NCAR Graphics
AUTOGRAPH
A Graphing Utility
Version 2.00  August 1987

Dave Kennison

Scientific Computing Division
National Center for Atmospheric Research
Boulder, Colorado
The Technical Note series provides an outlet for a variety of NCAR manuscripts that contribute in specialized ways to the body of scientific knowledge but which are not suitable for journal, monograph, or book publication. Reports in this series are issued by the NCAR Scientific Divisions. Designation symbols for the series include:

- **EDD - Engineering, Design, or Development Reports**
  Equipment descriptions, test results, instrumentation, and operating and maintenance manuals.

- **IA - Instructional Aids**
  Instruction manuals, bibliographies, film supplements, and other research or instructional aids.

- **PPR - Program Progress Reports**
  Field program reports, interim and working reports, survey reports, and plans for experiments.

- **PROC - Proceedings**
  Documentation of symposia, colloquia, conferences, workshops, and lectures. (Distribution may be limited to attendees.)

- **STR - Scientific and Technical Reports**
  Data compilations, theoretical and numerical investigations, and experimental results.

Copyright © 1987, University Corporation for Atmospheric Research. All Rights Reserved.

Published by: National Center for Atmospheric Research
Scientific Computing Division
P.O. Box 3000
Boulder, CO 80307

Additional copies of this manual are available from Documentation Distribution at the above address. Please contact Document Distribution for price information.

The National Center for Atmospheric Research is operated by the University Corporation for Atmospheric Research and is sponsored by the National Science Foundation. Any opinion, findings, conclusions, or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.
CONTENTS

SECTION I: INTRODUCTION AND OVERVIEW .................... 1

INTRODUCTION ............................................ 3

OVERVIEW ................................................ 4

How to Draw a Graph.................................... 4
A Simpler Way to Draw a Graph....................... 4
The Autograph Control Parameters................... 5
Control Parameter Names............................... 5
Control Parameters Are Floating-Point.............. 6
Use of Parameter-Access Routines.................... 7
Side Effects of Parameter-Setting................... 7
Saving and Restoring Defaults........................ 7
Special Values 'NULL/1.' and 'NULL/2.'.............. 8
The Graph Window..................................... 8
The Grid Window....................................... 9
The Grid Coordinate System......................... 10
The User Coordinate System.......................... 10
How to Graph "X as a Function of Y"................. 11
What a Background Consists Of...................... 11
The Four Axes......................................... 12
Abbreviated Form of Axis-Parameter Names......... 12
The Parameter 'AXIS/s/CONTROL.'.................... 13
The Parameter 'AXIS/S/s/Line.'....................... 13
Moving an Axis....................................... 13
The "Label Coordinate System" Along an Axis...... 14
Positioning of Major Tick Marks on an Axis....... 15
Appearance of Major Tick Marks..................... 15
Positioning/Appearance of Minor Tick Marks........ 16
Numeric Labels on an Axis........................... 17
Types of Numeric Labels............................ 17
Orientation of Numeric Labels...................... 18
Positioning of Numeric Labels...................... 18
Character Sizes in Numeric Labels.................. 19
Informational Labels................................. 19
The Predefined Labels............................... 21
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGCHIL(IFLG,ILBNM,INNO)</td>
<td>67</td>
</tr>
<tr>
<td>AGCHNL(IAXS,VILS,CHR,M,MCIM,NCIM,...)</td>
<td>67</td>
</tr>
<tr>
<td>AGPWRT(XPOS,YPOS,CHR,NCHS,ISIZ,IORI,ICEN)</td>
<td>69</td>
</tr>
<tr>
<td>SECTION III: PARAMETERS</td>
<td>71</td>
</tr>
<tr>
<td>'PRIMARY.'</td>
<td>74</td>
</tr>
<tr>
<td>'FRAME.'</td>
<td>75</td>
</tr>
<tr>
<td>'SET.'</td>
<td>75</td>
</tr>
<tr>
<td>'ROW.'</td>
<td>76</td>
</tr>
<tr>
<td>'INVERT.'</td>
<td>77</td>
</tr>
<tr>
<td>'WINDOW.'</td>
<td>78</td>
</tr>
<tr>
<td>'BACKGROUND.'</td>
<td>78</td>
</tr>
<tr>
<td>'NULL.'</td>
<td>80</td>
</tr>
<tr>
<td>'NULL/1.'</td>
<td>80</td>
</tr>
<tr>
<td>'NULL/2.'</td>
<td>81</td>
</tr>
<tr>
<td>'GRAPH.'</td>
<td>81</td>
</tr>
<tr>
<td>'GRAPH/LEFT.'</td>
<td>81</td>
</tr>
<tr>
<td>'GRAPH/RIGHT.'</td>
<td>82</td>
</tr>
<tr>
<td>'GRAPH/BOTTOM.'</td>
<td>82</td>
</tr>
<tr>
<td>'GRID.'</td>
<td>83</td>
</tr>
<tr>
<td>'GRID/LEFT.'</td>
<td>83</td>
</tr>
<tr>
<td>'GRID/RIGHT.'</td>
<td>83</td>
</tr>
<tr>
<td>'GRID/BOTTOM.'</td>
<td>84</td>
</tr>
<tr>
<td>'GRID/TOP.'</td>
<td>84</td>
</tr>
<tr>
<td>'GRID/SHAPE.'</td>
<td>84</td>
</tr>
<tr>
<td>'X.'</td>
<td>86</td>
</tr>
<tr>
<td>'X/MINIMUM.'</td>
<td>86</td>
</tr>
<tr>
<td>'X/MAXIMUM.'</td>
<td>87</td>
</tr>
<tr>
<td>'X/LOGARITHMIC.'</td>
<td>87</td>
</tr>
<tr>
<td>'X/ORDER.'</td>
<td>88</td>
</tr>
<tr>
<td>'X/NICE.'</td>
<td>88</td>
</tr>
<tr>
<td>'X/SMALLEST.'</td>
<td>89</td>
</tr>
<tr>
<td>'X/LARGEST.'</td>
<td>89</td>
</tr>
<tr>
<td>'Y.'</td>
<td>89</td>
</tr>
<tr>
<td>'Y/MINIMUM.'</td>
<td>90</td>
</tr>
<tr>
<td>'Y/MAXIMUM.'</td>
<td>90</td>
</tr>
<tr>
<td>'Y/LOGARITHMIC.'</td>
<td>90</td>
</tr>
<tr>
<td>'Y/ORDER.'</td>
<td>91</td>
</tr>
<tr>
<td>'Y/NICE.'</td>
<td>91</td>
</tr>
</tbody>
</table>

August 1987
'DASH/CHARACTER.' ................................... 115
'DASH/DOLLAR-QUOTE. '................................ 115
'DASH/PATTERNS. ' .................................. 115
'DASH/PATTERNS/n.' ................................... 116
'LABEL.' .............................................. 116
'LABEL/CONTROL.' .................................... 117
'LABEL/BUFFER.' ........................................ 117
'LABEL/BUFFER/LENGTH. ' .............................. 118
'LABEL/BUFFER/CONTENTS.' ............................ 118
'LABEL/BUFFER/NAMES.' ................................ 120
'LABEL/NAME.' ........................................ 120
'LABEL/DEFINITION.' .................................. 121
'LABEL/[DEFINITION/] SUPPRESSION.' .................. 121
'LABEL/[DEFINITION/] BASEPOINT. ' ................... 122
'LABEL/[DEFINITION/] BASEPOINT/X. ' .................. 122
'LABEL/[DEFINITION/] BASEPOINT/Y. ' ................... 123
'LABEL/[DEFINITION/] OFFSET. ' ....................... 123
'LABEL/[DEFINITION/] OFFSET/X. ' ..................... 123
'LABEL/[DEFINITION/] OFFSET/Y. ' ..................... 124
'LABEL/[DEFINITION/] ANGLE. ' ........................ 124
'LABEL/[DEFINITION/] CENTERING. ' ................... 124
'LABEL/[DEFINITION/] LINES. ' ........................ 125
'LABEL/[DEFINITION/] INDEX. ' ........................ 125
'LINE. ' ............................................. 125
'LINE/MAXIMUM. ' ..................................... 126
'LINE/END. ' ......................................... 126
'LINE/BUFFER.' ....................................... 127
'LINE/BUFFER/LENGTH. ' ................................ 127
'LINE/BUFFER/CONTENTS.' ............................. 128
'LINE/NUMBER. ' ...................................... 129
'LINE/DEFINITION. ' ................................... 130
'LINE/[DEFINITION/] SUPPRESSION. ' .................. 130
'LINE/[DEFINITION/] CHARACTER. ' .................... 131
'LINE/[DEFINITION/] TEXT. ' .......................... 131
'LINE/[DEFINITION/] LENGTH. ' ......................... 132
'LINE/[DEFINITION/] INDEX. ' ........................ 132
'SECONDARY. ' ......................................... 132
'SECONDARY/GRAPH. ' .................................. 133
'SECONDARY/USER. ' .................................... 133
'SECONDARY/CURVE.' .................................... 134
Section I: Introduction and Overview
INTRODUCTION

This is the unabridged write-up of AUTOGRAPH - a graphics package enabling the user to draw graphs, each with a labelled background and each displaying one or more curves.

The manual you are reading replaces AUTOGRAPH: The Unabridged Write-up, (NCAR/TN-245+IA), January 1985. AUTOGRAPH: A Graphing Utility contains updated example codes that have been rewritten for execution in a GKS environment. For a shortened description of AUTOGRAPH, see the "Utilities" section of the NCAR Graphics User's Guide.

This write-up is divided into several major sections: The section "OVERVIEW" presents the philosophy of the package and should eventually be read by any serious user of AUTOGRAPH. The section "ROUTINES" describes the various routines in the package and how to call them. The section "PARAMETERS" describes the "control parameters" which govern the behavior of AUTOGRAPH. The section "EXAMPLES" contains a set of graphs produced by AUTOGRAPH and the programs which produced them. The section "MESSAGES" describes the messages that accompany the possible error exits.

Each major section is divided into titled paragraphs, which are further divided into untitled paragraphs. In general, phrases like "the preceding paragraph" or "the following paragraph" are to be interpreted to refer to a titled paragraph.

Readers who wish only to quickly draw a simple graph may want to skip to the descriptions of the routines EZY, EZXY, EZMY, and EZMXY, in the section "ROUTINES". Others may wish to look first at the examples.

IMPORTANT NOTE: If you have existing programs that use the FORTRAN 66 version of AUTOGRAPH (any version of AUTOGRAPH acquired before October 23, 1984), you must first convert them to use the FORTRAN 77 version of AUTOGRAPH. See Appendix A, "Conversion from FORTRAN 66 AUTOGRAPH to FORTRAN 77 AUTOGRAPH," for instructions.
AUTOGRAPH: A Graphing Utility

OVERVIEW

This section describes the philosophy of AUTOGRAPH.

HOW TO DRAW A GRAPH

To draw a graph, a user program executes, directly or indirectly, a series of calls to AUTOGRAPH routines, typically as follows:

1. The routines AGSETP, AGSETF, AGSETI, and/or AGSETC are called to reset "primary control parameters" having unsatisfactory values.

2. The routine AGSTUP is called to perform "set-up" tasks (thus the name). It computes appropriate values for the "secondary control parameters".

3. The routine AGBACK is called to draw a background.

4. The routine AGCURV is called one or more times (once per curve) to draw the desired curves.

5. The system-plot-package routine FRAME is called to advance to a new frame.

To draw the next graph, all five steps are repeated; step 1 may, of course, be abbreviated or omitted entirely.

A SIMPLER WAY TO DRAW A GRAPH

Each of the routines EZY, EZXY, EZMY, and EZMXY performs a sequence of calls like that described above. A user program may call EZY to graph a single curve defined by the points (I,Y(I)), for I from 1 to N, EZXY to graph a single curve defined by the points (X(I),Y(I)), for I from 1 to N, EZMY to graph the M curves defined by the points (I,Y(I,J)), for I from 1 to N and J from 1 to M, and EZMXY to graph the M curves defined by the points (X(I),Y(I,J)) or (X(I,J),Y(I,J)), for I from 1 to N and J from 1 to M.

See the descriptions of these routines, in the section "ROUTINES". See also examples 1 through 4, in the section "EXAMPLES".
THE AUTOGRAPH CONTROL PARAMETERS

The labeled common block AGCONP contains the AUTOGRAPH "control parameters", each of which controls some aspect of the package's behavior. There are two types of these: "primary control parameters" and "secondary control parameters".

Each primary control parameter has a default value and is subject to change by a user program to produce a desired effect.

Each secondary control parameter is computed by AUTOGRAPH itself and is not normally subject to direct change by a user program. The values computed for some of the secondary control parameters may be of interest.

Access to all of the control parameters is provided by the routines AGSETP, AGSETF, AGSETI, AGSETC, AGGETP, AGGETF, AGGETI, and AGGETC. (The routines ANOTAT and DISPLA provide access to a limited subset of the control parameters and are provided principally for historical reasons; they are of interest mainly to users of the routines EZY, EZXY, EZMY, and EZMXY.)

In the following discussion, the long phrase "primary control parameter" will usually be shortened to just "parameter".

CONTROL PARAMETER NAMES

There are many groups of parameters. Each group has a keyword associated with it - like BACKGROUND or GRAPH or AXIS. Those groups which contain more than one parameter are divided into subgroups, each of which also has a keyword associated with it. The subgroups may be further subdivided in the same manner.

Group keywords are used to make up names of parameter groups and, ultimately, of individual parameters. This is done by stringing together the group keywords, in descending order, separated by slashes and terminated by a period. For example, the name 'AXIS.' refers to a group of 92 parameters describing the four axes, the name 'AXIS/LEFT.' to a subgroup of 23 parameters describing the left axis, the name 'AXIS/LEFT/NUMERIC.' to a further subgroup of 8 parameters describing the numeric labels on the left axis, and the name 'AXIS/LEFT/NUMERIC/TYPEx.' to a single parameter describing the type of numeric labels on the left axis.
Parameter-group names are used as arguments in calls to parameter-access routines to identify the parameter(s) whose values a user program wishes to "set" or "get". For example, the statement

CALL AGSETP ('AXIS/LEFT/NUMERIC/TYPE.',1.)

is used to set the value of the parameter specifying the type of numeric labels to be used on the left axis to "1."

Parameter-group names may be shortened considerably, both by abbreviation of the keywords and by omission of some keywords; for example, the name shown above may be shortened to 'LE/TY.'.

Complete information about the control parameters and their names is given below, in the section "PARAMETERS".

CONTROL PARAMETERS ARE FLOATING-POINT

All of the control parameters are floating-point - even those which serve as type specifiers, control flags, item counts, list pointers, and the like - for which integer variables would normally be used. This was done because of a portability problem which arose in implementing the parameter-access routines.

Those parameters which may only have discrete integral values are referenced internally using the FORTRAN function IFIX. For example: The parameter 'X/NICE.' corresponds to a variable in the common block AGCONP named QCXE, which may have any of the values "-1.", "0.", or "+1.". The function IFIX(QCXE) is used by AUTOGRAPH to recover the integer value.

Conceptually, some parameters have character-string values; for example, the parameter 'DASH/PATTERN/1.' may, conceptually, have the value '$$$$$1$$$$$'. Obviously, one must come up with a scheme which will allow any possible character string to be represented as a floating-point number. At one time, the floating-point equivalent of the memory address of the character string was used as the actual value of the parameter. This approach led to portability problems and has been abandoned in favor of the following: A character string which is to become the conceptual value of a parameter is stashed in a character-string array inside AUTOGRAPH and a floating-point identifier which will enable later retrieval of the string...
Overview

is stored as the actual value of the parameter. As it happens, such identifiers are always negative; positive values may therefore have other uses. For example, if 'DASH/PATTERN/1.' has a negative value, a character-string dash pattern is implied, but, if it has a positive value, a 16-bit binary dash pattern is implied.

USE OF PARAMETER-ACCESS Routines

The routine AGSETP (AGGETP) is called by a user program to "set" ("get") the floating-point values of a specified group of related parameters.

The routine AGSETF is used to store a floating-point number as the value of a single specified parameter, the routine AGGETF to retrieve the floating-point value of a single specified parameter.

The routine AGSETI is used to store the floating-point equivalent of an integer as the value of a single specified parameter, the routine AGGETI to retrieve the integer equivalent of the value of a single specified parameter.

The routine AGSETC is used to store a character string as the (conceptual) value of a single specified parameter, the routine AGGETC to retrieve the (conceptual) character-string value of a single specified parameter.

SIDE EFFECTS OF PARAMETER-SETTING

Setting certain individual parameters results, as a side effect, in "special action" by the routine AGSETP. For example, when 'BACKGROUND.' is given a new value, other parameters are also changed to create the desired background. These side effects occur whether AGSETP is reached directly from the user program or indirectly, by way of a call to one of the routines AGSETF, AGSETI, or AGSETC. They do not occur when AGSETP is asked to set one of the parameters in question as part of a multi-parameter group, only when it is asked to set that parameter individually.

SAVING AND RESTORING DEFAULTS

Parameters whose values have been changed by the user do not automatically revert to their default values. Re-creating the default state of AUTOGRAPH by resetting individual parameters can become quite tedious. The routines

August 1987
AUTOGRAPH: A Graphing Utility

AGSAVE, which saves the current state of AUTOGRAPH on a file, and AGRSTR, which restores a saved state of AUTOGRAPH from a file, should be used instead. These routines are described in the section "ROUTINES".

SPECIAL VALUES 'NULL/1.' AND 'NULL/2.'

The parameters 'NULL/1.' and 'NULL/2.' define the special values "null 1" and "null 2", which have the default values "1.E36" and "2.E36", respectively. These special values are used in a couple of ways:

- Certain parameters may be given an "actual" value or one of the "null" values. An actual value directly expresses the value of some quantity. The value "null 1" specifies that AUTOGRAPH is to choose an appropriate value to use, but that it is not to store that value in place of the "null 1". The value "null 2" specifies that AUTOGRAPH is to choose an appropriate actual value and store that actual value in place of the "null 2".

  Example: 'Y/MINIMUM.', which specifies the minimum y coordinate, has the default value "null 1", specifying that, for each graph, AUTOGRAPH is to choose (by examining the data) an appropriate minimum y. This parameter may be given an actual (non-null) value, thus imposing a desired minimum y, or it may be given the value "null 2", specifying that AUTOGRAPH is to choose an appropriate minimum y for the next graph and then use that value for following graphs.

- The value "null 1" is used in x/y coordinate data to signal missing points.

Note: If your x/y coordinate data might include the values "1.E36" or "2.E36", your program's first action should be to change the values of 'NULL/1.' and 'NULL/2.' to values which cannot possibly occur in the data.

THE GRAPH WINDOW

The parameters 'GRAPH/LEFT.', 'GRAPH/RIGHT.', 'GRAPH/BOTTOM.', and 'GRAPH/TOP.' serve to locate the edges of a rectangular "graph window" within the plotter frame. The first two are stated as fractions of the frame width, the second two as fractions of the frame height. These parameters have the default values "0.", "1.", "0.", and "1.", respectively,
specifying a graph window which fills the entire plotter frame.

The graph window is the area in which a graph, including labels, is to be drawn. A user program may limit a graph to any selected portion of the plotter frame. For example, changing the values of the parameters in the group 'GRAPH.' to "0.", ".5", ".5", and ".5", limits a graph to the lower left-hand quarter of the frame.

See example 6, in the section "EXAMPLES".

THE GRID WINDOW

The parameters 'GRID/LEFT.', 'GRID/RIGHT.', 'GRID/BOTTOM.', and 'GRID/TOP.' serve to locate the edges of a rectangular area within the graph window; the parameter 'GRID/SHAPE.' specifies the shape of a "grid window", to be centered in, and made as large as possible in, this area. The positions of the left and right edges are stated as fractions of the graph-window width and have default values ".15" and ".95"; the positions of the bottom and top edges are stated as fractions of the graph-window height and also have default values ".15" and ".95". The parameter 'GRID/SHAPE.' has the default value "0.", specifying a grid window which completely fills the area specified by the other parameters. Other values allow one to specify a grid window of any desired rectangular shape or of a shape implied by the x and y coordinate data.

The grid window is the portion of the graph window along the edges of which the axes are to be drawn and within which curves are to be drawn. Numeric and informational labels are ordinarily placed in the portion of the graph window which is outside the grid window. Various positioning parameters are stated in a "grid coordinate system" based on the grid window and curve-point coordinates are given by the user in a "user coordinate system" which maps into the grid window. If 'WINDOW.' has the value "1.", only those curve portions which lie inside the grid window are drawn. Character sizes and label-offset distances are specified as fractions of the smaller dimension of the grid window, so as to be in scale with the rest of the graph.

See example 7, in the section "EXAMPLES".
THE GRID COORDINATE SYSTEM

Internally, AUTOGRAPH makes use of a "grid coordinate system"; the user also makes use of this system at times in setting certain parameter values. The origin of the grid coordinate system is at the lower left-hand corner of the grid window. X coordinates run linearly from "0." to "1." horizontally, and y coordinates linearly from "0." to "1." vertically, in the grid window. Note that coordinate values outside the range (0.,1.) may be used to reference points outside the grid window.

THE USER COORDINATE SYSTEM

Curve-defining points are stated by a user program in the "user coordinate system". Fourteen parameters specify how that user coordinate system is mapped into the grid window. (The parameter 'INVERT.', described below, might be considered a fifteenth.)

The first seven of the fourteen, named 'X/MINIMUM.', 'X/MAXIMUM.', 'X/LOGARITHMIC.', 'X/ORDER.', 'X/NICE.', 'X/SMALLEST.', and 'X/LARGEST.', specify how user x coordinates are to be mapped onto the horizontal axis of the grid window. The default values of these parameters are such that the routine AGSTUP is forced to:

1. Compute, from the user's x-coordinate data, minimum and maximum values Xm and XM.
2. Compute "nice" (rounded) values Xm' and XM' such that the interval (Xm,XM) is completely contained in the interval (Xm',XM').
3. Map Xm' to the left edge, and XM' to the right edge, of the grid window. The mapping is linear.

The other seven parameters, named 'Y/MINIMUM.', 'Y/MAXIMUM.', etc., specify how user y coordinates are to be mapped onto the vertical axis of the grid window. The default values specify a mapping analogous to that of x coordinates.

By changing the values of these fourteen parameters appropriately, a variety of desirable ends may be achieved:
Values of Xm, XM, Ym, and/or YM may be specified, thus limiting the graph to a particular range of interest and/or forcing consistent scaling of a group of graphs. If the selection of these values is left up to AUTOGRAPH, a range of acceptable values may be specified.

Either or both mappings may be made logarithmic. (The logarithms of coordinate values are mapped linearly onto the axis.)

Either or both mappings may be flipped end-for-end. X coordinates may be made to decrease from left to right, y coordinates to decrease from bottom to top.

The rounding process for either or both mappings may be suppressed, forcing curves to be plotted full-scale.

See examples 6, 7, and 8, in the section "EXAMPLES".

HOW TO GRAPH "X AS A FUNCTION OF Y"

The parameter 'INVERT.' has the default value "0.". If it is set to "1." by a user program, the routines AGSTUP and AGCURV will behave as if their x and y arguments had been interchanged. In some sense, this provides a way of plotting "x as a function of y".

This parameter is of principal interest to the users of EZY, EZXY, EZMY, and EZMXY; those users who call the routines AGSTUP and AGCURV directly should probably leave the parameter zeroed.

See example 8, in the section "EXAMPLES".

WHAT A BACKGROUND CONSISTS OF

A background drawn by the routine AGBACK consists of four axes and up to eight informational labels, each of the latter having none or more lines of text; the total number of such lines must not exceed sixteen. Each of these entities is defined by a group of parameters and may be modified in a variety of ways.
THE FOUR AXES

The four axes are positioned along the edges of the grid window. There are a left y axis, a right y axis, a bottom x axis, and a top x axis. Each of the axes consists of a line, major tick marks, minor tick marks, and numeric labels. Numeric labels are placed at major-tick-mark positions.

