minor adjustments to the example programs.

The dimensions of the graphics output area vary across devices and when using the GRAPH windows. The dimensions may affect aspects of the graphics output – for example, the appearance of axes or the position of graphics elements that use explicit coordinates in units other than percent. You may need to adjust the dimensions of your graphics output area or the size of graphics elements to correct any differences you see. Most of the examples in this book used a GOPTIONS statement to set the graphics output area to 7 inches by 5 inches, which proved to be a good dimension for generating output displays for this book:

```
goptions hsize=7in vsize=5in;
```

These HSIZE= and VSIZE= settings are not shown in the example code and are not necessary for generating the output, but you may want to use similar settings if your output looks different from the output that is shown in the book.

The examples use explicit color specifications, and the example code contains the names of colors that approximate the colors you see in this book. The colors displayed on your device may differ from those shown in the examples because of differences in device capabilities.

The examples in this book do not specify actual names for the file structures used for SAS data libraries or for external files. This is because different operating environments and different computing installations use different conventions for naming files and directories. Instead, the examples refer to storage locations generically. For example, a LIBNAME statement to assign the libref STORE is shown as

```
libname store 'SAS-data-library';
```

For *SAS-data-library*, you should supply the complete specification for the storage location (for example, directory, MVS data set, or CMS minidisk) of the data library, using the form required by your operating environment.

Similarly, a FILENAME statement to assign the fileref MYFILE is shown as

```
filename myfile 'external-file';
```

For *external-file*, you should supply a fully qualified filename, using the form required for your operating environment.

If you are unsure of the requirements at your site, see your SAS Software Consultant for more information.

Some examples explicitly specify the font and height for text, the units, and border in procedure statements. For those examples that do not include explicit specifications, the following graphics options were used to produce uniform output:

RESET=GLOBAL   cancels all currently defined AXIS, LEGEND, FOOTNOTE, TITLE, PATTERN, and SYMBOL definitions.

GUNIT=PCT   specifies the unit specification for options is in percent, unless explicitly specified in another SAS statement.

FTEXT=SWISSB   specifies that all text uses the SWISSB font, unless explicitly specified in another SAS statement.

HTITLE=6   specifies that the first title line is 6 percent of the height of the graphics output area, unless explicitly specified in another SAS statement.

HTEXT=3   specifies that text is 3 percent of the height of the graphics output area, unless explicitly specified.

BORDER   causes a border to be drawn around the graphics output area. The border, which appears in most output, represents the boundaries of the graphics output area, whether or not a border is drawn by the program.

Note: The way that output is presented on your device depends on the environment in which you are running SAS/GRAPH software. △

Information You Should Know

This section outlines information you should know before you attempt to run the examples in this book.

Support Personnel

Most sites have personnel available to help users learn to run SAS System software. Record the name of the SAS Software Consultant, SAS Software Representative, and system administrator at your site. Also record the names of anyone else you regularly turn to for help with running SAS/GRAPH software.

Sample Programs

Most of the chapters in this book provide examples that demonstrate some of the major features of SAS/GRAPH software. To minimize the typing you must do to run the sample code yourself, the code is delivered to you through the SAS Sample Library. Depending on your operating environment, there are up to three ways that you can access the code that is in the sample library:

□ if you are viewing the sample code in SAS OnlineDoc, you can copy the code out of the OnlineDoc and paste it into the Program Editor in your SAS session. This alternative is not available if you do not have access to a Web browser in the operating environment where you are running your SAS session.

□ in most operating environments (excluding mainframe environments), you can access the sample code through the SAS Help facility. For example, from a SAS window's Help menu, you can choose SAS System Help to enter the help system. You can then choose the link for Sample SAS Programs and Applications, which takes you to the help page for the SAS Sample Library.

□ in most operating environments (excluding Windows), the SAS Sample Library may have been installed in your file system. If the SAS Sample Library has been installed at your site, ask your SAS Software Consultant where it is located.

To access the sample programs through SAS System Help or through your file system, you must understand the naming convention used for the samples. The naming convention for SAS/GRAPH samples is GR*cc*N*nn*, where *cc* is the chapter number and *nn* is the example number. For example, the code to create the first example output in Chapter 12 is stored in sample member GR12N01. The sample-library member name is usually displayed as a footnote in the output's lower-right corner.

□ In SAS System Help, the sample programs are organized by product. Within each product category, the samples are sorted alphabetically by title. Thus, to access

the code for the first example in Chapter 12, navigate in the help system to the SAS Sample Library page, choose SAS/GRAPH from the list of products, and then scroll to the title "Reference Guide Chapter 12 Output 1."

□ In your file system, the files that contain the sample code have file names that match the sample member names. For example, in a directory-based system, the code for sample member GR12N01 is located in a file named GR12N01.SAS.

Note: Some of the examples include LIBNAME and FILENAME statements. You must provide the name of the SAS data library or external file before running the example. △

Map Data Sets

To run the examples that draw maps, you need to know where the map data sets are stored on your system. Depending on your installation, the map data set may automatically be assigned a libref. Ask your SAS Software Consultant or system administrator where the map data sets are stored for your site.

Annotate Macros Data Set

To run the examples using Annotate macros, you need to know where the Annotate macro data set is stored on your system. Depending on your installation, the Annotate macro data set may automatically be assigned a fileref. Ask your SAS Software Consultant or system administrator where the Annotate macro data set is stored for your site.