The axes are defined by the parameter group named 'AXIS.', which has subgroups 'AXIS/LEFT.', 'AXIS/RIGHT.', 'AXIS/BOTTOM.', and 'AXIS/TOP.'. Each of these subgroups contains 23 parameters defining one of the four axes. These 23 parameters fall into six further subgroups, having the associated keywords CONTROL, LINE, INTERSECTION, FUNCTION, TICKS, and NUMERIC.

The default values of the axis parameters specify a "perimeter" background: All four axes are drawn; each has short, inward-pointing major and minor ticks; the left axis and the bottom axis have numeric labels (placed outside the grid window); the right-axis and top-axis numeric labels are suppressed. See examples 1 through 4, in the section "EXAMPLES".

A "half-axis" background is created by suppressing the right axis and the top axis completely. A "grid" background is created by extending the left-axis and bottom-axis ticks all the way across the grid window and suppressing the ticks on the other two axes. The parameter 'BACKGROUND.' allows the user to create these standard backgrounds easily; whenever its value is changed by a user-program call to AGSETP, AGSETF, or AGSETI, parameters in the group 'AXIS.' are modified to create the desired background. See examples 5, 6, and 8, in the section "EXAMPLES".

ABBREVIATED FORM OF AXIS-PARAMETER NAMES

In the ensuing discussions of the various parameters in the group 'AXIS.', the character "s" is used to stand for any one of the keywords "LEFT", "RIGHT", "BOTTOM", or "TOP". For example, 'AXIS/s/LINE.' stands for any one of 'AXIS/LEFT/LINE.', 'AXIS/RIGHT/LINE.', etc. This form is shorter and makes it clear that four different parameters or groups of parameters are being described at once.
THE PARAMETER 'AXIS/s/CONTROL.'

This parameter may be given any integral value from "-1." to "+4.".

- The value "-1." specifies that only the line portion of the axis specified by "s" is to be drawn.
- The value "0." specifies that no portion of the axis is to be drawn.
- A value from "1." to "4." specifies that all portions of the axis are to be drawn and tells AUTOGRAPH what liberties it may take in attempting to cope with numeric labels which will not fit along the axis without overlapping.

The precise meanings of each value are described in the section "PARAMETERS".

THE PARAMETER 'AXIS/s/LINE.'

This parameter has the default value "0.". Setting it to a "1." causes the line portion of the axis specified by "s" to be suppressed. Tick marks and/or numeric labels may still be drawn.

MOVING AN AXIS

Both of the parameters

'AXIS/s/INTERSECTION/GRID.'
'AXIS/s/INTERSECTION/USER.'

have the default value "null 1", specifying that the axis "s" is to be drawn in its normal position, along the edge of the grid window. If either parameter is given a non-null value, the axis "s" is moved away from its normal position in such a way as to intersect the sides of the grid which are perpendicular to it at a point specified by that non-null value. To move an x axis, a y coordinate is specified; to move a y axis, an x coordinate is specified.

The coordinate may be specified in the grid coordinate system or in the user coordinate system, depending on which parameter is used. If both
parameters are given non-null values, the user-system value takes precedence.

No axis may be moved outside the graph window. Attempting to do so moves the axis as far as the edge, but no farther.

See example 8, in the section "EXAMPLES".

THE "LABEL COORDINATE SYSTEM" ALONG AN AXIS

Each of the four axes has associated with it a "user" coordinate system (as described above, in the paragraph "THE USER COORDINATE SYSTEM") and a "label" coordinate system. The routine AGUTOL defines the relationship between the two coordinate systems for each of the four axes.

Tick marks are positioned at "nice" values in the label coordinate system, mapped to the user coordinate system, and then mapped onto the axis. Numeric labels are associated with major ticks and provide values in the label coordinate system.

The default version of AGUTOL defines the label system on each axis to be identical with the user system; a private version may be substituted in place of the default in order to change the label coordinate system for any one or more of the four axes.

Example: Suppose that the y-coordinate data is in miles/hour and it is desired that the left y axis be tick-marked and labeled in meters/second. The user program must include a subroutine AGUTOL with five arguments (four input, one output), as follows:

1. the number of the axis being drawn (1, 2, 3, or 4, implying the left, right, bottom, and top axes, respectively).

2. the value of 'AXIS/s/FUNCTION.' for that axis.

3. an integer specifying whether to map from the user system to the label system (+1) or vice-versa (-1).

4. an input value in one coordinate system.
5. an output value in the other system.

The user routine must, for the left axis only, multiply the input value by the appropriate constant and return the result as the output value; for all other axes, it must return an output value equal to the input value.

It is recommended that the default value of 'AXIS/s/FUNCTION.' (zero) be used to imply that AGUTOL should do the identity mapping for the axis "s"; other values may be used to select desired mappings. This gives a way to "turn off" the use of a special mapping for a given axis.

Note that the tick-marking and labeling of one x (y) axis of a graph may be completely different from that of the other x (y) axis of the graph. For example, the left y axis could be made to indicate "height in kilometers" and the right y axis "pressure in millibars".

See example 7, in the section "EXAMPLES".

POSITIONING OF MAJOR TICK MARKS ON AN AXIS

The parameter group named

'AXIS/s/TICKS/MAJOR/SPACING.'

contains three parameters, with associated keywords TYPE, BASE, and COUNT. These parameters are described in detail in the section "PARAMETERS". Major tick marks may be spaced linearly or logarithmically in the label coordinate system along the axis specified by "s", or suppressed altogether. Each of the TYPE and BASE parameters has the default value "null 1", allowing AUTOGRAPH to position major tick marks as it sees fit.

See examples 7 and 8 and the final example, in the section "EXAMPLES".

APPEARANCE OF MAJOR TICK MARKS

The parameter

'AXIS/s/TICKS/MAJOR/PATTERN.'

has an integral value from "0." to "65535." and specifies the dashed-line
pattern to be used for major ticks on the axis specified by "s". Each "0" bit in the lower 16 bits of the integral value specifies a gap 3 plotter units long, each "1" bit a solid portion 3 plotter units long. The default value "65535." (2 to the 16th minus 1) specifies a solid line. The value "0." may be used to suppress major tick marks on the axis "s".

The parameters

'AXIS/s/TICKS/MAJOR/LENGTH/OUTWARD.,'
'AXIS/s/TICKS/MAJOR/LENGTH/INWARD. '

specify the lengths of the outward-pointing and inward-pointing portions of the major ticks. Each is stated as a fraction of the smaller dimension of the grid window. If either of these values is made greater than or equal to "1.", it specifies a tick-mark portion which extends to the edge of the grid window and a little beyond, the magnitude of the "little beyond" being specified by the fractional portion of the parameter value. The default values give inward-pointing major ticks of length ".015" on all axes.

See example 8, in the section "EXAMPLES".

POSITIONING/APPEARANCE OF MINOR TICK MARKS

The parameter

'AXIS/s/TICKS/MINOR/SPACING. '

specifies the number of minor ticks which are to occur between pairs of major ticks on the axis specified by "s". Minor ticks are equidistantly spaced in the label coordinate system for that axis. The default value of this parameter is "null 1", allowing AUTOGRAPH to position minor ticks as it sees fit.

The parameters

'AXIS/s/TICKS/MINOR/PATTERN. ',
'AXIS/s/TICKS/MINOR/LENGTH/OUTWARD. '
'AXIS/s/TICKS/MINOR/LENGTH/INWARD. '

specify the dashed-line pattern, outward-pointing length, and inward-
pointing length of minor ticks. They are defined in the same way as the analogous major-tick parameters, except that the default inward-pointing tick length is ".010".

See the final example, in the section "EXAMPLES".

NUMERIC LABELS ON AN AXIS

The parameter group named 'AXIS/s/NUMERIC.' contains eight parameters describing the numeric labels on the axis specified by "s". These parameters are described in detail in the section "PARAMETERS"; they are described sketchily in succeeding paragraphs.

TYPES OF NUMERIC LABELS

The parameter named 'AXIS/s/NUMERIC/TYPE.' may be given any integral value from "0." to "3." or one of the values "null 1" or "null 2".

- The value "0." suppresses numeric labels on the axis specified by "s".

- The values "1.", "2.", and "3." specify the use of "scientific notation", "exponential notation", and "no-exponent notation", respectively.

- A null value gives AUTOGRAPH the freedom to use one of the values "1.", "2.", or "3." - whichever is most consistent with the label coordinate system along the axis.

The exact nature of the labels produced by a given value depends on the three parameters

'AXIS/s/TICKS/MAJOR/SPACING/TYPE.'
'AXIS/s/NUMERIC/EXponent.'
'AXIS/s/NUMERIC/FRACTION.'

See example 7 and the final example, in the section "EXAMPLES".

August 1987
ORIENTATION OF NUMERIC LABELS

The parameters

'AXIS/s/NUMERIC/ANGLE/1ST.'
'AXIS/s/NUMERIC/ANGLE/2ND.'

may have integral values "0.", "90.", "180.", or "270.". They specify the user's first and second choices for the orientation of numeric labels on the axis specified by "s". AUTOGRAPH will attempt to use the first choice (default value - "0." for all axes); if that leads to overlap problems and shrinking the labels either doesn't help or is not permitted and rotation is permitted (by the setting of 'AXIS/s/CONTROL.'), AUTOGRAPH may try the second choice (default value - "90." for all axes).

The values given represent angles measured in degrees counter-clockwise from horizontal.

POSITIONING OF NUMERIC LABELS

The parameter

'AXIS/s/NUMERIC/OFFSET.'

specifies on which side of the axis specified by "s" the numeric labels are to lie and the size of the gap to be left between the axis line and the numeric labels.

- A negative value specifies labels inside the grid window.

- A zero value specifies labels centered on the axis, suppresses the line portion of the axis, and moves the inward-pointing and outward-pointing portions of ticks out away from the axis so as to leave room for the labels.

- A positive value specifies labels outside the grid window.

The magnitude of the value specifies the distance from the axis to the nearest portion of the label, stated as a fraction of the smaller side of the grid window.
The default value for all axes is ".015".

CHARACTER SIZES IN NUMERIC LABELS

The parameters

'AXIS/s/Numeric/Width/Mantissa.'
'AXIS/s/Numeric/Width/Exponent.'

specify the widths of characters in the mantissa and exponent portions of the numeric labels on the axis specified by "s", stated as fractions of the smaller dimension of the grid window.

The sizes specified are those desired by the user. If an overlap problem arises and 'AXIS/s/Control.' is set so as to allow AUTOGRAPH to shrink the numeric labels, the characters may end up smaller than desired. No character is shrunk to less than the minimum readable size, however.

These parameters have default values ".015" and ".010", respectively, for all axes.

INFORMATIONAL LABELS

As many as m informational labels may be defined at any one time; normally, m = 8. The informational labels form a part of the background produced by a call to the routine AGBACK. Each of the informational labels is defined as follows:

1. Each label has a name - a short character string which uniquely identifies the label.

2. Each label has a "suppression flag", which may be set to enable or disable plotting of the label.

3. Each label is positioned relative to a "basepoint", whose x and y coordinates are specified in the grid coordinate system. Normally, the basepoint lies on one edge - but not at a corner point - of the grid window.
Emanating from the label basepoint is an "offset vector", whose x and y components are specified as signed fractions of the smaller dimension of the grid window. Normally, the offset vector is used to specify the size of the gap to be left between an informational label and the edge of the grid window. The presence or absence of an axis along that edge of the grid window is ignored when specifying this gap; see the paragraph "BACKGROUND OVERLAP PROBLEMS", below.

Emanating from the end of the offset vector is a "baseline", whose direction is specified as an angle in degrees ("0.", "90.", "180.", or "270.", measured counter-clockwise from horizontal). The text lines of a label are written parallel to, and in the same direction as, the baseline.

A centering option for each label determines whether the left edges, the centers, or the right edges, of the text lines are aligned with the end of the offset vector.

Each label may contain one or more text lines (or none). The total number of text lines in all labels must not exceed n - normally, n = 16. Each of the text lines is defined as follows:

Each text line has an integral position number which distinguishes it from every other line in the same label. Multiples of "100." are recommended. Lines with positive position numbers are drawn above the label baseline, lines with negative position numbers below the label baseline. A line with position number "0." is centered on the label baseline. "Above" and "below" are defined here from the viewpoint of a reader of the label. The position numbers of the lines in a label specify the order in which the lines appear - strictly decreasing from top to bottom - but do not determine the interline spacing, which is set by AUTOGRAPH itself.

Each line has a "suppression flag", which may be set so as to enable or disable drawing of the line.

Each line has a character-width specifier, stated as a fraction of the smaller dimension of the grid window.
The text of each line is defined by a character string and a count of the number of characters in the string — normally determined by AUTOGRAPH itself.

Note: The (EDITOR-style) string replacements

'FLLB(10,8) '='FLLB(10,m)'
'QBIM / 8.'='QBIM / m.'

where m is greater than or equal to 5, may be applied to the AUTOGRAPH source file to provide for a maximum of m labels. Similarly, the string replacements

'FLLN(6,16) '='FLLN(6,n)'
'QNIM / 16.'='QNIM / n.'

where n is greater than or equal to 5, may be applied to the AUTOGRAPH source file to provide for a maximum of n lines.

THE PREDEFINED LABELS

The section "PARAMETERS" describes in detail four "predefined" labels, named 'R', 'L', 'B', and 'T'. Each of these labels lies along one of the four edges of the grid window — the left edge, the right edge, the bottom edge, or the top edge.

The predefined labels greatly simplify the task of generating labels along the edges of the grid window. For example, if you want a "header label" above the grid window, you need only specify the desired character string to define the text of line number "100." of the label named 'T'.

The default definitions of the predefined labels specify a label reading "X" below the grid window and a label reading "Y" to the left of the grid window.

See examples 5 and 7 and the final example, in the section "EXAMPLES".
THE PARAMETER GROUP 'LABEL.'

The parameter group 'LABEL.' contains $10m+3$ parameters - normally, $m = 8$. Together with the parameters in the group 'LINE. ', they define the informational labels to be drawn by a call to the routine AGBACK. The parameters in the group 'LABEL.' are as follows:

- The parameter 'LABEL/CONTROL.' may be given the value "-1." to delete all currently-defined labels, the value "0." to temporarily disable the drawing of labels, the value "1." to enable the drawing of labels and prevent the shrinkage of labels when overlap problems arise, or the value "2." to enable the drawing of labels and allow shrinkage. The default value is "2.". See examples 5 and 6, in the section "EXAMPLES".

- The parameter 'LABEL/BUFFER/LENGTH.' should not normally be set by a user program. Its value is $m$, the maximum number of labels AUTOGRAPH can handle.

- The subgroup 'LABEL/BUFFER/CONTENTS.' consists of $10m$ words, in which the label definitions are stored. Normally, a user program should not attempt to store values in this block directly. See the paragraph "HOW TO ACCESS A LABEL DEFINITION", below.

- The parameter 'LABEL/NAME.' is used in the process of accessing a label definition. It functions as a switch, pointing to the label definition currently being accessed.

See examples 5 and 7 and the final example, in the section "EXAMPLES".

See the section "PARAMETERS" for further information about these parameters.

THE PARAMETER GROUP 'LINE.'

The parameter group 'LINE.' contains $6n+4$ parameters - normally, $n = 16$. They define the lines belonging to the various labels. The parameters in the group 'LINE.' are as follows:
The parameters 'LINE/MAXIMUM.' and 'LINE/END.' define the assumed maximum line length (default - 40 characters) and the line end character (default - '$'). These parameters come into play when a user program defines the text of a line. The character string tendered by the user is assumed to be of maximum length; if it is really shorter than that, it must be followed by the line end character. See example 8 and the final example, in the section "EXAMPLES".

The parameter 'LINE/BUFFER/LENGTH.' should not normally be set by a user program. Its value is n, the maximum number of lines AUTOGRAPH can handle.

The subgroup 'LINE/BUFFER/CONTENTS.' consists of 6n words, in which the line definitions are stored. Normally, a user program should not attempt to store values in this block directly. See the paragraph "HOW TO ACCESS A LINE DEFINITION", below.

The parameter 'LINE/NUMBER.' is used in the process of accessing a line definition. It functions as a switch, pointing to the line definition currently being accessed.

See the section "PARAMETERS" for further information about these parameters.

See examples 5 and 7 and the final example, in the section "EXAMPLES".

ACCESSING A LABEL DEFINITION

To access a label definition, a user program must first execute an AGSETC call to store the name of the label as the value of 'LABEL/NAME.' Such a call does not actually store the name as the value of that parameter. Instead, it causes the label buffer to be searched for the definition of the named label. If that definition is not found, a default definition is made up and inserted in the label buffer. In any case, the index of the definition is floated and stored as the value of the parameter 'LABEL/NAME.'.

Once 'LABEL/NAME.' has been set in this manner, the parameter group name 'LABEL/DEFINITION.' and subgroup names of the form 'LABEL/DEFINITION/...' may be used to access the parameters defining the label. These parameters...
are as follows:

- The parameter 'LABEL/DEFINITION/SUPPRESSION.' may be given the value "-2." to delete the label and all of its lines, the value "-1." to delete the lines of the label but leave the label itself defined, the value "0." to enable drawing of the label, and the value "1." to temporarily suppress drawing of the label. It has the default value "0.". When a label is deleted, 'LABEL/NAME.' and 'LINE/NUMBER.' become undefined; similarly, when the lines of a label are deleted, 'LINE/NUMBER.' becomes undefined.

- The parameters 'LABEL/DEFINITION/BASEPOINT/X.' and 'Y.' specify the coordinates, in the grid coordinate system, of the label's basepoint. The default basepoint is (.5,.5).

- The parameters 'LABEL/DEFINITION/OFFSET/X.' and 'Y.' specify the components of the label's offset vector, as signed fractions of the smaller dimension of the grid window. The default vector has zero components.

- The parameter 'LABEL/DEFINITION/ANGLE.' specifies the angle ("0.", "90.", "180.", or "270.") at which the label's baseline emanates from the end of its offset vector. The default angle is "0.".

- The parameter 'LABEL/DEFINITION/CENTER.' has the value "-1." to align the left ends, the value "0." to align the centers, and the value "+1." to align the right ends, of the lines of the label with the end of its offset vector. The default value is "0.".

- The parameters 'LABEL/DEFINITION/LINES.' and 'LABEL/DEFINITION/INDEX.' are not normally set by a user program; they are maintained by AUTOGRAPH. The former specifies the number of lines belonging to the label and the latter specifies the index (in the line buffer) of the definition of the first line belonging to the label. A default label has no lines - both of these parameters are zeroed.

See the section "PARAMETERS" for further information about these parameters.
Overview

See examples 5 and 7 and the final example, in the section "EXAMPLES".

ACCESSING A LINE DEFINITION

To access the definition of one of the lines of a label, a user program must first access the label definition by setting 'LABEL/NAME.', as described above. Then, it must execute an AGSETP (or AGSETI) call to store the number of the desired line as the value of 'LINE/NUMBER.'. Such a call does not actually store the specified number as the value of that parameter. Instead, it causes the line buffer to be searched for the definition of the desired line. If that definition is not found, a default definition is made up, inserted in the line buffer, and added to the linked list of definitions of lines belonging to the label. In any case, the index of the definition is floated and stored as the value of 'LINE/NUMBER.'.

Once 'LINE/NUMBER.' has been set in this manner, the parameter group name 'LINE/DEFINITION.' and subgroup names of the form 'LINE/DEFINITION/...' may be used to access the parameters defining the line. These parameters are as follows:

- The parameter 'LINE/DEFINITION/SUPPRESSION.' may be given the value "-1." to delete the line, the value "0." to enable drawing of the line, and the value "+1." to temporarily disable drawing of the line. It has the default value "0.". When a line is deleted, 'LINE/NUMBER.' becomes undefined.

- The parameter 'LINE/DEFINITION/CHARACTER-WIDTH.' specifies the desired width of each character in the line, stated as a fraction of the smaller dimension of the grid window. The default width is "0.015".

- The parameter 'LINE/DEFINITION/TEXT.' identifies the character string comprising the text of the label. The default value is a single blank.

- The parameter 'LINE/DEFINITION/LENGTH.' specifies the length of the character string. The default value is a "1.".

- The parameter 'LINE/DEFINITION/INDEX.' is not normally set by a user program. It is maintained by AUTOGRAPH and specifies the index (in the line buffer) of the next line belonging to the label.
AUTOGRAPH: A Graphing Utility

See the section "PARAMETERS" for further information about these parameters. See examples 5 and 7 and the final example, in the section "EXAMPLES".

Note: As a convenience to the user, an AGSETC call to set 'LINE/DEFINITION/TEXT.' sets both the parameters defining the text of the label - '...TEXT.' and '...LENGTH.' - the latter being computed by examining the string. The string must be no longer than the length specified by the value of 'LINE/MAXIMUM.' and, if shorter, it must be followed by the 'LINE/END.' character.

THE LABEL BOXES

Each informational label is considered to lie in one of six "label boxes", as follows:

- Box 1 lies to the left of the grid window. It contains all labels which have a basepoint on the left edge of the grid window and a leftward-pointing offset vector.

- Box 2 lies to the right of the grid window. It contains all labels which have a basepoint on the right edge of the grid window and a rightward-pointing offset vector.

- Box 3 lies below the grid window. It contains all labels which have a basepoint on the bottom edge of the grid window and a downward-pointing offset vector.

- Box 4 lies above the grid window. It contains all labels which have a basepoint on the top edge of the grid window and an upward-pointing offset vector.

- Box 5 lies in the interior of the grid window. It contains all labels which have a basepoint on some edge of the grid window and an inward-pointing offset vector.

- Box 6 covers the entire grid window and contains all of the remaining labels.
Three restrictions must be observed by the user: First, no label's basepoint may have coordinates (0.,0.), (0.,1.), (1.,0.), or (1.,1.); these corner points must be avoided. Second, no portion of any label in boxes 1 through 4 may lie inside the grid window. Third, no portion of any label in box 5 may lie outside the grid window.

The label-box concept is important in handling overlap problems, which are discussed in the next paragraph.

BACKGROUND OVERLAP PROBLEMS

The responsibility for avoiding background overlap problems might reasonably have been placed squarely on the shoulders of the user, except for one unpleasant fact: numeric labels are unpredictable critters. Accordingly, AUTOGRAPH accepts a part of the burden.

In attempting to keep the numeric labels on a given axis from overlapping each other, AUTOGRAPH may shrink and/or reorient them. Either or both of these actions may be suppressed by the user by resetting 'AXIS/s/CONTROL.'

If a problem still exists, some of the labels may be omitted - perhaps leaving only every second one, every third one, every fourth one, etc.

Informational labels are positioned by the user along the edges of the grid window as if numeric labels did not exist. AUTOGRAPH takes the following actions in attempting to prevent the informational labels from overlapping the numeric labels on any axis:

1. Box 1 labels (to the left of the grid window) are moved leftward, box 2 labels (to the right of the grid window) are moved rightward, box 3 labels (below the grid window) are moved downward, box 4 labels (above the grid window) are moved upward, and box 5 labels (inside the grid window) are moved inward. Box 6 labels are not moved.

2. If, during step 1, a label is shoved outside the graph window by the numeric labels on some axis, those numeric labels may be shrunk and/or re-oriented, as allowed by the user's setting of 'AXIS/s/CONTROL.'.

3. If one or more of the labels in a given box still lies partly outside the graph window, the labels in that box may be shrunk, depending on the current setting of 'LABEL/CONTROL.'. Each label in the box
shrinks toward the end of its offset vector.

4. If one or more of the labels in boxes 1 through 4 still lies partly outside the graph window, all of the labels in that box may be moved inward, shoving numeric labels ahead of them onto, and perhaps across, an axis.

The algorithms used to do all of this are not perfect; if pushed too severely, they may fail to produce an esthetically pleasing or even minimally acceptable graph. In such cases, the user must take remedial action.

Note: None of the actions described above modify any of the parameters except 'AXIS/S/NUMERIC/ANGLE/1ST.', which may be negated by subtracting a multiple of "360.". Also, no label is shrunk to less than a readable size.

DASHED-LINE PATTERNS FOR CURVES

The subroutine AGCURV draws curves (one per call). It does this by issuing calls to the routines DASHD, FRSTD, VECTD, and LASTD, in the DASHCHAR package. Each curve may thus be drawn using its own particular dashed-line pattern. One of AGCURV's arguments, called KDSH, specifies the dashed-line pattern to be used for a given curve:

- If KDSH is zero, the caller is assumed to have done his own call to DASHD. AGCURV does not call it.
- If KDSH is non-zero, AGCURV calls DASHD.
- If KDSH is positive, its value (modulo n) specifies one of n "user" patterns, defined by the parameter group named 'DASH.' See example 7, in the section "EXAMPLES".
- If KDSH is negative, its absolute value (modulo 26) specifies one of 26 "alphabetic" patterns. The curve is drawn using a solid line which is interrupted periodically by the selected letter of the alphabet. See example 8, in the section "EXAMPLES".

The nature of the "user" set of dashed-line patterns is discussed in the next paragraph.
THE PARAMETER GROUP 'DASH.'

The following parameters all belong to the group 'DASH.:

- The parameter 'DASH/SELECTOR.' specifies the type of dashed-line patterns to be used by the routines EZMY and EZMXY (EZY and EZXY always use the first of the "user" patterns). If the value of 'DASH/SELECTOR.' is zero or negative, the alphabetic set of (26) dashed-line patterns will be used; if its value is positive and has magnitude "n", then the first "n" of the "user" patterns will be used. The default value is "1."

- The parameter 'DASH/LENGTH.' specifies the assumed length of character-string dashed-line patterns; it must be set to the proper value prior to any AGSETC call setting one of the "user" dashed-line patterns 'DASH/PATTERN/n.'. The default value is "8."

- The parameter 'DASH/CHARACTER.' specifies the curve length devoted to a character other than a dollar sign or a quote in a character-string dashed-line pattern, stated as a fraction of the smaller side of the grid window. The default value is ".010".

- The parameter 'DASH/DOLLAR-QUOTE.' specifies the curve length devoted to a dollar sign (a solid section of the line) or a quote (a gap in the line), stated as a fraction of the smaller side of the grid window. The default value is ".010".

- Each of the parameters 'DASH/PATTERN/1.', 'DASH/PATTERN/2.', etc., up to 'DASH/PATTERN/26.' defines one of the dash patterns in the "user" group. Either of the routines AGSETT or AGSETF may be used to give one of these parameters a positive integral value between 0. and 65535., inclusive, in which case the low-order 16 bits of it are interpreted as a dash pattern; the 0 and 1 bits represent "pen-up" and "pen-down" segments three plotter units long. The routine AGSETC may be used to (in effect) store a character string (of the length specified by 'DASH/LENGTH.') as the value of one of these parameters. In such a character string, a quote represents a "pen-up" segment, a dollar sign represents a "pen-down" segment, and every other character is simply drawn as a part of the line. The default value of all 26 parameters is "65535.".

August 1987
AUTOGRAPH: A Graphing Utility

See examples 7 and 8, in the section "EXAMPLES".

PATTERNS USED BY EZY, EZXY, EZMY, AND EZMXY

Each of the routines EZY and EZXY, which draw one curve per call, calls AGCURV with KDSH equal to 1, specifying the use of the first of the "user-defined" set of dashed-line patterns (default - a solid line) for the single curve to be drawn.

Each of the routines EZMY and EZMXY, which draw one or more curves per call, calls AGCURV with KDSH equal to ISIGN(I,IDSH), where I is the number of the curve being drawn and IDSH is the integral value of 'DASH/SELECTOR.' This parameter has the default value "1.", specifying the use of the "user" set of dashed-line patterns (default - solid lines); the value "-1." specifies the use of the "alphabetic" set.

WINDOWING OF CURVES

The parameter 'WINDOW.' has the default value "0.". If it is set to a "1." by a user program, curves subsequently drawn by the routine AGCURV are "windowed". This means that only those portions lying inside the grid window are drawn; the effect is as if one were viewing the curve through an actual window.

See example 7, in the section "EXAMPLES".

USE OF PWRITX BY AUTOGRAPH

Normally, the routine PWRIT is used for all characters drawn by AUTOGRAPH. Actually, a routine AGPWRT is called; the default version of that routine just passes its arguments on to PWRIT. Trying to use PWRITX instead poses some problems. Because "function codes" may be used in a text string passed to PWRITX, the length of the string cannot be taken to match the actual number of characters to be drawn; moreover, PWRITX does not use the same plotter width for each character in a string. Thus, AUTOGRAPH cannot properly predict where on a graph a label drawn by PWRITX lies, which interferes with its handling of overlap problems; also, strings which are positioned relative to an end-point may not be properly aligned.
Nevertheless, there is a way to use PWRITX. The XLIB file AGUPWRTX contains a version of AGPWRT which does it. A string which is centered relative to a given position (like the "x-axis label", the "y-axis label", or the "graph label") is drawn directly, in its entirety, by PWRITX, and may therefore contain function codes to get Greek characters, subscripts, superscripts, etc. A string which is positioned relative to one end (which includes all numeric labels) is drawn by passing one character at a time to PWRITX, so that the same plotter width will be used for each; function codes must not be included in such strings. The results, while not as pleasing as one would normally expect from PWRITX, are more than just acceptable.

On the Cray, use the following JCL to get the source for this version of AGPWRT and compile it, so that it will replace the default version:

\begin{verbatim}
GETSRC,LIB=XLIB,FILE=AGUPWRTX,L=UPWRTX.
CFT,I=UPWRTX,L=0.
\end{verbatim}

The utility support file AGUPWRTX, on the distribution tape for NCAR Graphics, also contains this version of AGPWRT.

In my opinion, the "duplex" character set of PWRITX is far superior to the "complex" set. At present, one uses it by incorporating the following code at the beginning of one's program, prior to any call to PWRITX (directly or indirectly):

\begin{verbatim}
COMMON /PUSER/ MODE
  : :
  MODE = 1
\end{verbatim}

VARYING INTENSITIES, COLORS, ETC.

Three routines - AGCHAX, AGCHCU, and AGCHIL (the default versions of which do nothing) - are provided solely to be replaced by the user; the replacement versions may change intensities, line widths, colors, line styles, etc., for selected portions of a graph. Each is called just before an object is to be drawn and again just after it has been drawn, with arguments enabling the user version to completely identify what the current situation is and to make the appropriate calls. AGCHAX handles objects which are parts of axes, AGCHCU handles curves, and AGCHIL handles.
informational labels.

See the descriptions of these routines, in the section "ROUTINES".

NON-STANDARD NUMERIC LABELS

The routine AGCHNL (the default version of which does nothing) is called just after the character form of a numeric label has been constructed and just before it is to be drawn. The user may supply a version of this routine to transform selected numeric labels in any desired fashion and return them to AUTOGRAPH. This feature may be used to label an axis with the names of the months, Roman numerals, etc.

See the description of the routine AGCHNL, in the section "ROUTINES" and example 10, in the section "EXAMPLES".

SCATTERGRAMS AND HISTOGRAMS

Scattergrams, histograms, and other such specialized "graphs" are not directly provided for by AUTOGRAPH. Standard procedure is to suppress the advancing of the frame and the drawing of curves by EZ..., call EZ... with the appropriate x and y data to generate the background, draw the desired objects on that background, and then advance the frame.

See examples 11 and 12, in the section "EXAMPLES".

August 1987
Section II: Routines
This section describes all of the AUTOGRAPH routines of interest to the user. With two exceptions, they are subroutines rather than functions.

The subroutines EZY, EZXY, EZMY, and EZMXY provide a quick-and-dirty graph-drawing capability. The appearance of a graph drawn by one of these routines may be changed drastically by changing the values of primary control parameters.

The subroutines ANOTAT and DISPLA are provided principally for historical reasons. Each allows the user to provide new values for certain primary control parameters. These parameters may also be set by calls to lower-level routines.

The subroutines AGSETP, AGSETF, AGSETI, and AGSETC allow the user to set the values of parameters.

The subroutines AGGETP, AGGETF, AGGETI, and AGGETC allow the user to get the current values of parameters.

The subroutine AGSTUP must be called prior to calling AGBACK or AGCURV, both initially and after making any change in the primary control parameters. It examines the primary control parameters for consistency and computes the values of required secondary control parameters.

The subroutine AGBACK is called to draw a background.

The subroutine AGCURV is called to draw a single curve.

The subroutines AGSAVE and AGRSTR are called to save/restore the current state (commonly the default state) of AUTOGRAPH.

The function AGBNCH may be used to obtain the 16-character dash pattern which is equivalent to a specified 16-bit integer dash pattern.

The function AGDSHN may be used to obtain the 16-character name of a particular specified dash-pattern parameter.
AUTOGRAPH: A Graphing Utility

The subroutine AGUTOL is called by AUTOGRAPH to do the mapping from the user system to the label system (or vice-versa) along the four axes. The default version may be replaced by the user to obtain a desired mapping.

The subroutines AGCHAX, AGCHCU, and AGCHIL are called by AUTOGRAPH just before and just after drawing a particular element of a graph. The default do-nothing versions may be replaced by a user in order to obtain desired effects (color, different line styles, etc.).

The subroutine AGCHNL is called by AUTOGRAPH just after the character representation of a numeric label has been generated and just before it is to be drawn. The default do-nothing version may be replaced by a user in order to change the numeric labels in a desired way (to get names of months, Roman numerals, etc.).

The subroutine AGPWRT is called by AUTOGRAPH to draw a character string. The default version just calls the plot-package routine PWRIT. This routine may be replaced by a user version which calls PWRITX, PWRITY, or some other character-drawer.

Note: User versions of AGUTOL, AGCHAX, AGCHCU, AGCHIL, AGCHNL, and AGPWRT should not call any other AUTOGRAPH routine. No such call will have a useful effect, and, at worst, an infinite loop may result.

EZY(YDRA, NPTS, GLAB)

Draws, in a manner determined by the current values of the control parameters, a complete graph of a single curve through the points (I,YDRA(I)), for I from 1 to NPTS. The argument GLAB may be used to specify a "graph label", to be placed at the top of the graph.

EZY - USAGE

If the default values of the parameters are unchanged, calling EZY produces a graph having the following appearance: A "perimeter" background outlines a grid window which is 8/10 the width and 8/10 the height of the plotter frame and positioned slightly above and to the right of center within it. Each edge of the perimeter has short inward-pointing major and minor tick
marks, with major tick marks occurring at the ends of each edge. Numeric labels below major tick marks on the bottom edge of the perimeter, increasing in value from left to right, show the linear mapping of values of I onto the horizontal (x) axis of the graph; below them is the label "X". Numeric labels to the left of major tick marks on the left edge of the perimeter, increasing in value from bottom to top, show the linear mapping of values of YDRA(I) onto the vertical (y) axis of the graph; to the left of them is the label "Y". Above the perimeter is the label specified by "GLAB", if any. The curve itself is drawn as a solid line within the perimeter. A frame advance is done after the graph is drawn.

See example 1, in the section "EXAMPLES".

The appearance of a graph drawn by EZY may change greatly in response to parameter changes. See the routines ANOTAT, DISPLA, and AGSETP, below, and the section "PARAMETERS".

EZY - ARGUMENTS

REAL YDRA(NPTS); INTEGER NPTS; CHARACTER*(*) GLAB.

YDRA is a one-dimensional array of NPTS floating-point numbers, each of which defines the user coordinates of a point (FLOAT(I),YDRA(I)) on the desired curve. The current value of 'NULL/1.' (default value "1.E36") may be used in YDRA to signal missing points; curve segments on either side of a missing point are omitted.

NPTS is the number of curve points defined by the array YDRA.

GLAB is a character expression defining a new "graph label". (If the first character of this expression is "CHAR(0)", no new "graph label" is defined; the current one will continue to be used.) A character string defining a new graph label must either be of the exact length specified by the current value of 'LINE/MAXIMUM.' (default - 40 characters), or shorter; if shorter, it must be terminated by the character defined by the current value of 'LINE/END.' (default - a '$'). The string becomes the new text of line number 100 of the label named 'T'.

August 1987
EZXY(XDRA, YDRA, NPTS, GLAB)

Draws, in a manner determined by the current values of the control parameters, a complete graph of a single curve through the points (XDRA(I), YDRA(I)), for I from 1 to NPTS. The argument GLAB may be used to specify a "graph label", to be placed at the top of the graph.

EZXY - USAGE

If the default values of the parameters are unchanged, calling EZXY produces a graph having the following appearance: A "perimeter" background outlines a grid window which is 8/10 the width and 8/10 the height of the plotter frame and positioned slightly above and to the right of center within it. Each edge of the perimeter has short inward-pointing major and minor tick marks, with major tick marks occurring at the ends of each edge. Numeric labels below major tick marks on the bottom edge of the perimeter, increasing in value from left to right, show the linear mapping of values of XDRA(I) onto the horizontal (x) axis of the graph; below them is the label "X". Numeric labels to the left of major tick marks on the left edge of the perimeter, increasing in value from bottom to top, show the linear mapping of values of YDRA(I) onto the vertical (y) axis of the graph; to the left of them is the label "Y". Above the perimeter is the label specified by "GLAB", if any. The curve itself is drawn as a solid line within the perimeter. A frame advance is done after the graph is drawn.

See example 2, in the section "EXAMPLES".

The appearance of a graph drawn by EZXY may change greatly in response to parameter changes. See the routines ANOTAT, DISPLA, and AGSETP, below, and the section "PARAMETERS".

EZXY - ARGUMENTS

REAL XDRA(NPTS), YDRA(NPTS); INTEGER NPTS; CHARACTER*(*) GLAB.

XDRA is a one-dimensional array of NPTS floating-point numbers, defining the x coordinates of points on the curve.

YDRA is a one-dimensional array of NPTS floating-point numbers, defining the y coordinates of points on the curve.
The points on the curve have coordinates \((XDRA(I), YDRA(I))\), for \(I\) from 1 to \(NPTS\). The current value of 'NULL/1.' (default value "1.E36") may be used to signal missing data in these arrays. If either coordinate of a point is missing, the point is considered to be missing; curve segments on either side of a missing point are not drawn. Note: Because all non-missing coordinates are used in figuring the minimum and maximum user values along a given axis, it is safest to mark both coordinates as "missing".

\(NPTS\) is the number of curve points defined by the arrays \(XDRA\) and \(YDRA\).

\(GLAB\) is a character expression defining a new "graph label". (If the first character of this expression is "CHAR(0)", no new "graph label" is defined; the current one will continue to be used.) A character string defining a new graph label must either be of the exact length specified by the current value of 'LINE/maximum.' (default - 40 characters), or shorter; if shorter, it must be terminated by the character defined by the current value of 'LINE/end.' (default - a '$'). The string becomes the new text of line number 100 of the label named 'T'.

\[\text{EZMY}(YDRA, IDXY, MANY, NPTS, GLAB)\]

Draws, in a manner determined by the current values of the control parameters, a complete graph of one or more curves, each defined by a set of points \((I, YDRA(I,J))\) (or \((I, YDRA(J,I))\), depending on the current value of 'ROW.'), for \(I\) from 1 to \(NPTS\). The curve number \(J\) runs from 1 to \(MANY\). The argument \(GLAB\) may be used to specify a "graph label", to be placed at the top of the graph.

\[\text{EZMY - USAGE}\]

If the default values of the parameters are unchanged, calling \text{EZMY} produces a graph having the following appearance: A "perimeter" background outlines a grid window which is 8/10 the width and 8/10 the height of the plotter frame and positioned slightly above and to the right of center within it. Each edge of the perimeter has short inward-pointing major and minor tick marks, with major tick marks occurring at the ends of each edge. Numeric labels below major tick marks on the bottom edge of the perimeter, increasing in value from left to right, show the linear mapping of values.
AUTOGRAPH: A Graphing Utility

of I onto the horizontal (x) axis of the graph; below them is the label "X". Numeric labels to the left of major tick marks on the left edge of the perimeter, increasing in value from bottom to top, show the linear mapping of values of YDRA(I,J) onto the vertical (y) axis of the graph; to the left of them is the label "Y". Above the perimeter is the label specified by "GLAB", if any. The curves themselves are drawn as solid lines within the perimeter. A frame advance is done after the graph is drawn.

See example 3, in the section "EXAMPLES".

The appearance of a graph drawn by EZMY may change greatly in response to parameter changes. See the routines ANOTAT, DISPLA, and AGSETP, below, and the section "PARAMETERS".

EZMY - ARGUMENTS

REAL YDRA(IDXY,MANY) or YDRA(IDXY, NPTS) (depending on the current value of 'ROW.'); INTEGER IDXY, MANY, NPTS; CHARACTER(*) GLAB.

YDRA is a two-dimensional array of curve-point y coordinates. The current value of 'NULL/1.' (default value "1.E36") may be used in YDRA to signal missing points; curve segments on either side of a missing point are omitted.

If 'ROW.' is positive (the default), the first subscript of YDRA is a point number and the second is a curve number. If 'ROW.' is negative, the order of the subscripts is reversed ("row-wise", as opposed to "column-wise", storage).

IDXY is the first dimension of the array YDRA, required by EZMY in order to index the array properly.

If 'ROW.' is positive (the default), IDXY must be greater than or equal to NPTS; otherwise, IDXY must be greater than or equal to MANY.

MANY is the number of curves to be drawn by EZMY.

NPTS is the number of points defining each curve to be drawn by EZMY.
GLAB is a character expression defining a new "graph label". (If the first
character of this expression is "CHAR(0)", no new "graph label" is defined;
the current one will continue to be used.) A character string defining a
new graph label must either be of the exact length specified by the current
value of 'LINE/MAXIMUM.' (default - 40 characters), or shorter; if shorter,
it must be terminated by the character defined by the current value of
'LINE/END.' (default - a '$'). The string becomes the new text of line
number 100 of the label named 'T'.

EZMXY(XDRA,YDRA,IDXY,MANY,NPTS,GLAB)

Draws, in a manner determined by the current values of the control parame-
ters, a complete graph of one or more curves, each defined by a set of
points (XDRA(I),YDRA(I,J)) (or (XDRA(I),YDRA(J,I)) or (XDRA(I,J),YDRA(I,J))
or (XDRA(J,I),YDRA(J,I)), depending on the current value of 'ROW.'), for I
from 1 to NPTS. The curve number J runs from 1 to MANY. The argument GLAB
may be used to specify a "graph label", to be placed at the top of the
graph.

EZMXY - USAGE

If the default values of the parameters are unchanged, calling EZMXY pro-
duces a graph having the following appearance: A perimeter background out-
lines a grid window which is 8/10 the width and 8/10 the height of the
plotter frame and positioned slightly above and to the right of center
within it. Each edge of the perimeter has short inward-pointing major and
minor tick marks, with major tick marks occurring at the ends of each edge.
Numeric labels below major tick marks on the bottom edge of the perimeter,
increasing in value from left to right, show the linear mapping of values
of XDRA(I) onto the horizontal (x) axis of the graph; below them is the
label "X". Numeric labels to the left of major tick marks on the left edge
of the perimeter, increasing in value from bottom to top, show the linear
mapping of values of YDRA(I,J) onto the vertical (y) axis of the graph; to
the left of them is the label "Y". Above the perimeter is the label speci-
fied by "GLAB", if any. The curves themselves are drawn as solid lines
within the perimeter. A frame advance is done after the graph is drawn.
The appearance of a graph drawn by EZMXY may change greatly in response to parameter changes. See the routines ANOTAT, DISPLA, and AGSETP, below, and the section "PARAMETERS".

EZMXY - ARGUMENTS

REAL XDRA(NPTS) or XDRA(IDXY,MANY) or XDRA(IDXY,NPTS), depending on the current value of 'ROW.'; REAL YDRA(IDXY,MANY) or YDRA(IDXY,NPTS), depending on the current value of 'ROW.'; INTEGER IDXY, MANY, NPTS; CHARACTER*(*) GLAB.

XDRA is a one-dimensional or two-dimensional array of curve-point x coordinates. The current value of 'NULL/1.' (default value "1.E36") may be used in XDRA to signal missing points; curve segments on either side of a missing point are not drawn. Note: Because all non-missing coordinates are used in figuring the minimum and maximum user values along a given axis, it is safest to mark both coordinates as "missing".

If 'ROW.' has the absolute value "1." (the default), XDRA is singly-dimensional. It is subscripted by point number.

If 'ROW.' has the absolute value "2." or greater, XDRA is doubly-dimensional. It is subscripted by point number and curve number, in that order if 'ROW.' is positive (the default), in the reverse order if 'ROW.' is negative.

YDRA is a two-dimensional array of curve-point y coordinates. The current value of 'NULL/1.' (default value "1.E36") may be used in YDRA to signal missing points; curve segments on either side of a missing point are not drawn. Note: Because all non-missing coordinates are used in figuring the minimum and maximum user values along a given axis, it is safest to mark both coordinates as "missing".

If 'ROW.' is positive (the default), YDRA is subscripted by point number and curve number, in that order; otherwise, YDRA is subscripted by curve number and point number, in that order.
IDXY is the first dimension of the arrays XDRA (if it is doubly-dimensional) and YDRA (unconditionally), required by EZMXY in order to index these arrays properly.

If 'ROW.' is positive (the default), IDXY must be greater than or equal to NPTS; otherwise, IDXY must be greater than or equal to MANY.

MANY is the number of curves to be drawn by EZMXY.

NPTS is the number of points defining each curve to be drawn by EZMXY.

GLAB is a character expression defining a new "graph label". (If the first character of this expression is "CHAR(0)", no new "graph label" is defined; the current one will continue to be used.) A character string defining a new graph label must either be of the exact length specified by the current value of 'LINE/MAXIMUM.' (default - 40 characters), or shorter; if shorter, it must be terminated by the character defined by the current value of 'LINE/END.' (default - a '$'). The string becomes the new text of line number 100 of the label named 'T'.

ANOTAT(XLAB,YLAB,LBAC,LSET,NDSH,DSHC)

Changes the values of certain primary control parameters, purportedly having to do with "annotation" of a graph.

ANOTAT - USAGE

The subroutine ANOTAT is provided principally for historical reasons. Each of the parameters referenced by its argument list can be set individually by means of the routines AGSETP, AGSETF, and/or AGSETI. In fact, ANOTAT is implemented using calls to these routines.

See example 8, in the section "EXAMPLES".

ANOTAT - ARGUMENTS

INTEGER LBAC, LSET, NDSH; CHARACTER*(*) XLAB, YLAB, DSHC(NDSH).
XLAB is a character expression defining a new "x-axis label". (If the first character of this expression is "CHAR(0)", no new "x-axis label" is defined; the current one will continue to be used.) A character string defining a new x-axis label must either be of the exact length specified by the current value of 'LINE/MAXIMUM.' (default - 40 characters), or shorter; if shorter, it must be terminated by the character defined by the current value of 'LINE/END.' (default - a '$'). The string becomes the new text of line number -100 of the label 'B'.

YLAB is a character expression defining a new "y-axis label". (If the first character of this expression is "CHAR(0)", no new "y-axis label" is defined; the current one will continue to be used.) A character string defining a new y-axis label must either be of the exact length specified by the current value of 'LINE/MAXIMUM.' (default - 40 characters), or shorter; if shorter, it must be terminated by the character defined by the current value of 'LINE/END.' (default - a '$'). The string becomes the new text of line number 100 of the label 'L'.

LBAC, if non-zero, has the integer value 1, 2, 3, or 4, the floating-point equivalent of which is to become the new value of 'BACKGROUND.'. (If LBAC is zero, no change is to be made in the current value.) The value "1" specifies a perimeter background, the value "2" a grid background, the value "3" a half-axis background, and the value "4" no background at all.

See the discussion of 'BACKGROUND.', in the section "PARAMETERS".

LSET, if non-zero, is an integer having the absolute value 1, 2, 3, or 4, the floating-point equivalent of which is to be stored (by means of a call to AGSETI) as the new value of 'SET.'. If LSET is zero, no change is to be made in the current value of 'SET.'.

See the discussion of 'SET.', in the section "PARAMETERS".

NDSH, if zero, specifies that no change is to be made in the parameters which specify the dashed-line patterns to be used for curves. If NDSH is non-zero, it specifies an integer value whose floating-point equivalent is to be stored as the new value of 'DASH/SELECTOR.' (which has the default value "1.").
If NDSH is negative, 'DASH/SELECTOR.' is set negative, forcing EZMY and EZMXY to use internally-defined "alphabetic" patterns for the MANY curves drawn ("A" for the first, "B" for the second, ..., "Z" for the 26th, "A" for the 27th, etc.). The routines EZY and EZXY are unaffected.

If NDSH is greater than zero, it must be less than or equal to 26, and the next argument, DSHC, must contain NDSH dashed-line patterns comprising the new "user" set of patterns. The fact that 'DASH/SELECTOR.' is set positive forces EZMY and EZMXY to use this set of patterns. (The routines EZY and EZXY always use the first pattern in this set.) The contents of the array DSHC are copied to storage local to AUTOGRAPH and pointers to them are installed as the values of 'DASH/PATTERNS/1.', '.../2.', etc.

See the discussion of 'DASH.', in the section "PARAMETERS".

DSHC is meaningful only when NDSH is greater than zero. In this case, it must be an array of NDSH character strings, each of the length specified by the current value of 'DASH/LENGTH.' Each character string represents a dashed-line pattern; dollar signs mean "pen down", quotes mean "pen up", and other characters mean "draw me".

See the discussion of 'DASH.', in the section "PARAMETERS".

DISPLA(LFRA,LROW,LTYP)

Changes the values of certain primary control parameters, purportedly having to do with the "display" of a graph.

DISPLA - USAGE

The subroutine DISPLA is provided principally for historical reasons. Each of the parameters referenced by its argument list can be set individually by means of the routines AGSETP and/or AGSETI. In fact, DISPLA is implemented using calls to these routines.

See the final example, in the section "EXAMPLES".

August 1987
AUTOGRAPH: A Graphing Utility

DISPLA - ARGUMENTS

INTEGER LFRA, LROW, LTYP.

LFRA, if non-zero, has an integer value, the floating-point equivalent of which is to become the new value of 'FRAME.'. If LFRA is zero, no change is to be made in the current value of 'FRAME.'.

The default value of 'FRAME.' is "1.", specifying that each of the routines EZY, EZXY, EZMY, and EBMXY is to do a frame advance after drawing a graph. The value "2." specifies that these routines should do no frame advance, the value "3." that they should do a frame advance before drawing a graph.

LROW, if non-zero, has an integer value, the floating-point equivalent of which is to become the new value of 'ROW.' If LROW is zero, no change is to be made in the current value of 'ROW.'.

This parameter affects the way in which the routines EZMY and EBMXY interpret the arguments XDRA and YDRA, as follows:

If 'ROW.' is positive, the first subscript of YDRA is a point number and the second subscript is a curve number. If 'ROW.' is negative, the order of the subscripts is reversed (row-wise, rather than column-wise, storage).

If the absolute value of 'ROW.' is "1.", XDRA is singly-subscripted; its subscript is a point number. If the absolute value of 'ROW.' is "2." or greater, XDRA is doubly-subscripted; the order of the subscripts is the same as for YDRA.

The default value of 'ROW.' is "1.", specifying that XDRA is singly-subscripted by point number and that YDRA is doubly-subscripted by point number and curve number, in that order.

LTYP, if non-zero, is an integer specifying new values for 'X/LOGARITHMIC.' and 'Y/LOGARITHMIC.' If LTYP is zero, no change is to be made in the current values.

The parameter 'X/LOGARITHMIC.' has the default value "0.", specifying a linear mapping of user x coordinates onto the horizontal axis of the grid window; it may be given either of the two values "-1." or "+1." to specify
a logarithmic mapping. The value "-1." protects it from being reset as a
side effect of setting 'SET.'. DISPLA generates the value "0." or "-1.".

The parameter 'Y/LOGARITHMIC.' is defined similarly and affects the mapping
of user y coordinates onto the vertical axis of the grid window.

A non-zero LTYP resets these values, as follows:

<table>
<thead>
<tr>
<th>LTYP</th>
<th>'X/LOGARITHMIC.'</th>
<th>'Y/LOGARITHMIC.'</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>linear</td>
<td>linear</td>
</tr>
<tr>
<td>2</td>
<td>linear</td>
<td>logarithmic</td>
</tr>
<tr>
<td>3</td>
<td>logarithmic</td>
<td>linear</td>
</tr>
<tr>
<td>4</td>
<td>logarithmic</td>
<td>logarithmic</td>
</tr>
</tbody>
</table>

AGSETP(TPWN,FURA,LURA)

Allows a user program to reset the values of a group of parameters containing one or more elements.

AGSETP - USAGE

The subroutine AGSETP is called with a character string identifying a group of parameters (possibly a single parameter), an array containing new values of those parameters, and the length of the array, as arguments. It transfers the new values into the appropriate locations in the labeled common block AGCONP, thus modifying the effect of subsequent calls to other AUTOGRAPH routines.

If a single parameter is being set, one of the routines AGSETF, AGSETI, or AGSETC (which see, below) may be used instead.

When certain parameters are set individually, AGSETP takes further "special" action. For example, if 'BACKGROUND.' is set, thereby requesting one of the standard types of backgrounds, AGSETP changes a number of other parameters in order to achieve the desired effect. The parameters which imply such special action are as follows:
AUTOGRAPH: A Graphing Utility

'SET.'
'BACKGROUND.'
'NULL/1.' and 'NULL/2.'
'LABEL/CONTROL.'
'LABEL/NAME.'
'LABEL/DEFINITION/SUPPRESSION.'
'LINE/NUMBER.'
'LINE/DEFINITION/SUPPRESSION.'
'LINE/DEFINITION/TEXT.'

See the section "PARAMETERS" for descriptions of the parameters whose values may be set.

AGSETP - ARGUMENTS

CHARACTER*(*) TPGN; REAL FURA(LURA); INTEGER LURA.

TPGN is a character string of the form 'k(1)/k(2)/...k(n).', where each of the k(i)'s is a keyword. The keyword k(1) specifies a group of parameters, k(2) a subgroup of that group, k(3) a subgroup of that subgroup, etc. The whole string is the name of some group of parameters the user wishes to set.

For example, 'AXIS.' is the name of a 92-word group of parameters describing the four axes, 'AXIS/RIGHT.' is the name of a 23-word subgroup describing the right y axis,

'AXIS/RIGHT/INTERSECTION.'

is the name of a 2-word further subgroup describing the intersection of the right y axis with the bottom of the grid window, and

'AXIS/RIGHT/INTERSECTION/USER.'

is the name of a single parameter specifying the point of intersection of the right y axis with the bottom of the grid window as an x coordinate in the user coordinate system.

Obviously, these names can sometimes become rather long. There are various ways in which they may be shortened. First, since the fifth and following...
characters of each keyword are ignored, they may be omitted; this would shorten

'AXIS/RIGHT/INTERSECTION/USER.'

to

'AXIS/RIGH/INTE/USER.'

Even fewer characters may be used, as long as no ambiguity of interpretation arises. To be completely safe, use at least the first three characters of the group keyword and at least the first two characters of each subgroup keyword; this would shorten the example above to 'AXI/RI/IN/US.'.

Moreover, certain group and subgroup keywords may be omitted entirely; for example, 'AXI/RI/IN/US.' may be shortened to 'RI/IN/US.'.

Names may also be lengthened in various ways in order to improve their readability. Blanks may be used as desired on either side of a keyword. Any sequence of characters not including a slash or a period may be inserted after a keyword, separated from it by at least one blank. For example, the name

'DASH PATTERN / CHARACTER WIDTH .'

is equivalent to, and considerably more meaningful than,

'DAS/CH.' (or even 'DASH/CHARACTER. ')

A complete list of the parameters may be found in the section "PARAMETERS", below.

FURA is a user array (of length LURA) containing the new values of the parameters in the group specified by TPGN, in the same order as they appear in the group.

All parameters have floating-point values (because of a portability problem which arose in implementing AGSETF). Those which represent intrinsically integral quantities have a value of the form "FLOAT(n)", where "n" is the integral quantity being represented. Some parameters intrinsically take on character-string values; the floating-point quantity stored as the value of
such a parameter is typically an identifier allowing for later retrieval of the character string from a character storage area inside AUTOGRAPH. The routines AGSETC and AGGETC may be used to set/get the character-string values of such parameters.

LURA is the length of FURA (the number of floating-point elements in it). Its value may be less than, equal to, or greater than, the length of the parameter group specified by TPGN. The number of values transferred from FURA is the minimum of the two (but not less than one). This means that if, for example, you only wish to set the first two parameters of a 100-parameter group, you may do so by using LURA = 2.

AGSETF(TPGN,FUSR)

Allows a user program to store a floating-point number as the value of a single parameter.

AGSETF - USAGE

This subroutine transfers the value of FUSR to a local array FURA, dimensioned 1, and executes the statement

    CALL AGSETP (TPGN,FURA, 1)

See the description of AGSETP, above. The "special actions" described there may result from a call to AGSETF.

AGSETF - ARGUMENTS

CHARACTER*(*) TPGN; REAL FUSR.

TPGN is an parameter identifier, as described for AGSETP, above. If a group of more than one parameter is specified, only the first element of that group will be affected by the call.

FUSR is the floating-point value to be given to the parameter specified by TPGN.
AGSETI (TPGN, IUSR)

Allows a user program to store the floating-point equivalent of an integer as the value of a single parameter.

AGSETI - USAGE

This subroutine stores the value FLOAT(IUSR) in a local array FURA, dimensioned 1, and then executes the statement

    CALL AGSETP(TPGN, FURA, 1)

See the description of AGSETP, above. The "special actions" described there may result from a call to AGSETI.

AGSETI - ARGUMENTS

CHARACTER(*) TPGN; INTEGER IUSR.

TPGN is a parameter identifier, as described for AGSETP, above. If a group of more than one parameter is specified, only the first element of that group will be affected by the call.

IUSR is the integer equivalent of the floating-point value to be given to the parameter specified by TPGN.

AGSETC (TPGN, CUSR)

Allows a user program to (in effect) store a character string as the value of a specified single parameter.

AGSETC - USAGE

This subroutine stores the character string CUSR in an internal string storage space, generates a floating-point identifier allowing for later retrieval of the character string, stores that identifier in a local array.
AUTOGRAPH: A Graphing Utility

FURA, dimensioned 1, and then executes the statement

CALL AGSETP (TPGN,FURA,1)

See the description of AGSETP, above. The "special actions" described there may result from a call to AGSETC.

AGSETC - ARGUMENTS

CHARACTER*(*) TPGN, CUSR.

TPGN is a parameter identifier, as described for AGSETP, above. The specified parameter must be one of those which intrinsically have values of type character: 'LINE/END.', 'LABEL/NAME.', 'LINE/TEXT.', or 'DASH/PATTERN/n.'

CUSR is a desired character string.

If 'LINE/END.' is being set, only the first character of CUSR will be used.

If 'LABEL/NAME.' is being set, the length of CUSR will be taken to be "MAXO(1,LEN(CUSR))".

If the text of a label is being set, CUSR must either be of the exact length specified by 'LINE/MAXIMUM.' (40 characters, by default) or shorter; if shorter, it must be terminated by the character defined by 'LINE/END.' (default - a '$').

If a dash pattern is being set, the length of CUSR will be taken to be the minimum of "LEN(CUSR)" and the value specified by 'DASH/LENGTH.'.

AGGETP(TPGN,FURA,LURA)

Allows a user program to get the values of a group of parameters containing one or more elements.
AGGETP - USAGE

The subroutine AGGETP is called with a character string identifying a group of parameters (possibly a single parameter), an array to receive the values of those parameters, and the length of the array, as arguments. It transfers values from the appropriate locations in the labeled common block AGCONP to the user array.

If a single parameter is being retrieved, one of the routines AGGETF, AGGETI, or AGGETC (which see, below) may be used instead. No "special" action is implied for any single parameter (as is the case for AGSETP).

See the section "PARAMETERS" for descriptions of parameters whose values may be retrieved.

AGGETP - ARGUMENTS

CHARACTER*(*) TPGN; REAL FURA(LURA); INTEGER LURA.

TPGN is a character string of the form 'k(1)/k(2)/...k(n).', where each of the k(i)'s is a keyword. The keyword k(1) specifies a group of parameters, k(2) a subgroup of that group, k(3) a subgroup of that subgroup, etc. The whole string is the name of some group of parameters the user wishes to get.

See the AGSETP argument TPGN, above, for an example and additional comments.

FURA is a user array (of length LURA) into which the floating-point values of the parameters in the group specified by TPGN are to be transferred, in the same order as they appear in the group.

See the AGSETP argument FURA, above, for additional comments.

LURA is the length of FURA. Its value may be less than, equal to, or greater than, the length of the group specified by TPGN. The number of values transferred into FURA is the minimum of the two (but not less than one). You may, for example, get the first two parameters of a 100-parameter group by using LURA = 2.
AGGETF(TPGN,FUSR)

Allows a user program to retrieve the floating-point value of a single parameter.

AGGETF - USAGE

This subroutine executes the statement

    CALL AGGETP (TPGN,FURA,1)

where FURA is a local array, dimensioned 1, and then sets FUSR equal to FURA(1).

See the description of AGGETP, above.

AGGETF - ARGUMENTS

CHARACTER*(*) TPGN; REAL FUSR.

TPGN is a parameter identifier, as described for AGGETP, above. If a group of more than one parameter is specified, only the first element of that group will be retrieved by the call.

FUSR receives the floating-point value of the parameter specified by TPGN.

AGGETI(TPGN,IUSR)

Allows a user program to retrieve the integer equivalent of the floating-point value of a single parameter.

AGGETI - USAGE

This subroutine executes the statement

    CALL AGGETP (TPGN,FURA,1)
where FURA is a local array, dimensioned 1, and then sets IUSR equal to IFIX(FURA(1)).

See the description of AGGETP, above.

AGGETI - ARGUMENTS

CHARACTER*(*) TPGN; INTEGER IUSR.

TPGN is a parameter identifier, as described for AGGETP, above. If a group of more than one parameter is specified, only the first element of that group will be retrieved by the call.

IUSR receives the integer equivalent of the floating-point value of the parameter specified by TPGN.

AGGETC(TPGN,CUSR)

Allows a user program to retrieve (in effect) the character-string values of certain single parameters.

AGGETC - USAGE

This subroutine executes the statement

   CALL AGGETP (TPGN,FURA,1)

where FURA is a local array, dimensioned 1. It then retrieves, from AUTOGRAPH's character storage space, the character string identified by FURA(1), and returns that string as the value of CUSR.

See the description of AGGETP, above.

AGGETC - ARGUMENTS

CHARACTER*(*) TPGN, CUSR.
AUTOGRAPH: A Graphing Utility

TPGN is a parameter identifier, as described for AGGETP, above. The specified parameter must be one of those which intrinsically have values of type character: 'LINE/END.', 'LABEL/NAME.', 'LINE/TEXT.', or 'DASH/PATTERN/n.'

CUSR receives the desired character string.

AGSTUP(XDRA,NVIX,IIVX,NEVX,IIEX,...)

(The remaining arguments are YDRA, NVIY, IIVY, NEVY, and IIEY.) Performs "set-up" tasks required before AGBACK and AGCURV may be called. Basically, AGSTUP examines the current values of the primary control parameters for errors and computes from them and from its arguments the values of secondary control parameters. The primary and secondary control parameters together determine how the routines AGBACK and AGCURV will behave.

AGSTUP - USAGE

The subroutine AGSTUP is normally called once per graph, just prior to the sequence of calls to AGBACK and/or AGCURV which actually draws the graph.

The call to AGSTUP may be omitted only if (1) no primary control parameters have been changed since the last time AGSTUP was called and (2) the position of the grid window and the mapping of user x/y coordinates into the grid window would be unaffected by the AGSTUP call. The routine must be called at least once, for the first graph; for succeeding graphs, if the call can be omitted, it should be, since the routine is rather time-consuming.

Note that each of the routines Ezy, EZXY, EZMY, and EZMXY unconditionally executes a call to AGSTUP (via a routine called AGEZSU) before calling AGBACK and/or AGCURV.

An appropriate call to the plot-package routine SET is executed by AGSTUP.

AGSTUP - ARGUMENTS

REAL XDRA(*),YDRA(*); INTEGER NVIX, IIVX, NEVX, IIEX, NVIY, IIVY, NEVY, IIEY.
The first five arguments of AGSTUP are meaningful only if at least one of 'X/MINIMUM.' and 'X/MAXIMUM.' has the value "null 1" or "null 2", specifying that AUTOGRAPH is to determine for itself the minimum and/or maximum x coordinate in the user's data. Similarly, the second five arguments are meaningful only if at least one of 'Y/MINIMUM.' and 'Y/MAXIMUM.' has the value "null 1" or "null 2".

XDRA is an array of user x coordinates.

NVIX is the number of "vectors" of data from XDRA to be considered in computing the minimum and/or maximum x values.

IIVX is the index increment between two "vectors" in XDRA. The first element of the first vector is XDRA(1), the first element of the second vector is XDRA(1+IIVX), the first element of the third vector is XDRA(1+IIVX*2), etc.

NEVX is the number of elements of each vector in XDRA to be considered in computing the minimum and/or maximum x values.

IIEX is the index increment between two consecutive elements of a vector in XDRA. The second element of the first vector is XDRA(1+IIEX), the third element of the first vector is XDRA(1+IIEX*2), etc.

If IIEX has the value 0, the contents of XDRA are ignored completely; the minimum and maximum x values are considered to be "1." and FLOAT(NEVX), respectively.

YDRA, NVIY, IIVY, NEVY, and IIEX are treated analogously, but define the user y coordinates.

Some examples:
**AUTOGRAPH: A Graphing Utility**

<table>
<thead>
<tr>
<th>X array</th>
<th>Data to be used</th>
<th>XDRA</th>
<th>NVIX</th>
<th>IIVX</th>
<th>NEVX</th>
<th>IIEEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>X(100)</td>
<td>(all data)</td>
<td>X(1)</td>
<td>1</td>
<td>-</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>(X(I),I=1,99,2)</td>
<td>X(1)</td>
<td>1</td>
<td>-</td>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>(X(I),I=51,99,2)</td>
<td>X(51)</td>
<td>1</td>
<td>-</td>
<td>25</td>
<td>2</td>
</tr>
<tr>
<td>X(10,10)</td>
<td>(all data)</td>
<td>X(1,1)</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X(1,1)</td>
<td>1</td>
<td>-</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>((X(I,J),I=1,10),</td>
<td>X(1,1)</td>
<td>6</td>
<td>10</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>J=1,6)</td>
<td>X(1,1)</td>
<td>10</td>
<td>1</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>((X(I,J),I=3,7),</td>
<td>X(3,3)</td>
<td>7</td>
<td>10</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>J=3,9))</td>
<td>X(3,3)</td>
<td>5</td>
<td>1</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>((X(I,J),I=3,7,4),</td>
<td>X(3,3)</td>
<td>3</td>
<td>30</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>J=3,9,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(none)</td>
<td>1., 2., ... 62.</td>
<td>-</td>
<td>-</td>
<td>62</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: The character "-" is used above to indicate an argument which is ignored and may be given a dummy value.

Normally, the x and y coordinate data tendered to AGSTUP is the same data which will later be used to draw curves. However, this need not be the case. For example, one could call AGSTUP with a two-word XDRA (setting NVIX=1, IIVX=1, NEVX=2, and IIEEX=1) containing a desired minimum and maximum to be used, disregarding the real data.

If 'INVERT.' is given the value "1." (in place of its default value "0."), AGSTUP will behave as if its first five arguments had been interchanged with its last five, so that x-coordinate values refer to vertical distances, and y-coordinate values to horizontal distances, on the graph. This parameter affects AGCURV in a similar manner, thus allowing one to plot "x as a function of y".
AGBACK

Draws the background specified by the current values of the control parameters - the primary parameters with default values or with values supplied by the user, and the secondary parameters with values computed by AGSTUP.

AGBACK - USAGE

Just call it. See the section "PARAMETERS" for descriptions of the parameters which affect the appearance of a background drawn by AGBACK. See also the description of AGSTUP, above.

An appropriate call to the plot-package routine SET is executed by AGBACK.

AGBACK - ARGUMENTS

None.

AGCURV(XVEC,IIEX,YVEC,IIEY,NEXY,KDSH)

Draws a curve in a manner specified by the current values of the control parameters - the primary parameters with default values or with values supplied by the user, and the secondary parameters with values computed by AGSTUP.

AGCURV - USAGE

The subroutine AGCURV, given the x and y coordinates of a set of data points, draws the curve defined by those points, using a dashed-line pattern selected by the final argument.

See the section "PARAMETERS" for a description of the parameters which affect the behavior of AGCURV. One parameter of particular interest is 'WINDOW.', which has the default value "0.". If 'WINDOW.' is given the value "1.", any portion of a curve which lies outside the grid window is omitted. No distortion of any curve segment results; the effect is exactly as if the curve were viewed through a window. There is an additional advantage in setting 'WINDOW.' to "1.": if either the x coordinates, or
the y coordinates, or both, are mapped logarithmically into the grid window and zero or negative values occur in the data, AGCURV treats those values as missing-point signals, rather than bombing with an ALOG10 error.

AGCURV - ARGUMENTS

REAL XVEC(*), YVEC(*); INTEGER IIEX, IIEY, NEXY, KDSH

XVEC, when IIEX is non-zero, is a singly-subscripted array containing NEXY x-coordinate data - curve point 1 has x coordinate XVEC(1), curve point 2 has x coordinate XVEC(1+IIEX), curve point 3 has x coordinate XVEC(1+IIEX*2), etc. When IIEX is zero, the array XVEC is ignored - curve point 1 has x coordinate "1.", curve point 2 has x coordinate "2.", etc.

If the value of any x coordinate matches the current value of 'NULL/1.' (default - "1.E36"), the corresponding point is considered to be missing - curve segments on either side of that point are not drawn.

IIEX, if non-zero, is the index increment between one x coordinate in XVEC and the next. If IIEX is zero, the array XVEC is ignored, as described above.

YVEC is just like XVEC, but provides y coordinate data.

IIEY is just like IIEX, but describes the use (or non-use) of YVEC.

NEXY is the number of curve points - the number of x/y coordinate pairs to be used.

Note: If 'INVERT.' is given the value "1." (in place of its default value "0."), AGCURV will behave as if the arguments XVEC and IIEX had been interchanged with the arguments YVEC and IIEY, so that x-coordinate values refer to vertical distances, and y-coordinate values to horizontal distances, on the graph. This parameter affects AGSTUP in a similar manner, thus allowing one to plot "x as a function of y".

KDSH specifies the dashed-line pattern to be used in drawing the curve. (Since the routines DASHD, FRSTD, VECTD, and LASTD, in the package DASHCHAR, are used to draw the curve, it may have its own particular dashed-line pattern.)
If KDSH is zero, the user is assumed to have done his own call to DASHD; AGCURV will do no such call.

If KDSH is non-zero and negative, the function $\text{MOD}(-\text{KDSH} - 1, 26)+1$ determines which of 26 "alphabetic" patterns is to be used; each of these generates a solid line interrupted by one of the letters of the alphabet. The value 1 implies that an "A" will be used, the value 2 that a "B" will be used, ... the value 27 that an "A" will be used again, etc.

If KDSH is non-zero and positive, the function $\text{MOD}($KDSH - 1, n$)+1$ determines which of n "user" patterns is to be used; these n patterns are defined by the parameters in the group named 'DASH.' - the default values specify one solid-line pattern.

Note: The routines EZY and EZXY, which draw one curve per call, always call AGCURV with KDSH = 1. The routines EZMY and EZMXY, which draw one or more curves per call, call AGCURV with KDSH = $\text{ISIGN}(p,q)$, where p is the number of the curve being drawn (p is between 1 and MANY, inclusive) and q is the current integral value of 'DASH/SELECTOR.'.

AGSAVE(IFNO)

Saves the current state of AUTOGRAPH for later restoration by AGRSTR.

AGSAVE - USAGE

Calling AGSAVE saves the current state of AUTOGRAPH (frequently the default state) by writing, on the unit specified by IFNO, the current values of all the parameters and the contents of the character storage space referenced by certain of those parameters.

AGSAVE - ARGUMENTS

INTEGER IFNO.

IFNO is the number of a unit to which a single unformatted record is to be written. It is the user's responsibility to position this unit. AGSAVE does not rewind it, either before or after writing the record.
AGRSTR(IFNO)

Restores a saved state of AUTOGRAPH.

AGRSTR - USAGE

Calling AGRSTR restores AUTOGRAPH to a previously saved state (frequently the default state) by reading, from the unit specified by IFNO, values of all the parameters and the contents of the character storage space referenced by certain of those parameters.

AGRSTR - ARGUMENTS

INTEGER IFNO.

IFNO is the number of a unit from which a single unformatted record is to be read. It is the user's responsibility to position this unit. AGRSTR does not rewind it, either before or after reading the record.

AGBNCH(IDSH)

Provides an easy way to convert binary dash patterns into character dash patterns.

AGBNCH - USAGE

AGBNCH is a function, of type CHARACTER*16, and must be declared as such in a user program referencing it:

CHARACTER*16 AGBNCH

The value of AGBNCH(IDSH), where IDSH is an integer in the range 0 to 65535 (2**16-1) representing a 16-bit binary dash pattern, is the equivalent character dash pattern.
AGBNCH - ARGUMENTS

INTEGER IDSH.

IDSH is an integer in the range 0 to 65535, inclusive.

AGDSHN(IDSH)

Provides an easy way to generate the names of parameters in the group 'DASH/PATTERN.', for use in calls to AGSETC and AGGETC.

AGDSHN - USAGE

AGDSHN is a function, of type CHARACTER*16, and must be declared as such in a user program referencing it:

    CHARACTER*16 AGDSHN

The value of AGDSHN(IDSH), where IDSH is an integer "n" in the range 1 to 26, is the name of the nth dash-pattern parameter - that is to say, it is the character string 'DASH/PATTERN/n.'.

AGDSHN - ARGUMENTS

INTEGER IDSH.

IDSH is an integer in the range 1 to 26, inclusive.

AGUTOL(IAXS,FUNS,IDMA,VINP,VOTP)

Provides a way for the user to change the user-system-to-label-system mapping for one or more of the four axes.
AGUTOL - USAGE

This routine is not called by the user program, but by AUTOGRAPH itself. It defines the user-system-to-label-system mapping for all four axes. The default version makes the label system match the user system on all four axes. The user may supply his own version to change the mapping on one or more of the axes. Mappings defined by the subroutine must be continuous and monotonic.

Note: A user version of AGUTOL should not call any other AUTOGRAPH routine.

AGUTOL - ARGUMENTS

INTEGER IAXS, IDMA; REAL FUNS, VINP, VOTP.

IAXS is the number of the axis. The values 1, 2, 3, and 4 imply the left, right, bottom, and top axes, respectively.

FUNS is the value of 'AXIS/s/FUNCTION.', which may be used to select the desired mapping function for the axis IAXS. It is recommended that the default value (zero) be used to specify the identity mapping. The value may be integral ("1.", "2.", etc.) and serve purely to select the code to be executed or it may be the value of an actual parameter in the equations defining the mapping.

IDMA specifies the direction of the mapping. A value greater than zero indicates that VINP is a value in the user system and that VOTP is to be a value in the label system, a value less than zero the opposite. The mappings in one direction must be the mathematical inverses of the mappings in the other direction or AUTOGRAPH will probably go bananas.

VINP is an input value in one coordinate system.

VOTP is an output value in the other coordinate system.
AGCHAX(IFLG, IAXS, IPRT, VILS)

Provides a way for the user to change the color, intensity, line style, etc., of various portions of the axes.

AGCHAX - USAGE

This routine is not called by the user program, but by AUTOGRAPH itself, just before and just after each of the objects making up an axis is drawn. The default version does nothing. The user may supply his own version to change the color, intensity, line style, etc. of selected portions of the axis.

Note: A user version of AGCHAX should not call any other AUTOGRAPH routine.

AGCHAX - ARGUMENTS

INTEGER IFLG, IAXS, IPRT; REAL VILS.

IFLG is zero if a particular object is about to be drawn, non-zero if it has just been drawn.

IAXS is the number of the axis being drawn. The values 1, 2, 3, and 4 indicate the left, right, bottom, and top axes, respectively.

IPRT indicates the part of the axis being drawn. Possible values are as follows:

1 implies the line of the axis.

2 implies a major tick.

3 implies a minor tick.

4 implies the mantissa of a numeric label.

5 implies the exponent of a numeric label.
VILS is the value in the label system at the point where the part is being
drawn. For IPRT = 1, VILS is zero.

AGCHCU(IFLG,KDSH)

Provides a way for the user to change the color, intensity, line style,
etc., of curves drawn by AUTOGRAPH.

AGCHCU - USAGE

This routine is not called by the user program, but by AUTOGRAPH itself,
just before and just after each curve is drawn. The default version does
nothing. The user may supply his own version to change the color, inten-
sity, line style, etc. of selected curves.

Note: A user version of AGCHCU should not call any other AUTOGRAPH rou-
tine.

AGCHCU - ARGUMENTS

INTEGER IFLG, KDSH.

IFLG is zero if a particular object is about to be drawn, non-zero if it
has just been drawn.

KDSH is the value of AGCURV's last argument, as follows:

<table>
<thead>
<tr>
<th>AGCURV called by</th>
<th>Value of KDSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>EZY</td>
<td>1</td>
</tr>
<tr>
<td>EZXY</td>
<td>1</td>
</tr>
<tr>
<td>EZMY</td>
<td>&quot;n&quot; or &quot;-n&quot;; n is the curve number</td>
</tr>
<tr>
<td>EZMXY</td>
<td>&quot;n&quot; or &quot;-n&quot;; n is the curve number</td>
</tr>
<tr>
<td>the user program</td>
<td>the user value</td>
</tr>
</tbody>
</table>

August 1987
AGCHIL(FIG, LBNM, LNNO)

Provides a way for the user to change the color, intensity, text style, etc., of the informational labels.

AGCHIL - USAGE

This routine is not called by the user program, but by AUTOGRAPH itself, just before and just after each informational-label line is drawn. The default version does nothing. The user may supply his own version to change the appearance of selected lines of text.

Note: A user version of AGCHIL should not call any other AUTOGRAPH routine.

AGCHIL - ARGUMENTS

INTEGER IFIG, LNNO; CHARACTER(*) LBNM.

IFLG is zero if a particular object is about to be drawn, non-zero if it has just been drawn.

LBNM is a character variable containing the name of the label being drawn.

LNNO is the number of the line being drawn.

AGCHNL(IAXS, VILS, CHRM, MCIM, NCIM, ...)

(The remaining arguments are IPXM, CHRE, MCIE, and NCIE.) Provides a way for the user to substitute arbitrary character strings for the numeric labels generated by AUTOGRAPH.

AGCHNL - USAGE

This routine is not called by the user program, but by AUTOGRAPH itself, just after the character string representing each numeric label has been generated and just before it is written on the graph. The user may change the character string in any desired way. Axes may thereby be labeled using...
the names of the months, Roman numerals, etc.

Note: A user version of AGCHNL should not call any other AUTOGRAPH routine.

AGCHNL - ARGUMENTS

INTEGER IAXS, MCIM, NCIM, IPXM, MCIE, NCIE; REAL VILS; CHARACTER*(*) CHRM, CHRE.

IAXS is the number of the axis being drawn. The values 1, 2, 3, and 4 imply the left, right, bottom, and top axes, respectively.

VILS is the value to be represented by the numeric label, in the label system for the axis. The value of VILS must not be altered.

CHRM, on entry, is a character string containing the mantissa of the numeric label, as it will appear if AGCHNL makes no changes. If the numeric label includes a "times" symbol, it is represented by a blank in CHRM. (See IPXM, below.) CHRM may be modified.

MCIM is the length of CHRM - the maximum number of characters that it will hold. The value of MCIM must not be altered.

NCIM, on entry, is the number of meaningful characters in CHRM. If CHRM is changed, NCIM should be changed accordingly.

IPXM, on entry, is zero if there is no "times" symbol in CHRM; if it is non-zero, it is the index of a character position in CHRM. If AGCHNL changes the position of the "times" symbol in CHRM, removes it, or adds it, the value of IPXM must be changed.

CHRE, on entry, is a character string containing the exponent of the numeric label, as it will appear if AGCHNL makes no changes. CHRE may be modified.

MCIE is the length of CHRE - the maximum number of characters that it will hold. The value of MCIE must not be altered.
Routines

NCIE, on entry, is the number of meaningful characters in CHRE. If CHRE is changed, NCIE should be changed accordingly.

AGPWRT(XPOS,YPOS,CHRS,NCHS,ISIZ,IORI,ICEN)

Provides a way for the user to change the style of all text strings drawn by AUTOGRAPH.

AGPWRT - USAGE

This routine is not called by the user program, but by AUTOGRAPH itself, to draw a numeric label or an informational label on the graph. The default version just calls the plot-package routine PWRIT. The user may supply a replacement version. For example, the XLIB file AGUPWRITX contains a version of AGPWRT which calls PWRITX, which produces much nicer-looking labeling.

Note: A user version of AGPWRT should not call any other AUTOGRAPH routine.

AGPWRT - ARGUMENTS

REAL XPOS, YPOS; CHARACTER(*) CHRS; INTEGER NCHS, ISIZ, IORI, ICEN.

XPOS is the x coordinate of a point relative to which the text string is to be positioned.

YPOS is the y coordinate of a point relative to which the text string is to be positioned.

CHRS is the text string to be drawn.

NCHS is the length of CHRS - the number of characters in the text string.

ISIZ specifies the size of the characters to be used. The values 1, 2, 3, and 4 imply character widths of 8, 12, 16, and 24 plotter units, respectively. Larger values specify the character width directly.
AUTOGRAPH: A Graphing Utility

IORI is the orientation angle of the text string, measured in degrees counter-clockwise from a vector which is horizontal and pointing to the right.

ICEN is the centering option. The value "-1" implies that the text is to be written with (XPOS,YPOS) in the center of the left edge of the leftmost character, the value "0" that (XPOS,YPOS) is to be in the center of the entire string, and the value "+1" that (XPOS,YPOS) is to be in the center of the right edge of the rightmost character.
Section III: Parameters
The AUTOGRAPH control parameters reside in the labeled common block AGCONP. There are currently 485 of them, of which 336 are "primary" and 149 are "secondary". Primary control parameters have default values and are subject to change by a user program to produce some desired effect on the behavior of AUTOGRAPH and/or on the nature of a graph being drawn. Secondary control parameters are computed by AUTOGRAPH itself and are not normally subject to change by a user program.

User access to these parameters is provided by the routines AGSETP, AGSETF, AGSETI, AGSETC, AGGETP, AGGETF, AGGETI, and AGGETC (all of which are described in the section "ROUTINES", above). The first argument in a call to one of these routines is a character string naming a group of parameters (perhaps containing only a single parameter) which the user wishes to "set" or "get". Each such string has the form

'k(1)/k(2)/...k(n).'

where k(1) is a keyword identifying a major group of parameters, k(2) is a keyword identifying a subgroup of that major group, k(3) is a keyword identifying a further subgroup of that subgroup, and so on. Only the first three characters of k(1) and the first two characters of the others need be used; also, certain keywords may be omitted.

Because of certain portability considerations, all of the parameters have floating-point values. The routine AGSETP may be used to set the floating-point values of the parameters in any group, the routine AGGETP to retrieve those values. The routine AGSETF (AGGETF) may be used to set (get) the floating-point value of any single parameter.

Some parameters may only take on discrete integral values (like "0.", "1.", "-6.", or "65535.") and are used in roles for which integers would normally be used. The routine AGSETI (AGGETI) may be used to set (get) the integer value of any single parameter of this type.

Other parameters intrinsically represent character strings; the floating-point value of the parameter is an identifier, generated when the character string is passed to AUTOGRAPH and enabling the character string to be retrieved from AUTOGRAPH's character storage space when it is needed. The
routine AGSETC (AGGETC) must be used to set (get) the character-string value associated with any single parameter of this type.

Many parameters have a limited range of acceptable values. What generally happens when a parameter is given an out-of-range value is that AUTOGRAPH (usually the routine AGSTUP) resets that value to the value at the nearer end of the acceptable range.

Setting certain parameters (individually, rather than as part of a multi-parameter group) implies, as a side effect, "special action" by the routine AGSETP (which may be called directly by the user or indirectly by way of a user call to AGSETP, AGSETI, or AGSETC). For example, setting the parameter 'BACKGROUND.' to request a particular background type causes a number of other parameters to be changed in order to achieve the desired result.

Each of the named parameter groups is described below. Square brackets are used to mark portions of a name which may be omitted; the notation

'k(1)/k(2)/...[k(i)/]...k(n).'

indicates that the keyword k(i) and the following slash may be omitted. In each description, the simplest form of the name is given. If a multi-parameter group is named, its subgroups are listed, in the order in which they occur in the group. If a single parameter is named, the default value of that parameter is given and any "special action" by AGSETP is described.

'PRIMARY.'

Simplest form of name: 'PRI.'

This group consists of all 336 primary control parameters, in the order in which they appear below. It was originally provided to give users the capability of saving and restoring the state of AUTOGRAPH. The routines AGSAVE and AGRSTR (which see) should now be used for this purpose.
'FRAME.'

Simplest form of name: 'FRA.'

An integral floating-point number specifying when a frame advance is to be done by the routines EZY, EZXY, EZMY, and EZMXY and having one of three possible values:

 fecha El valor "1." especifica un avance de marco después de dibujar un gráfico.
 fecha El valor "2." especifica no avance de marco en absoluto.
 fecha El valor "3." especifica un avance de marco antes de dibujar un gráfico.

Default value: "1." (frame advance after drawing graph).

'SET.'

Simplest form of name: 'SET.'

An integral floating-point number specifying whether or not the arguments of the last call to "SET" are used to determine the linear/log nature of the current graph, the position of the grid window and/or the x/y minimum/maximum values.

(Note: The routine SET is part of the system plot package. Its first four arguments specify a portion of the plotter frame, its next four arguments specify the minimum and maximum x and y coordinate values to be mapped to that portion, and its ninth argument specifies the linear/log nature of the mapping. The routine GETSET, which is also a part of the system plot package, is used to retrieve the arguments of the last call to SET.)

Giving 'SET.' a value (individually, rather than as part of a group) has both an immediate effect and a delayed effect. The immediate effect, which occurs in the routine AGSETP, is to return most of the parameters in the groups 'GRID.', 'X.', and 'Y.' to their default values. (Exceptions are 'X/LOGARITHMIC.' and 'Y/LOGARITHMIC.', which may have values making them immune to such resetting.) The delayed effect, which occurs in the routine
AGSTUP, depends on the value given to 'SET.'.

There are eight acceptable values of 'SET.', four of which are just the negatives of the other four. Using a negated value suppresses the drawing of curves by the routines EZY, EZXY, EZMY, and EZMXY. Acceptable absolute values of 'SET.' are as follows:

- The value "1." means that the arguments of the last SET call are not to be used by AGSTUP.

- The value "2." means that 'X/LOGARITHMIC.' and 'Y/LOGARITHMIC.' are to be given values ("0." or "-1.") consistent with the ninth argument of the last SET call and that parameters in the group 'GRID.' are to be given values consistent with the first four arguments of the last SET call.

- The value "3." means that 'X/LOGARITHMIC.' and 'Y/LOGARITHMIC.' are to be given values ("0." or "-1.") consistent with the ninth argument of the last SET call and that the other parameters in the groups 'X.' and 'Y.' are to be given values consistent with the fifth through eighth arguments of the last SET call.

- The value "4." means a combination of the actions specified by the values "2." and "3."

Default value: "1." (no arguments of last SET call used).

Special action by AGSETP: As described above, if 'SET.' is set (individually, rather than as part of a group) to any value by an AGSETP call, the parameters in the groups 'GRID.', 'X.', and 'Y.' are reset to their default values. The parameter 'X/LOGARITHMIC.' is reset to its default value ("0.") only if it has the value "+1."; a value of "-1." is not changed; 'Y/LOGARITHMIC.' is treated similarly.

'SROW.'

Simplest form of name: 'ROW.'
Parameters

An integral floating-point number specifying the assumed dimensioning of x and y coordinate data arrays used in calls to the routines EZMY and EZMXY. There are four possibilities:

- The value "-2." means that both x and y arrays are subscripted by curve number and point number, in that order.

- The value "-1." means that y arrays are subscripted by curve number and point number, in that order, but that x arrays are subscripted by point number only. (The same x-coordinate data is used for all the curves.)

- Either of the values "0." or "1." means that y arrays are subscripted by point number and curve number, in that order, but that x arrays are subscripted by point number only. (The same x-coordinate data is used for all of the curves.)

- The value "+2." means that both x and y arrays are subscripted by point number and curve number, in that order.

Default value: "1." (y by point and curve numbers, x by point number only).

'INVERT.'

Simplest form of name: 'INV.'

An integral floating-point number having the value "0." or "1."; giving it the value "1." causes the routines AGSTUP and AGCURV to behave as if arguments defining x-coordinate data had been interchanged with arguments defining y-coordinate data, thus, in some sense, allowing one to graph "x as a function of y". This parameter is principally intended for users of the routines EZY, EZXY, EZMY, and EZMXY.

Default value: "0." (no inversion of x and y arguments).
AUTOGRAPH: A Graphing Utility

'WINDOW.'

Simplest form of name: 'WIN.'

An integral floating-point number having the value "0." or "1."; giving it the value "1." causes the routine AGCURV to use the subroutine AGQURV, rather than AGKURV, for drawing curves. The result is that curve portions falling outside the grid window are omitted. See the description of the routine AGCURV, above.

Default value: "0." (no windowing of curves).

'BARRAY.'

Simplest form of name: 'BAC.'

An integral floating-point number specifying the type of background to be drawn by AGBACK. There are four acceptable values:

- The value "1." specifies a "perimeter" background.
- The value "2." specifies a "grid" background.
- The value "3." specifies a "half-axis" background.
- The value "4." specifies no background at all.

Default value: "1." (a "perimeter" background).

Special action by AGSETP: If 'BACKGROUND.' is set (individually, rather than as part of a group) by a call to AGSETP, the desired background is created by changing the following parameters:

'[AXIS/]s/CONTROL.'

'[AXIS/]s/[TICKS/]MAJOR/[LENGTH/]INWARD.'

'[AXIS/]s/[TICKS/]MINOR/[LENGTH/]INWARD.'

'LABEL/CONTROL.'
where "s" stands for "LEFT", "RIGHT", "BOTTOM", and "TOP". This determines which of the axes are plotted, how long the inward-pointing portions of major and minor tick marks are to be, and whether or not informational labels are to be plotted. Values used are as follows:

The value "1." (perimeter background) sets:

's/CONTROL.' to "4." for all s;
's/MAJOR/INWARD.' to ".015" for all s;
's/MINOR/INWARD.' to ".010" for all s;
'LABEL/CONTROL.' to "2.".

The value "2." (grid background) sets:

's/CONTROL.' to "4." for "s" = "LEFT" and "BOTTOM'',
's/CONTROL.' to "-1." for "s" = "RIGHT" and "TOP'';
's/MAJOR/INWARD.' to "1." for all s;
's/MINOR/INWARD.' to "1." for all s;
'LABEL/CONTROL.' to "2.".

The value "3." (half-axis background) sets:

's/CONTROL.' to "4." for "s" = "LEFT" and "BOTTOM'',
's/CONTROL.' to "-1." for "s" = "RIGHT" and "TOP'';
's/MAJOR/INWARD.' to ".015" for all s;
's/MINOR/INWARD.' to ".010" for all s;
'LABEL/CONTROL.' to "2.".

The value "4." (no background) sets:

's/CONTROL.' to "0." for all s;
's/MAJOR/INWARD.' to ".015" for all s;
's/MINOR/INWARD.' to ".010" for all s;
'LABEL/CONTROL.' to "0.".

The default values of these thirteen parameters correspond to the default value of 'BACKGROUND.' Note that, if they are changed directly, the value of 'BACKGROUND.' may not reflect the actual nature of the background defined by them.
'NULL.'

Simplest form of name: 'NUL.'

This group contains the two "nulls" (or "special values") 'NULL/1.' and 'NULL/2.'.

'NULL/1.'

Simplest form of name: 'NUL/1.'

A floating-point number "null 1", used in the following ways by AUTOGRAPH:

- Certain parameters have by default, or may be given, the value "null 1", specifying that the routine AGSTUP is to choose values for them. The value chosen for a given parameter is not back-stored in place of the "null 1"; thus, a unique value will be chosen for each graph drawn.

- If a curve point specified by the user has x and/or y coordinates equal to "null 1", that curve point is ignored. It is not used in computing minimum and maximum values. Curve segments on either side of it are not drawn.

Default value: "1.E36" (an arbitrary value).

Special action by AGSETP: If 'NULL/1.' is changed (individually, rather than as part of a group) by an AGSETP call, the entire list of primary parameters is scanned - any value equal to the old "null 1" is replaced by the new one.
Parameters

'NULL/2.'

Simplest form of name: 'NUL/2.'

A floating-point number "null 2". Certain parameters may be given the value "null 2", specifying that the routine AGSTUP is to choose values for them. The value chosen for a given parameter is back-stored in place of the "null 2"; thus, a unique value may be chosen for the first graph of a series and then used for all remaining graphs in the series.

Default value: "2.E36" (an arbitrary value).

Special action by AGSETP: If 'NULL/2.' is changed (individually, rather than as part of a group) by an AGSETP call, the entire list of primary parameters is scanned - any value equal to the old "null 2" is replaced by the new one.

'GRAPH.'

Simplest form of name: 'GRA.'

A group of four parameters describing the position of the "graph window" within the plotter frame. A graph drawn by AUTOGRAPH (including labels) is forced to lie entirely within this window. Subgroups and the number of parameters in each are as follows:

'GRAPH/LEFT.' (1)
'GRAPH/RIGHT.' (1)
'GRAPH/BOTTOM.' (1)
'GRAPH/TOP.' (1)

'GRAPH/LEFT.'

Simplest form of name: 'GRA/LE.'
AUTOGRAPH: A Graphing Utility

A floating-point number between "0." and "1." specifying the position of the left edge of the graph window as a fraction of the distance from the left edge to the right edge of the plotter frame.

Default value: "0." (left edge of plotter frame).

'GRAPH/RIGHT.'

Simplest form of name: 'GRA/RI.'

A floating-point number between "0." and "1." specifying the position of the right edge of the graph window as a fraction of the distance from the left edge to the right edge of the plotter frame.

Default value: "1." (right edge of plotter frame).

'GRAPH/BOTTOM.'

Simplest form of name: 'GRA/BO.'

A floating-point number between "0." and "1." specifying the position of the bottom edge of the graph window as a fraction of the distance from the bottom edge to the top edge of the plotter frame.

Default value: "0." (bottom edge of plotter frame).

'GRAPH/TOP.'

Simplest form of name: 'GRA/TO.'

A floating-point number between "0." and "1." specifying the position of the top edge of the graph window as a fraction of the distance from the bottom edge to the top edge of the plotter frame.
Parameters

Default value: "0." (top edge of plotter frame).

'GRID.'

Simplest form of name: 'GRI.'

A group of five parameters describing the position and shape of the "grid window" within the graph window. Subgroups and the number of parameters in each are as follows:

'GRID/LEFT.' (1)
'GRID/RIGHT.' (1)
'GRID/BOTTOM.' (1)
'GRID/TOP.' (1)
'GRID/SHAPE.' (1)

'GRID/LEFT.'

Simplest form of name: 'GRI/LE.'

A floating-point number between "0." and "1." specifying the position of the left edge of the area in which the grid window is to be placed, stated as a fraction of the distance from the left edge to the right edge of the graph window.

Default value: ".15".

'GRID/RIGHT.'

Simplest form of name: 'GRI/RI.'

A floating-point number between "0." and "1." specifying the position of the right edge of the area in which the grid window is to be placed, stated...
AUTOGRAPH: A Graphing Utility

as a fraction of the distance from the left edge to the right edge of the graph window.

Default value: ".95".

'GRID/BOTTOM.'

Simplest form of name: 'GRI/BO.'

A floating-point number between "0." and "1." specifying the position of the bottom edge of the area in which the grid window is to be placed, stated as a fraction of the distance from the bottom edge to the top edge of the graph window.

Default value: ".15".

'GRID/TOP.'

Simplest form of name: 'GRI/TO.'

A floating-point number between "0." and "1." specifying the position of the top edge of the area in which the grid window is to be placed, stated as a fraction of the distance from the bottom edge to the top edge of the graph window.

Default value: ".95".

'GRID/SHAPE.'

Simplest form of name: 'GRI/SH.'

A floating-point number specifying the shape of the grid window. The grid window, whatever its shape, is centered in, and made as large as possible in, the area specified by the first four parameters in the group 'GRID.'.
The value of 'GRID/SHAPE.' falls in one of four possible ranges, as follows:

- A value less than "0." specifies the negative of the desired ratio of the grid window's width to its height. For example, the value "-2." specifies a grid window which is twice as wide as it is high.

- The value "0." specifies a grid window of exactly the same shape as the area specified by the first four parameters in the group 'GRID.' The grid window therefore fills that area completely.

- A value "s" between "0." and "1." specifies a grid window whose shape is determined by the range of the user's coordinate data, reverting to the shape of the area specified by the first four arguments in the group 'GRID.' if the ratio of the shorter side of the grid window to the longer side of the grid window would thereby be made less than "s". For example, if "s" were given the value ".5" and the user x coordinate data ranged in value from "0." to "10." and the user y coordinate data ranged in value from "0." to "15.", the grid window would be made two-thirds as wide as it was high; however, if the y coordinate data ranged in value from "0." to "100.", the grid window would not be made one-tenth as wide as it is high, but would instead be made to fill the entire area specified by the first four arguments of the group 'GRID.'.

- A value "s" greater than or equal to "1." specifies a grid window whose shape is determined by the range of the user's coordinate data, reverting to a square if the ratio of the longer side of the grid window to the shorter side of the grid window would thereby be made greater than "s".

Note that, if 'GRID/SHAPE.' is given a value greater than "0.", AUTOGRAPH assumes that the user's x and y coordinate data have the same units (both in inches, for example) and that the outline of a real two-dimensional object is to be graphed without distortion. The grid window is shaped in such a way as to accomplish this. This feature should not be used when either 'X/LOGARITHMIC.' or 'Y/LOGARITHMIC.' has a non-zero value; doing so will yield strange results.
Note that either of the values "-1." or "+1." produces a square and that the value "-1.61803398874989" produces a golden rectangle.

Default value: "0."

'X.'

Simplest form of name: 'X.'

A group of seven parameters specifying the mapping of the user's x-coordinate data onto the horizontal axis of the grid window. Subgroups and the number of parameters in each are as follows:

'X/MINIMUM.' (1)
'X/MAXIMUM.' (1)
'X/LOGARITHMIC.' (1)
'X/ORDER.' (1)
'X/NICE.' (1)
'X/SMALLEST.' (1)
'X/LARGEST.' (1)

See also 'SET.' and 'INVERT.', above.

'X/MINIMUM.'

Simplest form of name: 'X/MI.'

A floating-point number specifying the minimum user x coordinate to be considered. This parameter normally has the value "null 1", specifying that the routine AGSTUP should examine the user's x-coordinate data and find the minimum value for itself.

If the value "null 2" is used, it will be replaced, the next time AGSTUP is called, by an actual minimum value computed by AGSTUP.
Parameters

If a non-null value is used, AGSTUP will not examine the user's x-coordinate data; the given value will be considered to be the minimum.

If both 'X/MINIMUM.' and 'X/MAXIMUM.' are given non-null values, the former should have a lesser value than the latter.

Default value: "1.E36" ("null 1").

'X/MAXIMUM.'

Simplest form of name: 'X/MA.'

Analogous to 'X/MINIMUM.', above; it specifies the way in which the maximum x coordinate is to be determined.

Default value: "1.E36" ("null 1").

'X/LOGARITHMIC.'

Simplest form of name: 'X/LO.'

An integral floating-point number having one of the values "-1.", "0.", or "+1.".

- The value "0." specifies that the mapping of user x coordinates onto the horizontal axis of the grid window is to be linear.

- The values "-1." and "+1." specify that the mapping is to be logarithmic, in which case all user x-coordinate data must be greater than zero. The value "-1." is immune to change when 'SET.' (which see, above) is reset; the value "+1." is not.

Default value: "0." (linear x mapping).

August 1987
AUTOGRAPH: A Graphing Utility

'X/ORDER.'

Simplest form of name: 'X/OR.'

An integral floating-point number having one of the values "0." or "1."

- The value "0." specifies that the values of user x coordinates mapped to the horizontal axis of the grid window should increase from left to right.

- The value "1." specifies that user x coordinates should decrease from left to right.

Default value: "0." (increase from left to right).

'X/NICE.'

Simplest form of name: 'X/NI.'

An integral floating-point number having one of the values "-1.", "0.", or "+1."

- The value "-1." specifies that user x-coordinate data are to be mapped onto the horizontal axis of the grid window in such a way as to force major-tick positions at the endpoints of the bottom x axis.

- The value "+1." specifies that user x-coordinate data are to be mapped onto the horizontal axis of the grid window in such a way as to force major-tick positions at the endpoints of the top x axis.

- The value "0." specifies that the x-coordinate data are to be mapped so as to range from the left edge of the grid window to the right edge of the grid window; major-tick positions are not forced at the ends of either x axis.

Default value: "-1." (bottom axis "nice").
'X/SMALLEST.'

Simplest form of name: 'X/SM.'

This parameter comes into play when AGSTUP is called upon to compute the minimum x coordinate (when 'X/MINIMUM.' has a null value); if the value of 'X/SMALLEST.' is non-null, values less than it will not be considered in the computation.

Default value: "1.E36" ("null 1").

'X/LARGEST.'

Simplest form of name: 'X/LA.'

This parameter comes into play when AGSTUP is called upon to compute the maximum x coordinate (when 'X/MAXIMUM.' has a null value); if the value of 'X/LARGEST.' is non-null, values greater than it will not be considered in the computation.

Default value: "1.E36" ("null 1").

'Y.'

Simplest form of name: 'Y.'

A group of seven parameters specifying the mapping of the user's y-coordinate data onto the vertical axis of the grid window. Subgroups and the number of parameters in each are as follows:
'Y/MINIMUM.'  (1)
'Y/MAXIMUM.'  (1)
'Y/LOGARITHMIC.'  (1)
'Y/ORDER.'  (1)
'Y/NICE.'  (1)
'Y/SMALLEST.'  (1)
'Y/LARGEST.'  (1)

See also 'SET.' and 'INVERT.', above.

'Y/MINIMUM.'

Simplest form of name: 'Y/MI.'

Analogous to 'X/MINIMUM.', above; it specifies the way in which the minimum y coordinate is to be determined.

Default value: "1.E36" ("null 1").

'Y/MAXIMUM.'

Simplest form of name: 'Y/MA.'

Analogous to 'X/MAXIMUM.', above; it specifies the way in which the maximum y coordinate is to be determined.

Default value: "1.E36" ("null 1").

'Y/LOGARITHMIC.'

Simplest form of name: 'Y/LO.'

Analogous to 'X/LOGARITHMIC.', above; it specifies whether the mapping of y coordinates is linear or logarithmic.
Default value: "0." (linear y).

'Y/ORDER.'

Simplest form of name: 'Y/OR.'

Analogous to 'X/ORDER.', above; it specifies whether y-coordinates increase or decrease from bottom to top.

Default value: "0." (increase from bottom to top).

'Y/NICE.'

Simplest form of name: 'Y/NI.'

Analogous to 'X/NICE.', above; it specifies whether the left y axis, the right y axis, or neither, is to be "nice".

Default value: "-1." (left axis "nice").

'Y/SMALLEST.'

Simplest form of name: 'Y/SM.'

Analogous to 'X/SMALLEST.', above; comes into play when AGSTUP is called upon to compute the minimum y coordinate.

Default value: "1.E36" ("null 1").
AUTOGRAPH: A Graphing Utility

'Y/LARGEST.'

Simplest form of name: 'Y/LA.'

Analogous to 'X/LARGEST.', above; comes into play when AGSTUP is called upon to compute the maximum y coordinate.

Default value: "1.E36" ("null 1").

'AXIS.'

Simplest form of name: 'AXI.'

A group of 92 parameters describing four axes: the left axis, the right axis, the bottom axis, and the top axis. Subgroups and the number of parameters in each are as follows:

' [AXIS/]LEFT.' (23)
' [AXIS/]RIGHT.' (23)
' [AXIS/]BOTTOM.' (23)
' [AXIS/]TOP.' (23)

The elements of the subgroups are interleaved in the group; that is to say, the first elements of the four subgroups constitute elements 1 through 4 of the group, the second elements of the four subgroups constitute elements 5 through 8 of the group, and so on.

'[AXIS/]s.'

(where "s" means "any one of the keywords LEFT, RIGHT, BOTTOM, or TOP").

Simplest form of name: 's.'

A group of 23 parameters describing the axis specified by "s". Subgroups and the number of parameters in each are as follows:
Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>[AXIS/]/s/CONTROL</td>
<td>(1)</td>
</tr>
<tr>
<td>[AXIS/]/s/LINE</td>
<td>(1)</td>
</tr>
<tr>
<td>[AXIS/]/s/INTERSECTION</td>
<td>(2)</td>
</tr>
<tr>
<td>[AXIS/]/s/FUNCTION</td>
<td>(1)</td>
</tr>
<tr>
<td>[AXIS/]/s/TICKS</td>
<td>(10)</td>
</tr>
<tr>
<td>[AXIS/]/s/NUMERIC</td>
<td>(8)</td>
</tr>
</tbody>
</table>

[AXIS/]/s/CONTROL

Simplest form of name: 's/CO.'

An integral floating-point number having one of the values "-1.", "0.", "1.", "2.", "3.", or "4." and controlling certain aspects of the drawing of the axis specified by "s", as follows:

- The value "-1." specifies that only the line portion of the axis may be drawn; tick marks and numeric labels are suppressed.
- The value "0." specifies that no portion of the axis may be drawn.
- A positive value specifies that all portions of the axis may be drawn and specifies what actions AUTOGRAPH may take to prevent numeric-label overlap problems, as follows:
  - The value "1." specifies that numeric labels may not be shrunk or rotated.
  - The value "2." specifies that numeric labels may be shrunk, but not rotated.
  - The value "3." specifies that numeric labels may be rotated, but not shrunk.
  - The value "4." specifies that numeric labels may be both shrunk and/or rotated.
AUTOGRAPH: A Graphing Utility

Default value: "4." for all "s" (all axes drawn, numeric labels may be shrunk and/or rotated).

\\[
[A\text{X}/]s/\text{LINE}.
\\]

Simplest form of name: 's/LI.'

An integral floating-point number having one of the values "0." or "1."

- The value "0." specifies that the line portion of the axis specified by "s" may be drawn.

- The value "1." suppresses the line portion of the axis specified by "s".

Default value: "0." for all "s" (line portions of all axes may be drawn).

\\[
[A\text{X}/]s/\text{INTERSECTION}.
\\]

Simplest form of name: 's/IN.'

A group of two parameters

\\[
[A\text{X}/]s/\text{INTERSECTION/GRID}.
[A\text{X}/]s/\text{INTERSECTION/USER}.
\\]

each having the default value "1.E36" ("null 1"). Giving either of them a non-null value causes the axis specified by "s" to be moved away from its normal position on one edge of the grid window. If both are given non-null values, '.../USER.' takes precedence over '.../GRID.'.

If the left y axis or the right y axis is moved, it remains vertical, but intersects the bottom of the grid window at a specified x coordinate. Similarly, if the bottom x axis or the top x axis is moved, it remains horizontal, but intersects the left edge of the grid at a specified y coordinate.
Parameters

No axis may be moved outside the current graph window; if an attempt is made to do so, the axis is moved as far as the edge and no farther.

\[
[AXIS/] s/INTERSECTION/GRID. \\
\text{Simplest form of name: 's/IN/GR.'}
\]

A floating-point number which, if not equal to the current "null 1", specifies, in the grid coordinate system, the x coordinate (if "s" = "LEFT" or "RIGHT") or the y coordinate (if "s" = "BOTTOM" or "TOP") of the point of intersection of the axis specified by "s" with the perpendicular sides of the grid window.

Default value: "1.E36" ("null 1") for all "s" (axes lie on the edges of the grid window).

\[
[AXIS/] s/INTERSECTION/USER. \\
\text{Simplest form of name: 's/IN/US.'}
\]

A floating-point number which, if not equal to the current "null 1", specifies, in the user coordinate system, the x coordinate (if "s" = "LEFT" or "RIGHT") or the y coordinate (if "s" = "BOTTOM" or "TOP") of the point of intersection of the axis specified by "s" with the perpendicular sides of the grid window.

Default value: "1.E36" ("null 1") for all "s" (axes lie on the edges of the grid window).

\[
[AXIS/] s/FUNCTION. \\
\text{Simplest form of name: 's/FU.'}
\]
A floating-point number, passed as an argument to the subroutine AGUTOL; this subroutine defines the user-system-to-label-system mappings, and thus the label coordinate systems, for all the axes. The default version of AGUTOL defines the identity mapping for all axes; a user version may be substituted to define any desired set of mappings. It is intended that 'AXIS/s/FUNCTION.' be used within a user version of AGUTOL as a function selector. It is further recommended that the value "0." select the identity mapping, thus providing a way to re-create the default situation.

Tick marks on the axis specified by "s" are positioned in the label coordinate system. Numeric labels on the axis give values in the label coordinate system.

See the description of AGUTOL in the section "ROUTINES"; see also example 7.

Default value: "0." for all "s" (identity mapping for all axes).

' [AXIS/]s/TICKS.'

Simplest form of name: 's/TI.'

A group of ten parameters describing the tick marks, if any, which are to be a part of the axis specified by "s". Subgroups and the number of parameters in each are as follows:

' [AXIS/]s/[TICKS/]MAJOR.' (6)
' [AXIS/]s/[TICKS/]MINOR.' (4)

' [AXIS/]s/[TICKS/]MAJOR.'

Simplest form of name: 's/MA.'

A group of six parameters describing the major tick marks, if any, which are to be a part of the axis specified by "s". Subgroups and the number of parameters in each are as follows:
parameters in each are as follows:

'\[AXIS/\]s/[TICKS/\]MAJOR/SPACING.' \(3\) 
'\[AXIS/\]s/[TICKS/\]MAJOR/PATTERN.' \(1\) 
'\[AXIS/\]s/[TICKS/\]MAJOR/LENGTH.' \(2\)

'\[AXIS/\]s/[TICKS/\]MAJOR/SPACING.'

Simplest form of name: 's/MA/SP.'

A group of three parameters describing the way in which major tick marks, if any, are to be spaced along the axis specified by "s". Subgroups and the number of parameters in each are as follows:

'\[AXIS/\]s/[TICKS/\]MAJOR/[SPACING/\]TYPE.' \(1\)
'\[AXIS/\]s/[TICKS/\]MAJOR/[SPACING/\]BASE.' \(1\)
'\[AXIS/\]s/[TICKS/\]MAJOR/[SPACING/\]COUNT.' \(1\)

'\[AXIS/\]s/[TICKS/\]MAJOR/[SPACING/\]TYPE.'

Simplest form of name: 's/MA/TY.'

A floating-point number specifying where major tick marks are to be placed along the axis specified by "s" (that is to say, at what values in the label coordinate system along that axis). Let "b" represent the value of the parameter '.../BASE.' (described next) and "k" represent an arbitrary integer. Then, there are six acceptable values of '.../TYPE.:'

- The value "0." specifies that no major tick marks are to be drawn on the axis.
- The value "1." specifies major tick marks at values of the form plus or minus b times k.
The value "2." specifies major tick marks at values of the form plus or minus b times 10 to the power k.

The value "3." specifies major tick marks at values of the form plus or minus b to the power k.

The value "null 1" specifies that AUTOGRAPH should use a value "1.", "2.", or "3." - whichever it considers best.

The value "null 2" specifies that AUTOGRAPH should use a value "1.", "2.", or "3." - whichever it considers best - and replace the "null 2" by that value.

Notice that major tick marks on a linear axis may be spaced logarithmically and that major tick marks on a logarithmic axis may be spaced linearly; this is sometimes useful.

Default value: "1.E36" ("null 1") for all "s" (AUTOGRAPH spaces major tick marks as it sees fit).

''[AXIS/]s/[TICKS/]MAJOR/[SPACING/]BASE.''

Simplest form of name: 's/MA/BA.'

A floating-point number which, if greater than zero and non-null, specifies the base value ("b", in the preceding parameter description) used in spacing major tick marks in the label coordinate system along the axis specified by "s". A negative or zero value suppresses major tick marks on the axis. A null value causes AUTOGRAPH to pick an appropriate base value and, if the null was a "null 2", to backstore that value in place of the "null 2".

Default value: "1.E36" ("null 1") for all "s" (AUTOGRAPH picks the base values).
Parameters

'\[AXIS/]s/[TICKS/]MAJOR/[SPACING/]COUNT.'

Simplest form of name: 's/MA/CO.'

A floating-point number, having an integral value "n" greater than or equal to 0. A negative value is treated as if it were a zero. The value n is only used when major tick marks are to be spaced linearly and the base value ("b", in the preceding parameter descriptions) is to be chosen by AUTOGRAPH. In this case, n is a rough estimate of the minimum number of major tick marks to be placed on the axis specified by "s". The actual number used may vary between "n+2" and "5n/2+4" (approximately).

Default value: "6." for all "s" (somewhere between 8 and 19 major tick marks per linear axis).

'\[AXIS/]s/[TICKS/]MAJOR/PATTERN.'

Simplest form of name: 's/MA/PA.'

A floating-point number specifying the dashed-line pattern to be used for major tick marks on the axis specified by "s". Normally, its integer equivalent is a 16-bit integer in which "0" bits specify "pen-up" segments (gaps) 3 plotter units long and "1" bits specify "pen-down" segments (solids) 3 plotter units long. The value "0." turns off the major tick marks, the value "65535." (decimal) = "177777." (octal) makes them solid lines. If the value "null 1" is used, the next call to AGSTUP resets it to "65535." (decimal).

Default value: "1.E36" ("null 1") for all "s" (solid-line patterns).

'\[AXIS/]s/[TICKS/]MAJOR/LENGTH.'

Simplest form of name: 's/MA/LE.'

A group of two parameters determining the length of the outward-pointing and inward-pointing portions of the major tick marks on the axis specified.
by "s". Subgroups and the number of parameters in each are as follows:

' [AXIS/]s/[TICKS/]MAJOR/[LENGTH/]OUTWARD.'  (1)  
' [AXIS/]s/[TICKS/]MAJOR/[LENGTH/]INWARD.'  (1)

A floating-point number specifying the length of the outward-pointing portion of each major tick mark on the axis specified by "s". The value must be of the form "e", "1.+e", or "-e", where "e" is greater than or equal to "0." and less than "1." and represents a fraction of the smaller dimension of the grid window.

Note: "Outward" is defined relative to the normal position of the axis "s", even when that axis has been moved away from its normal position.

- When a value "e" is used, each major tick mark extends outward "e" units from the axis.

- When a value "1.+e" is used, each major tick mark extends outward to the farther edge of the grid window and then "e" units beyond that edge. (If the axis is not moved away from its normal position, "1.+e" has the same effect as "e".)

- When a value "-e" is used, the first "e" units of the inward-pointing portion of each major tick mark are erased. (This can be used to create off-axis major tick marks - for whatever that may be worth.)

Default value: "0." for all "s" (all major ticks point inward).
'[AXIS/]s/[TICKS/]MAJOR/[LENGTH/]INWARD.'

Simplest form of name: 's/MA/IN.'

A floating-point number specifying the length of the inward-pointing portion of each tick mark on the axis specified by "s". The value must be of the form "e", "l.+e", or "-e", where e is greater than or equal to "0." and less than "1." and represents a fraction of the smaller dimension of the grid window.

Note: "Inward" is defined relative to the normal position of the axis "s", even when that axis has been moved away from its normal position.

- When a value "e" is used, each major tick mark extends inward "e" units from the axis.

- When a value "l.+e" is used, each major tick mark extends inward to the farther edge of the grid window and then "e" units beyond that edge. This feature is used to create grid backgrounds.

- When a value "-e" is used, the first "e" units of the outward-pointing portion of each major tick mark are erased.

Default value: "0.15" for all "s" (all major ticks point inward).

' [AXIS/]s/[TICKS/]MINOR.'

Simplest form of name: 's/MI.'

A group of four parameters describing the minor tick marks, if any, which are to be a part of the axis specified by "s". Subgroups and the number of parameters in each are as follows:

' [AXIS/]s/[TICKS/]MINOR/SPACING.' (1)
' [AXIS/]s/[TICKS/]MINOR/PATTERN.' (1)
' [AXIS/]s/[TICKS/]MINOR/LENGTH.' (2)
'[AXIS/]s/[TICKS/]MINOR/SPACING.'

Simplest form of name: 's/MI/SP.'

A floating-point number specifying the desired number of minor tick marks to be distributed between each pair of major tick marks on the axis specified by "s". Acceptable values are as follows:

- A value less than "1." suppresses minor tick marks completely.
- A value greater than or equal to "1." which is non-null should be integral; it specifies the number of minor tick marks directly.
- The values "null 1" and "null 2" specify that AUTOGRAPH is to choose a reasonable integral value; if a "null 2" is specified, it is replaced by the integral value chosen.

The minor tick marks, if any, are spaced linearly in the label coordinate system along the axis specified by "s". Note that the appropriate value for the usual sort of logarithmic axis is "8."; this causes the minor tick marks between two major tick marks at label-system values 10**n and 10**(n+1) to be placed at the label-system values 2*10**n, 3*10**n, 4*10**n, ..., 9*10**n.

Default value: "1.E36" ("null 1") for all "s" (AUTOGRAPH chooses appropriate values).

' [AXIS/]s/[TICKS/]MINOR/PATTERN.'

Simplest form of name: 's/MI/PA.'

A floating-point number specifying the dashed-line pattern to be used for minor tick marks on the axis specified by "s"; analogous to '[AXIS/]s/[TICKS/]MAJOR/PATTERN.', described above.
Parameters

Default value: "1.E36" ("null 1") for all "s" (solid-line patterns).

'[AXIS/]s/[TICKS/]MINOR/LENGTH.'

Simplest form of name: 's/MI/LE.'

A group of two parameters determining the length of the outward-pointing and inward-pointing portions of the minor tick marks on the axis specified by "s". Subgroups and the number of parameters in each are as follows:

' [AXIS/]s/[TICKS/]MINOR/[LENGTH/]OUTWARD.' (1)
' [AXIS/]s/[TICKS/]MINOR/[LENGTH/]INWARD.' (1)

'[AXIS/]s/[TICKS/]MINOR/[LENGTH/]OUTWARD.'

Simplest form of name: 's/MI/OU.'

A floating-point number specifying the length of the outward-pointing portion of each minor tick mark on the axis specified by "s"; analogous to '...MAJOR/[LENGTH/]OUTWARD.' , described above.

Default value: "0." for all "s" (all minor ticks point inward).

'[AXIS/]s/[TICKS/]MINOR/[LENGTH/]INWARD.'

Simplest form of name: 's/MI/IN.'

A floating-point number specifying the length of the inward-pointing portion of each minor tick mark on the axis specified by "s"; analogous to '...MAJOR/[LENGTH/]INWARD.' , described above.

Default value: ".010" for all "s" (all minor ticks point inward).
A group of eight parameters describing the numeric labels, if any, which are to be a part of the axis specified by "s". Subgroups and the number of parameters in each are as follows:

'[AXIS/]s/[NUMERIC/]TYPE.'  (1)
'[AXIS/]s/[NUMERIC/]EXPO.'  (1)
'[AXIS/]s/[NUMERIC/]FRACTION.'  (1)
'[AXIS/]s/[NUMERIC/]ANGLE.'  (2)
'[AXIS/]s/[NUMERIC/]OFFSET.'  (1)
'[AXIS/]s/[NUMERIC/]WIDTH.'  (2)

The three parameters

'[AXIS/]s/[NUMERIC/]TYPE.'
'[AXIS/]s/[NUMERIC/]EXPO.'
'[AXIS/]s/[NUMERIC/]FRACTION.'

will be described together, because they are so closely interdependent. They specify the type of numeric labels to be used (at major-tick positions) on the axis specified by "s". A fourth parameter,

'[AXIS/]s/[TICKS/]MAJOR/[SPACING/]TYPE.',

described above, also affects the type of numeric labels to be used. I shall refer to these four parameters in the ensuing discussion using short forms of their names ('s/TYPE.', 's/EXPO.', 's/FRAC.', and 's/MAJOR/TYPE.', respectively).
Parameters

All four have the default value "null 1" (except for the first, which has the default value "0." for "s" = "RIGHT" and "TOP"), leaving AUTOGRAPH free to choose values which are consistent with each other and with other parameters describing the axis specified by "s". Any one or more of them may be given the value "null 2" (in which case an actual value chosen by AUTOGRAPH is backstored over the "null 2") or an actual integral floating-point value. Acceptable actual values are as follows:

- Setting 's/TYPE.' to "0." turns off the numeric labels on the axis specified by "s". The other three parameters are then ignored.

- Setting 's/TYPE.' to "1." selects "scientific" notation. Each numeric label is written in the form

  \[-\] [i] [.][f] x 10 e

  where brackets enclose portions which may be independently present or absent and "e" is a superscript exponent.

The parameter 's/EXPO.' specifies the length of "i" (the number of characters), thus also specifying the value of the exponent "e". If 's/EXPO.' has a value less than or equal to zero, "i" is omitted. If 's/EXPO.' is less than zero and has the integral absolute value "n", the fraction "f" is forced to have "n" leading zeroes.

The parameter 's/FRAC.' specifies the length of "f" (the number of characters). If 's/FRAC.' is less than or equal to zero, "f" is omitted. If 's/FRAC.' is less than zero, the decimal point is omitted.

If "[i] [.][f]" has the value "1.", the first part of the label is omitted, leaving only "10 e".

If the entire label has the value "0.", the single character "0" is used.

The value of 's/MAJOR/TYPE.' is immaterial.

- Setting 's/TYPE.' to "2." selects "exponential" notation, the exact nature of which depends on the value of 's/MAJOR/TYPE.', as follows:
If 's/MAJOR/TYP.' has the value "1." (all major ticks at values of the form plus or minus b times k), each numeric label is written in the form

\[-] [i] [.] [f] x 10 \text{e}

where brackets enclose portions which may be independently present or absent and "e" is a superscript exponent.

The parameter 's/EXPO.' specifies the integral value of the exponent "e".

The parameter 's/FRAC.' specifies the length of "f" (the number of characters). If 's/FRAC.' is less than or equal to zero, f is omitted. If 's/FRAC.' is less than zero, the decimal point is omitted.

If the label value is exactly zero, the single character "0" is used.

If 's/MAJOR/TYP.' has the value "2." (all major ticks at values of the form plus or minus b times 10 to the power k), each numeric label is written in the form

\[-] [i] [.] [f] x 10 \text{e}

where brackets enclose portions which may be independently present or absent and "e" is a superscript exponent.

The parameter 's/EXPO.' specifies the integral value of the exponent "e" when "k" equals "0." The value of "e" is 's/EXPO.' plus "k".

The parameter 's/FRAC.' specifies the length of "f" (the number of characters). If 's/FRAC.' is less than or equal to zero, "f" is omitted. If 's/FRAC.' is less than zero, the decimal point is omitted.

If the label value is exactly zero, the single character "0" is used.
If 's/MAJOR/TYPE.' has the value "3." (all major ticks at values of the form plus or minus b to the power k), each numeric label is written in the form

\[ [-] \ [i] \ [.] \ [f] \ e \]

where brackets enclose portions which may be independently present or absent and "e" is a superscript exponent.

The parameter 's/EXPO.' is ignored. The value of "e" is "k".

The parameter 's/FRAC.' specifies the length of "f" (the number of characters). If 's/FRAC.' is less than or equal to zero, "f" is omitted. If 's/FRAC.' is less than zero, the decimal point is omitted.

Note that "[i] [.] [f]" expresses the value of "b".

Setting 's/TYPE.' to "3." selects "no-exponent" notation, the exact nature of which depends on the value of 's/MAJOR/TYPE.', as follows:

If 's/MAJOR/TYPE.' has the value "1." (all major ticks at values of the form plus or minus b times k), each numeric label is written in the form

\[ [-] \ [i] \ [.] \ [f] \]

where brackets enclose portions which may be independently present or absent.

The parameter 's/EXPO.' is ignored.

The parameter 's/FRAC.' specifies the length of "f" (the number of characters). If 's/FRAC.' is less than or equal to zero, "f" is omitted. If 's/FRAC.' is less than zero, the decimal point is omitted.

If the label value is exactly zero, the single character "0" is used.
If 's/MAJOR/TYP=2.' has the value "2." (all major ticks at values of the form plus or minus b times 10 to the power k), each numeric label is written in the form

[-] [i] [.] [f]

where brackets enclose portions which may be independently present or absent.

The parameter 's/EXPO.' is ignored.

The length of "f" (the number of characters) is specified by the function

\[
\text{MAX}'s/FRAC.',0) - k
\]

if this quantity is greater than zero, and

\[
\text{MIN}'s/FRAC.',0)
\]

otherwise. This may appear somewhat formidable, but it produces a simple, desirable result. Suppose, for example, that 's/FRAC.' = "1."; "b" = "3.6", and "k" ranges from "-3" to "+3"; the labels produced are

.0036, .036, .36, 3.6, 36., 360., and 3600.

The parameter 's/FRAC.' may be viewed as specifying the length of "f" when "k" is zero. If the function value is less than or equal to zero, "f" is omitted; if it is less than zero, the decimal point is omitted.

If 's/MAJOR/TYP=3.' has the value "3." (all major ticks at values of the form plus or minus b to the power k), each numeric label is written in the form

[-] [i] [.] [f]

if "k" is greater than or equal to zero, and in the form

August 1987
Parameters

[-] 1/ [i] [.] [f]

if "k" is less than zero. Brackets enclose portions which may be independently present or absent.

The parameter 's/EXPO.' is ignored.

The length of "f" (the number of characters) is specified by the function

's/FRAC.' * ABS(k)

if "k" is non-zero, or

MIN('s/FRAC.',0)

if "k" is zero. Again, this function produces a simple result.

Suppose that 's/FRAC.' = "1.", "b" = "1.1", and "k" ranges from "-3" to "+3"; the labels produced are

1/1.331, 1/1.21, 1/1.1, 1., 1.1, 1.21, and 1.331

The parameter 's/FRAC.' may be viewed as specifying the length of "f" when "k" is equal to 1. If the function value is less than or equal to zero, "f" is omitted; if it is less than zero, the decimal point is omitted.

Another example: Suppose 's/FRAC.' = "-1.", "b" = "2.", and "k" ranges from "-4" to "+4"; the labels produced are

1/16, 1/8, 1/4, 1/2, 1, 2, 4, 8, and 16

Default value: "1.E36" ("null 1") for all three for all "s" (AUTOGRAPH chooses values to use), except for

'RIGHT/[NUMERIC/]TYPE.' and
'TOP/[NUMERIC/]TYPE.',

which are zeroed to suppress the numeric labels on the right and top axes.

August 1987
'[AXIS/]s/[NUMERIC/]EXPONENT.'

Simplest form of name: 's/EX.'

See the discussion of '[AXIS/]s/[NUMERIC/]TYPE.', above.

' [AXIS/]s/[NUMERIC/]FRACTION.'

Simplest form of name: 's/TR.'

See the discussion of '[AXIS/]s/[NUMERIC/]TYPE.', above.

' [AXIS/]s/[NUMERIC/]ANGLE.'

Simplest form of name: 's/AN.'

A group of two integral floating-point numbers specifying the orientation angle of numeric labels on the axis specified by "s". Subgroups and the number of parameters in each are as follows:

' [AXIS/]s/[NUMERIC/]ANGLE/1ST.' (1)
' [AXIS/]s/[NUMERIC/]ANGLE/2ND.' (1)

' [AXIS/]s/[NUMERIC/]ANGLE/1ST.'

Simplest form of name: 's/AN/1S.'

An integral floating-point number having one of the values "0.", "90.", "180.", or "270." (plus or minus a small multiple of "360."), specifying the user's first choice for the orientation angle of numeric labels on the axis specified by "s". The value is stated in degrees counter-clockwise
Parameters

from a left-to-right horizontal vector.

The routine AGSTUP decides whether the first choice or the second choice is to be used. The second choice is used only when the first choice leads to overlap problems and the current value of '[AXIS/]s/CONTROL.' is a "3." or a "4." and the second choice works out better than the first. If AGSTUP decides to use the first choice, it leaves the first-choice parameter with a positive value; if it decides to use the second choice, it leaves the first-choice parameter with a negative value. Values are made positive or negative by adding and subtracting multiples of "360.".

Default value: "0." for all "s" (horizontal labels preferred on all axes).

' [AXIS/]s/ [NUMERIC/] ANGLE/2ND.'

Simplest form of name: 's/AN/2N.'

An integral floating-point number having one of the values "0.", "90.", "180.", or "270." (plus or minus a small multiple of "360.") specifying the user's second choice for the orientation angle of numeric labels on the axis specified by "s". The value is stated in degrees counter-clockwise from a left-to-right horizontal vector. See the description of the preceding parameter.

Default value: "90." for all "s" (vertical labels, readable from the right, on all axes).

' [AXIS/]s/ [NUMERIC/] OFFSET.'

Simplest form of name: 's/OF.'

A floating-point number specifying the desired position of numeric labels relative to the axis specified by "s".

If the value is positive, numeric labels are to be placed toward the outside of the grid. If the value is negative, numeric labels are to be
placed toward the inside of the grid. In either of these two cases, the magnitude of the value specifies the distance from the line portion of the axis to the nearest part of any numeric label, stated as a fraction of the smaller dimension of the grid window. Note: "Inside" and "outside" are defined relative to the normal position of the axis "s", even when that axis has been moved away from its normal position.

If the value is exactly zero, each numeric label is centered on the axis. In this case, the line portion of the axis is suppressed and major and minor tick marks are moved outward so as not to overlap the numeric labels.

Default value: ".015" for all "s" (all labels outside the grid).

'[AXIS/]s/[NUMERIC/]WIDTH.'

Simplest form of name: 's/WI.'

A group of two floating-point parameters specifying the widths of characters to be used in numeric labels on the axis specified by "s". Subgroups and the number of parameters in each are as follows:

'[AXIS/]s/[NUMERIC/]WIDTH/MANTISSA.' (1)
'[AXIS/]s/[NUMERIC/]WIDTH/EXONENT.' (1)

'[AXIS/]s/[NUMERIC/]WIDTH/MANTISSA.'

Simplest form of name: 's/WI/MA.'

A floating-point number specifying the width of characters to be used in the "mantissa" of each numeric label on the axis specified by "s", expressed as a fraction of the smaller dimension of the grid window.

Default value: ".015" for all "s".
Parameters

'\[AXIS/s([NUMERIC/])WIDTH/EXPONENT.\']

Simplest form of name: 's/WI/EX.'

A floating-point number specifying the width of characters to be used in the exponent of each numeric label on the axis specified by "s", expressed as a fraction of the smaller dimension of the grid window.

Default value: ".010" for all "s".

'DASH.'

Simplest form of name: 'DAS.'

A group of thirty parameters, the first of which determines what dashed-line patterns are to be used by the routines EZMY and EZMXY and the rest of which describe the "user" set of dashed-line patterns (as opposed to the "alphabetic" set, which is defined by code in the subroutine AGCURV and is not subject to change by the user). Subgroups and the number of parameters in each are as follows:

'DASH/SELECTOR.' (1)
'DASH/LENGTH.' (1)
'DASH/CHARACTER.' (1)
'DASH/DOLLAR-QUOTE.' (1)
'DASH/PATTERNS.' (26)

'DASH/SELECTOR.'

Simplest form of name: 'DAS/SE.'

The parameter 'DASH/SELECTOR.' is given a negative integral value to specify that the routines EZMY and EZMXY should use the "alphabetic" set of 26 dashed-line patterns for the curves they draw and a positive integral value "n", less than or equal to 26, to specify that EZMY and EZMXY should
use the first "n" patterns in the "user" set of dashed-line patterns, as defined by the current values of the remaining parameters in the group 'DASH.'.

Each of the patterns in the "alphabetic" set specifies a solid line interrupted periodically by a letter of the alphabet. Each of the patterns in the "user" set is as defined by the user. The default "user" set produces all solid lines.

The routines EZY and EZZX, which draw but one curve per call, always use the first of the patterns in the "user" set; they are unaffected by the value of 'DASH/SELECTOR.'.

The selected pattern set is used in a circular fashion. For example, if 'DASH/SELECTOR.' has the value "3." and EZMY is used to draw nine curves, pattern 1 is used for curves 1, 4, and 7, pattern 2 for curves 2, 5, and 8, and pattern 3 for curves 3, 6, and 9.

Default value: "+1." (The first element of the "user" set of dashed-line patterns is to be used by EZMY and EZZM.)

----

'DASH/LENGTH.'

Simplest form of name: 'DAS/LE.'

An integral floating-point number specifying how long character-string dashed-line patterns are expected to be. In a user call to ANOTAT with a positive fifth argument (implying that the sixth argument is an array of character-string dashed-line patterns) or in a user call to AGSETC setting 'DASH/PATTERN/n.' (in which case the second argument is such a pattern), the specified character strings must be of the length specified by the current value of 'DASH/LENGTH.'.

Default value: "8." (dashed-line patterns are expected to be eight characters long).
Parameters

'DASH/CHARACTER.'

Simplest form of name: 'DAS/CH.'

A floating-point number specifying the width of each character (other than a dollar sign or a quote) which is drawn along a curve as directed by a character-string dashed-line pattern (whether from the "alphabetic" set or from the "user" set). This width is expressed as a fraction of the smaller dimension of the grid window.

Default value: ".010"

---

'DASH/DOLLAR-QUOTE.'

Simplest form of name: 'DAS/DO.'

A floating-point number specifying the line length corresponding to a dollar sign (solid) or a quote (gap) in a character-string dashed-line pattern, expressed as a fraction of the smaller dimension of the grid window.

Default value: ".010"

---

'DASH/PATTERNS.'

Simplest form of name: 'DAS/PA.'

A group of 26 parameters defining the "user" set of dashed-line patterns. Subgroups and the number of parameters in each are as follows:

'DASH/PATTERNS/1.' (1)
'DASH/PATTERNS/2.' (1)

... (1)

'DASH/PATTERNS/26.' (1)
'DASH/PATTERNS/n.'

Simplest form of name: 'DAS/PA/n.'

(The symbol "n" represents an integer between "1" and "26", inclusive.) An integral floating-point number defining the "n"th dashed-line pattern in the "user" set.

If the value is positive, it must be between "0." and "65535.", inclusive, and is interpreted as a 16-bit binary pattern in which each "0" bit specifies a "pen-up" gap segment 3 plotter units long and each "1" bit specifies a "pen-down" solid segment 3 plotter units long. Such a pattern may be defined by a user call to AGSETI or AGSETF.

If the value is negative, it serves as an identifier, allowing AUTOGRAPH to retrieve, from its character storage space, a character string in which each single quote specifies a "pen-up" gap segment, each dollar sign specifies a "pen-down" solid segment, and each other character is simply to be drawn as a part of the line. Such a pattern may be defined by a user call to AGSETC.

Note that the function "AGDSHN" allows a user to easily generate the name of the "n"th dash pattern.

Default values: "65535." for all "n" (solid lines).

'LABEL.'

Simplest form of name: 'LAB.'

A group of 3+10*n parameters, where "n" is the current value of 'LABEL/BUFFER/LENGTH.' (8, by default) describing up to "n" informational labels. These labels are a part of the background drawn by a call to the routine AGRACK. Subgroups and the number of parameters in each are as follows:
Parameters

'LABEL/CONTROL.' (1)
'LABEL/BUFFER/LENGTH.' (1)
'LABEL/BUFFER/CONTENTS.' (10*n)
'LABEL/NAME.' (1)

'LABEL/CONTROL.'

Simplest form of name: 'LAB/CO.'

An integral floating-point number having the value "0.", "1.", or "2.". Values greater than "2." are changed to a "2." by the next AGSTUP call. Values less than "0." are changed to a "0." by the next AGSTUP call; negative values have a special use, however (see below).

• The value "0." disables the drawing of informational labels. They remain defined, however.

• The value "1." enables the drawing of informational labels and specifies that they may not be shrunk in response to overlap problems.

• The value "2." enables the drawing of informational labels and specifies that they may be shrunk in response to overlap problems.

Default value: "2." (labels enabled, shrinkable).

Special action by AGSETP: An AGSETP call which sets this parameter (individually, rather than as part of a group) to a negative value results in the deletion of all currently-defined labels. Note that the negative value is changed to a zero by the next AGSTUP call; thus, the drawing of informational labels is disabled until re-enabled by the user.

'LABEL/BUFFER.'

Simplest form of name: 'LAB/BU.'
A group of $1+10\times n$ parameters, where "n" is the current value of 'LABEL/BUFFER/LENGTH.' (8, by default). Subgroups and the number of parameters in each are as follows:

- 'LABEL/BUFFER/LENGTH.' (1)
- 'LABEL/BUFFER/CONTENTS.' (10*n)

---

'LABEL/BUFFER/LENGTH.'

Simplest form of name: 'LAB/BU/LE.'

An integral floating-point number specifying the number of 10-word label definitions the label buffer will hold. A user program may need to retrieve, but must not set, the value of this parameter, since its value must match the second dimension of the label buffer.

Increasing the size of the label buffer requires modifying the AUTOGRAPH source code. See the paragraph "INFORMATIONAL LABELS", in the section "OVERVIEW".

Default value: "8.".

---

'LABEL/BUFFER/CONTENTS.'

Simplest form of name: 'LAB/BU/CO.'

This parameter group may be thought of as an array $FLLB$, dimensioned $10\times n$, containing up to $n$ 10-word label definitions. For a second subscript $j$,

- $FLLB(1,j)$ is either a floating-point "0.", saying that no label is defined by this 10-word block, or it is non-zero, in which case it identifies a character string in AUTOGRAPH's character-string storage area; the character string serves as a name for the label defined by this 10-word block. When $FLLB(1,j)$ is non-zero:
Parameters

* FLLB(2,j) is either a "0.", to enable drawing of the label, or a "1.", to disable drawing of the label,

* FLLB(3,j) and FLLB(4,j) are the x and y coordinates of the label's "basepoint", in the grid coordinate system,

* FLLB(5,j) and FLLB(6,j) are the x and y components of the label's "offset vector", stated as signed fractions of the smaller dimension of the grid window,

* FLLB(7,j) is an integral floating-point number "0.", "90.", "180.", or "270.", specifying the angle at which the label's "baseline" emanates from the end of its offset vector,

* FLLB(8,j) is an integral floating-point number specifying how the lines of the label are to be positioned relative to the end of the offset vector ("-1." to line up the left ends, "0." to line up the centers, or "+1." to line up the right ends),

* FLLB(9,j) is an integral floating-point count of the number of lines belonging to the label, and

* FLLB(10,j) is an integral floating-point pointer specifying the second subscript (in the line buffer) of the first line of the label (the one having the largest line number), or, if no lines belong to the label, a "0.".

It is not recommended that a user program change the contents of this buffer directly. Label definitions should be accessed indirectly by means of the parameters 'LABEL/NAME.' and 'LABEL/[DEFINITION/]...'.

Default values: The label buffer contains four pre-defined labels, corresponding to the four edges of the grid window. They are as follows:
The description of 'LINE/BUFFER/CONTENTS.', below, gives the default values for the definitions of the lines which belong to these labels.

'LABEL/BUFFER/NAMES.'

Simplest form of name: 'LAB/BU/NA.'

This group is a subset of the previous one. It provides a way of retrieving the names of all currently-defined labels.

'LABEL/NAME.'

Simplest form of name: 'LAB/NA.'

An integral floating-point pointer which, if non-zero, specifies a particular label in the label buffer - the one which is to be referenced by the parameter group 'LABEL/DEFINITION.' (which see, below).

Setting 'LABEL/NAME.' is the required first step in accessing a particular label definition.

Default value: "0." (undefined).

Special action by AGSETP: To access the definition of a particular label, one must first call AGSETP with 'LABEL/NAME.' as the first argument and the
name of the label one wishes to access as the second argument. This causes
AGSETP (which is called by AGSETC) to search for the definition of the
desired label in the label buffer. If that definition is not found, a new
one is made up and inserted in the label buffer. In either case,
'LABEL/NAME.' is given a floating-point value whose integer equivalent
specifies the second subscript of the label definition in the label buffer.

The definition of a new label has the name specified by the user, a
suppression flag "0.", a basepoint (.5,.5), an offset vector (0.0), a
baseline angle "0.", a centering option "0.", a line count "0.", and a
first-line index "0.".

'LABEL/DEFINITION.'

Simplest form of name: 'LAB/DE.'

A set of nine parameters defining the label specified by the current value
of 'LABEL/NAME.'. If 'LABEL/NAME.' has the value "0.", referencing this
group or a parameter in it causes an error exit. Subgroups and the number
of parameters in each are as follows:

'LABEL/[DEFINITION/]SUPPRESSION.' (1)
'LABEL/[DEFINITION/]BASEPOINT.' (2)
'LABEL/[DEFINITION/]OFFSET.' (2)
'LABEL/[DEFINITION/]ANGLE.' (1)
'LABEL/[DEFINITION/]CENTERING.' (1)
'LABEL/[DEFINITION/]LINES.' (1)
'LABEL/[DEFINITION/]INDEX.' (1)

'LABEL/[DEFINITION/]SUPPRESSION.'

Simplest form of name: 'LAB/SU.'

An integral floating-point "suppression flag" having the value "0." or "1."
and specifying whether drawing of the label specified by 'LABEL/NAME.' is

August 1987 121
AUTOGRAPH: A Graphing Utility

enabled ("0.") or disabled ("1.").

Default value for a new label: "0." (label enabled).

Special action by AGSETP: If a user program attempts to set this parameter
(individually, rather than as part of a group) to a negative value, the
lines of the label specified by 'LABEL/NAME.' are deleted and
'LINE/NUMBER.' is zeroed. If the negative value is less than "-1."
the label is deleted as well and 'LABEL/NAME.' is zeroed. (Deleting a label
means that its name cell is set to "0.".)

'LABEL/[DEFINITION/]BASEPOINT.'

Simplest form of name: 'LAB/BA.'

A set of two parameters specifying the x and y coordinates of the basepoint
of the label specified by 'LABEL/NAME.', in the grid coordinate system.
The label is positioned relative to this basepoint. Subgroups and the
number of parameters in each are as follows:

'LABEL/[DEFINITION/]BASEPOINT/X.' (1)
'LABEL/[DEFINITION/]BASEPOINT/Y.' (1)

'LABEL/[DEFINITION/]BASEPOINT/X.'

Simplest form of name: 'LAB/BA/X.'

The x coordinate of the basepoint of the label specified by 'LABEL/NAME.'.
The value "0." refers to the left edge of the grid window, the value "1."
to the right edge of the grid window.

Default value for a new label: ".5" (centered).
Parameters

'LABEL/[DEFINITION/]BASEPOINT/Y.'

Simplest form of name: 'LAB/BA/Y.'

The y coordinate of the basepoint of the label specified by 'LABEL/NAME.'.
The value "0." refers to the bottom edge of the grid window, the value "1." to the top edge of the grid window.

Default value for a new label: ".5" (centered).

'LABEL/[DEFINITION/]OFFSET.'

Simplest form of name: 'LAB/OF.'

A set of two parameters specifying the x and y components of the offset vector of the label specified by 'LABEL/NAME.', as signed fractions of the smaller dimension of the grid window. The offset vector has its basepoint at the label basepoint. Subgroups and the number of parameters in each are as follows:

'LABEL/[DEFINITION/]OFFSET/X.' (1)
'LABEL/[DEFINITION/]OFFSET/Y.' (1)

'LABEL/[DEFINITION/]OFFSET/X.'

Simplest form of name: 'LAB/OF/X.'

The x component of the offset vector of the label specified by 'LABEL/NAME.' - negative toward the left edge, positive toward the right edge, of the grid window. The magnitude represents a fraction of the smaller dimension of the grid window.

Default value for a new label: "0." (zero-length vector).
'LABEL/[DEFINITION/]OFFSET/Y.'

Simplest form of name: 'LAB/OF/Y.'

The y component of the offset vector of the label specified by 'LABEL/NAME.' - negative toward the bottom edge, positive toward the top edge, of the grid window. The magnitude represents a fraction of the smaller dimension of the grid window.

Default value for a new label: "0." (zero-length vector).

'LABEL/[DEFINITION/]ANGLE.'

Simplest form of name: 'LAB/AN.'

An integral floating-point number having one of the values "0.", "90.", "180.", or "270.", and specifying the direction in which the baseline of the label specified by 'LABEL/NAME.' emanates from the end of its offset vector, measured counter-clockwise from a left-to-right horizontal vector. All the lines of a label are written parallel to its baseline and in the direction of the baseline.

Default value for a new label: "0." (horizontal, left to right).

'LABEL/[DEFINITION/]CENTERING.'

Simplest form of name: 'LAB/CE.'

An integral floating-point number specifying the alignment of the lines of the label specified by 'LABEL/NAME.' with the end of its offset vector. A negative value aligns the left ends, a zero value the centers, and a positive value the right ends, of the lines.

Default value for a new label: "0." (centers aligned).
'LABEL/[DEFINITION/] LINES.'

Simplest form of name: 'LAB/LI.'

An integral floating-point number specifying the number of lines in the label specified by 'LABEL/NAME.'.

This parameter is updated by AUTOGRAPH as lines are added to or deleted from the label and should not be set by a user program.

Default value for a new label: "0." (no lines).

'LABEL/[DEFINITION/] INDEX.'

Simplest form of name: 'LAB/IN.'

An integral floating-point number specifying the second subscript (in the line buffer) of the first line belonging to the label specified by 'LABEL/NAME.' - a zero if no line belongs to the label.

This parameter is updated by AUTOGRAPH as lines are added to or deleted from the label and should not be set by a user program.

Default value for a new label: "0." (no lines).

'LINE.'

Simplest form of name: 'LIN.'

A group of 4+6*n parameters, where "n" is the current value of 'LINE/BUFFER/LENGTH.' (16, by default) describing up to "n" lines, each of which is a part of some informational label. Subgroups and the number of parameters in each are as follows:
'LINE/MAXIMUM.'  
Simplest form of name: 'LIN/MA.'

An integral floating-point number specifying the assumed maximum length of a character string delivered to AUTOGRAPH for use as the text of a label line. Such a character string may occur as the first argument of a call to ANOTAT (defining the text of line "100." in the label 'L'), as the second argument of a call to ANOTAT (defining the text of line "-100." in the label 'B'), as the last argument of a call to one of the routines EZY, EZXY, EZMY, or EZMXY (defining the text of line "100." in the label 'T'), or as the second argument of a call to AGSETC whose first argument is 'LINE/DEFINITION/TEXT.' (defining the text of any line). In each of these cases, the character string must be of the length specified by 'LINE/MAXIMUM.' or shorter. If it is shorter, its last character must be the character specified by 'LINE/END.', described below.

This parameter may be given any desired non-negative integral value.

Default value: "40.".

'LINE/END.'
Simplest form of name: 'LIN/EN.'
Parameters

A character string whose first character is the one used to mark the end of a character string defining the text of a label line (in calls to ANOTAT, EZY, EZXY, EZMY, EZMXY, and AGSETC), when that character string is shorter than the current maximum specified by 'LINE/MAXIMUM.' (as described above).

The terminator character does not become a part of the text of the line. It is stripped off, so that only the preceding characters constitute the text of the line.

Default value: "'$'".

'LINE/BUFFER.'

Simplest form of name: 'LIN/BU.'

A group of 1+6*n parameters, where "n" is the current value of 'LINE/BUFFER/LENGTH.' (16, by default). Subgroups and the number of parameters in each are as follows:

'LINE/BUFFER/LENGTH.' (1)
'LINE/BUFFER/CONTENTS.' (6*n)

'L OLINE/BUFFER/LENGTH.'

Simplest form of name: 'LIN/BU/LE.'

An integral floating-point number specifying the number of 6-word line definitions the line buffer will hold. A user program may need to retrieve, but must not set, the value of this parameter, since its value must match the second dimension of the line buffer.

Increasing the size of the line buffer requires modifying the AUTOGRAPH source code. See the paragraph "INFORMATIONAL LABELS", in the section "OVERVIEW".

August 1987
'LINE/BUFFER/CONTENTS.'

Simplest form of name: 'LIN/BU/CO.'

This group may be thought of as an array FLIN, dimensioned 6xn, containing up to n 6-word line definitions. For a second subscript j,

- FLIN(1,j) is either a floating-point "null 1", saying that no label line is defined by this 6-word block, or an integral floating-point "line number", saying that it does define a label line, in which case:
  - FLIN(2,j) is either "0.", to enable drawing of the line, or "1.", to disable drawing of the line.
- FLIN(3,j) is the floating-point width of each character of the line, stated as a fraction of the smaller dimension of the grid window.
- FLIN(4,j) is an integral floating-point number serving as the identifier of the character string defining the text of the line and enabling it to be retrieved from AUTOGRAPH's internal character storage space.
- FLIN(5,j) is an integral floating-point count of the number of characters in the text of the line.
- FLIN(6,j) is an integral floating-point number specifying the second subscript (in the line buffer) of the next line of the label to which this line belongs (that one of the remaining lines in the chain with the largest line number) or, if there is no next line, a "0.".

It is not recommended that a user program change the contents of this buffer directly. Line definitions should be accessed indirectly by means of the parameters 'LINE/NUMBER.' and 'LINE/[DEFINITION/]...'.

Default values: The line buffer contains four pre-defined lines, each of which belongs to one of the four pre-defined labels. They are as follows:
The description of 'LABEL/BUFFER/CONTENTS.', above, gives default values for the definitions of the four labels which contain these lines.

'LINE/NUMBER.'

Simplest form of name: 'LIN/NU.'

An integral floating-point pointer which, if non-zero, specifies a particular line in the line buffer - the one which is to be referenced by the parameter group 'LINE/DEFINITION.' (which see, below).

Setting this parameter is the required first step in accessing a particular line definition.

Default value: "0." (undefined).

Special action by AGSETP: To access the definition of a particular line of a particular label, one must ensure that 'LABEL/NAME.' (which see, above) is set. Then, one must call AGSETI with 'LINE/NUMBER.' as the first argument and the number of the line one wishes to access as the second argument. This causes AGSETP (which is called by AGSETI) to search the line buffer for the definition of a line belonging to the label specified by the current value of 'LABEL/NAME.' and having the desired line number. If no such definition is found, a new one is made up, inserted in the line buffer, and linked into the proper place in the chain of lines belonging to the label. In either case, 'LINE/NUMBER.' is given an integral floating-point value specifying the second subscript of the line definition in the line buffer.
The definition of a new line has the number specified by the user, a suppression flag "0.", a character width ".015", a pointer to the text string "' '", and a text length "1.".

Note: The "line numbers" are used to identify the lines of a label and to specify their positions relative to each other and to the baseline of the label. Lines having positive line numbers are drawn above the label baseline, lines having zero line numbers are drawn along the label baseline, and lines having negative line numbers are drawn below the label baseline. A line having a greater line number than another line is drawn above that line. ("Above" and "below" are used here from the viewpoint of someone reading the label.) The magnitudes of the line numbers in no way affect inter-line spacing, which is determined by AUTOGRAPH itself.

'LINE/DEFINITION.'

Simplest form of name: 'LIN/DE.'

A group of five parameters defining the line specified by 'LINE/NUMBER.'.

If 'LINE/NUMBER.' has the value "0.", referencing a parameter in this group causes an error exit. Subgroups and the number of parameters in each are as follows:

'LINE/[DEFINITION/]SUPPRESSION.' (1)
'LINE/[DEFINITION/]CHARACTER.' (1)
'LINE/[DEFINITION/]TEXT.' (1)
'LINE/[DEFINITION/]LENGTH.' (1)
'LINE/[DEFINITION/]INDEX.' (1)

'LINE/[DEFINITION/]SUPPRESSION.'

Simplest form of name: 'LIN/SU.'

An integral floating-point number having the value "0." or "1." and specifying whether drawing of the line specified by 'LINE/NUMBER.' is enabled
Parameters

("O.") or disabled ("1.").

Default value for a new line: "0." (line enabled).

Special action by AGSETP: If a user program attempts to set this parameter (individually, rather than as a part of a group) to a negative value, the line specified by 'LINE/NUMBER.' is deleted and 'LINE/NUMBER.' is reset to "0.". (Deleting a line means that it is unlinked from the chain of lines belonging to its label and that its number cell is set to "null 1".)

'LINE/[DEFINITION/] CHARACTER.'

Simplest form of name: 'LIN/CH.'

A floating-point number specifying the desired width of each character of the line specified by 'LINE/NUMBER.', stated as a fraction of the smaller dimension of the grid window.

Default value for a new line: ".015".

'LINE/[DEFINITION/] TEXT.'

Simplest form of name: 'LIN/TE.'

An integral floating-point number serving as an identifier for a character string stored away in AUTOGRAPH's internal character storage space.

Default value for a new line: "' ".

Special action by AGSETP: When this parameter is set by a call to AGSETC, the character string appearing as the second argument of AGSETC is stored in a character storage array inside AUTOGRAPH, an identifier allowing for later retrieval of the string is generated, and the value of that identifier is stored (by AGSETP, which is called by AGSETC) as the parameter value. At that time, the length of the string is determined and 'LINE/[DEFINITION/]LENGTH.' is set. See 'LINE/MAXIMUM.' and
AUTOGRAPH: A Graphing Utility

'LINE/TERMINATOR.', above.

'LINE/[DEFINITION/]LENGTH.'

Simplest form of name: 'LIN/LE.'

An integral floating-point count of the number of characters in the text of the line specified by 'LINE/NUMBER.' Setting this parameter less than or equal to zero suppresses the drawing of the line. See also the description of 'LINE/[DEFINITION/]TEXT.', above.

Default value for a new line: "1." (one character - a blank).

'LINE/[DEFINITION/]INDEX.'

Simplest form of name: 'LIN/IN.'

An integral floating-point number specifying the second subscript (in the line buffer) of the next line of the label - a zero if there is no next line.

This parameter is updated by AUTOGRAPH as lines are added to or deleted from the label and should not be set by a user program.

'SECONDARY.'

Simplest form of name: 'SEC.'

A group of 149 "secondary" control parameters. These are not normally set by a user program, but are computed by AUTOGRAPH itself (the routine AGSTUP). Their values may be of use in some applications. Subgroups and the number of parameters in each are as follows:
Parameters

'SECONDARY/GRAPH.' (4)
'SECONDARY/USER.' (4)
'SECONDARY/CURVE.' (4)
'SECONDARY/DIMENSIONS.' (3)
'SECONDARY/AXIS.' (80)
'SECONDARY/LABEL.' (54)

'SECONDARY/GRAPH.'

Simplest form of name: 'SEC/GR.'

A group of four floating-point numbers specifying the x coordinates of the left and right edges of the graph window and the y coordinates of the bottom and top edges of the graph window, in the grid coordinate system. These values are used by AUTOGRAPH to determine whether a point whose coordinates are expressed in the grid coordinate system lies inside or outside the graph window.

If the parameters in the group 'GRID.' have their default values (".15", ".95", ".15", ".95", and "0."), these four parameters will be given the values ":1875", ":1625", ":1875", and ":0625". Note that ":1875 = (0-.15)/(.95-.15)" and that ":0625 = (1-.15)/(.95-.15)."

'SECONDARY/USER.'

Simplest form of name: 'SEC/US.'

A set of four floating-point numbers specifying the x coordinates of the left and right edges of the grid window and the y coordinates of the bottom and top edges of the grid window, in the user coordinate system. These values are used in mapping user curve points into the grid window. The routines AGSTUP, AGBACK, and AGCURV use these four numbers as arguments 5 through 8 in calls to the system-plot-package routine SET.
AUTOGRAPH: A Graphing Utility

'SECONDARY/CURVE.'

Simplest form of name: 'SEC/CU.'

A group of four floating-point numbers specifying the x coordinates of the left and right edges of the grid (curve) window and the y coordinates of the bottom and top edges of the grid (curve) window. The x coordinates are stated as fractions of the distance from left to right, and the y coordinates as fractions of the distance from bottom to top, in the plotter frame. The routines AGSTUP, AGBACK, and AGCURV use these four numbers as arguments 1 through 4 in calls to the system-plot-package routine SET. If the parameters in the groups 'GRAPH.' and 'GRID.' have their default values, these four parameters are given the values ".15", ".95", ".15", and ".95".

'SECONDARY/DIMENSIONS.'

Simplest form of name: 'SEC/DI.'

A group of three floating-point numbers, the first two of which specify the width and height of the grid window and the third of which is equal to the smaller of the first two. Each is stated as a number of plotter units. If the parameters in the groups 'GRAPH.' and 'GRID.' have their default values and the plotter being used has 1024x1024 addressable positions, then each of these three parameters will be given the value $818.4 = (.95-.15) \times 1023$.

'SECONDARY/AXIS.'

Simplest form of name: 'SEC/AX.'

A group of eighty parameters having to do with the drawing of the four axes. Subgroups and the number of parameters in each are as follows:
The parameters from the subgroups are interleaved in the group; that is to say, the first elements of the subgroups comprise elements 1 through 4 of the group, the second elements of the subgroups comprise elements 5 through 8 of the group, and so on.

'SECONDARY/[AXIS/] s.'

Simplest form of name: 'SEC/s.'

A group of twenty parameters having to do with the drawing of the axis specified by "s", where "s" is one of the keywords "LEFT", "RIGHT", "BOTTOM", or "TOP". Subgroups and the number of parameters in each are as follows:

'SECONDARY/[AXIS/] s/POSITION.' (6)
'SECONDARY/[AXIS/] s/TICKS.' (3)
'SECONDARY/[AXIS/] s/NUMERIC.' (11)

'SECONDARY/[AXIS/] s/POSITION.'

Simplest form of name: 'SEC/s/PO.'

A group of six floating-point numbers, the first three of which describe a point at the beginning of axis "s" and the last three of which describe a point at the end of axis "s". The first two numbers of each triplet are the x and y coordinates of the point, in the grid coordinate system. The third number of each triplet is a user-system x or y coordinate (an x coordinate for a horizontal axis, a y coordinate for a vertical axis) of the point.
'SECONDARY/[AXIS/]s/TICKS.'

Simplest form of name: 'SEC/s/TI.'

A group of three floating-point numbers, specifying the values AUTOGRAPH has chosen to use for the primary parameters

'[AXIS/]s/[TICKS/]MAJOR/[SPACING/]TYPE.'
'[AXIS/]s/[TICKS/]MAJOR/[SPACING/]BASE.'
'[AXIS/]s/[TICKS/]MINOR/SPACING.'

(which see, above). These secondary parameters are used to hold the values AUTOGRAPH chooses for the corresponding primary parameters so as not to disturb "null 1" values of those primary parameters.

'SECONDARY/[AXIS/]s/Numeric.'

Simplest form of name: 'SEC/s/NU.'

A group of eleven floating-point numbers having to do with the generation of numeric labels on the axis specified by "s". The first three of these specify the values AUTOGRAPH has chosen to use for the primary parameters

'[AXIS/]s/[NUMERIC/]TYPE.'
'[AXIS/]s/[NUMERIC/]EXPONENT.'
'[AXIS/]s/[NUMERIC/]FRACTION.'

(which see, above). The secondary parameters are used so as not to disturb "null 1" values of the primary parameters.

The fourth parameter is an integral floating-point count of the number of characters in the longest numeric-label mantissa on the axis "s".

The fifth parameter is an integral floating-point count of the number of characters in the longest numeric-label exponent on the axis "s".
The sixth parameter is the necessary multiplicative "reduction factor" (between "0." and "1.") to be applied to the sizes of numeric labels on the axis "s" in order to make them fit without overlap problems.

The seventh, eighth, ninth, and tenth parameters are floating-point numbers specifying the width of the space required by numeric labels to the left (outward), to the right (inward), at the beginning and at the end of the axis "s" - each is stated as a fraction of the width or height of the grid window, depending on the orientation of the axis "s".

The eleventh parameter indicates the linear/log nature of the axis specified by "s".

'SECONDARY/LABEL.'

Simplest form of name: 'SEC/LA.'

A group of fifty-four parameters describing the six "label boxes", each of which provides a mechanism for moving and/or shrinking a particular group of labels in attempting to keep any label in that group from overlapping an axis or extending outside the current graph window. Subgroups and the number of parameters in each are as follows:

'SECONDARY/LABEL/LEFT.'  (9)
'SECONDARY/LABEL/RIGHT.'  (9)
'SECONDARY/LABEL/BOTTOM.' (9)
'SECONDARY/LABEL/TOP.'    (9)
'SECONDARY/LABEL/CENTER.'  (9)
'SECONDARY/LABEL/GRAPH.'   (9)

The parameters of the subgroups are interleaved in the group. The first elements of the subgroups form elements 1 through 6 of the group, the second elements of the subgroups form elements 7 through 12 of the group, and so on.
AUTOGRAPH: A Graphing Utility

'SECONDARY/LABEL/b.'

Simplest form of name: 'SEC/LA/b.'

where the keyword "b" specifies the label box, as follows:

- If "b" = "LEFT", label box 1 is specified. It contains all labels having a basepoint on the left edge of the grid window and a leftward-pointing offset vector. These labels are to be moved leftward as required to avoid overlapping any numeric labels on either y axis.

- If "b" = "RIGHT", label box 2 is specified. It contains all labels having a basepoint on the right edge of the grid window and a rightward-pointing offset vector. These labels are to be moved rightward as required to avoid overlapping any numeric labels on either y axis.

- If "b" = "BOTTOM", label box 3 is specified. It contains all labels having a basepoint on the bottom edge of the grid window and a downward-pointing offset vector. These labels are to be moved downward as required to avoid overlapping any numeric labels on either x axis.

- If "b" = "TOP", label box 4 is specified. It contains all labels having a basepoint on the top edge of the grid window and an upward-pointing offset vector. These labels are to be moved upward as required to avoid overlapping any numeric labels on either x axis.

- If "b" = "CENTER", label box 5 is specified. It contains all labels having a basepoint on some edge of the grid window and an inward-pointing offset vector. These labels are to be moved inward as required to avoid overlapping numeric labels on any axis.

- If "b" = "GRAPH", label box 6 is specified. It contains all labels not specifically assigned to one of the other boxes. These labels are not moved, but may still be shrunk as required to avoid their running outside the grid window.
Prior to a call to AGSTUP, the nine parameters in this group are undefined. Following an AGSTUP call, but preceding an AGBACK call, they have what I shall call "interim" values. Following an AGBACK call, they have what I shall call "final" values.

The first parameter in the group is a "reduction factor" for the widths of characters in the labels in box "b". This parameter may have the interim value "0.", specifying that no actual value has yet been computed, or "1.", specifying that the user has prohibited shrinkage of labels in box "b" (by giving 'LABEL/CONTROL.' the value "1."). The final value of the reduction factor may be "-1.", specifying that minimum-sized labels were used, but even they led to overlap problems, or a value between "0." and "1.", specifying the actual reduction factor applied when the labels were drawn.

The next four parameters in the group specify the grid-system x coordinates of the left and right edges, and the grid-system y coordinates of the bottom and top edges, of label box "b". The interim values specify the box in which the labels must be made to fit in order to avoid overlap, the final values the box in which the labels were actually made to fit.

The last four parameters in the group specify the grid-system x coordinates of the left and right edges, and the grid-system y coordinates of the bottom and top edges, of the label box "b" which would result if all the labels were reduced to minimum size. The interim values specify an unmoved box, the final values a (possibly) moved box.
Section IV: Examples
EXAMPLES

This write-up contains the FORTRAN text and Xerox laser printer output from thirteen jobs using AUTOGRAPH to produce graphs. These graphs illustrate many of the capabilities of AUTOGRAPH.

The duplex character set of PWRITX was used for all labelling. See the paragraph "USE OF PWRITX BY AUTOGRAPH", in the section "OVERVIEW".

Additions to this set of examples will be gratefully accepted. Of particular interest are examples illustrating pitfalls of AUTOGRAPH and the manner in which they may be avoided, examples showing creative solutions to graphing problems which are apt to be encountered frequently by others (examples 9 and 11 are additions of this sort), and, last but not least, examples of graphs which just "look neat".

Each of the example programs uses a subroutine called BNDARY to draw a boundary line indicating where the edge of the plotter frame is. This subroutine is not a part of AUTOGRAPH or of the NCAR plot package. It is written as follows:

```
SUBROUTINE BNDARY
C
C Routine to draw the plotter-frame edge.
C
CALL PLOTIT ( 0, 0,0)
CALL PLOTIT (32767, 0,1)
CALL PLOTIT (32767,32767,1)
CALL PLOTIT ( 0,32767,1)
CALL PLOTIT ( 0, 0,1)
RETURN
END
```
EXAMPLE 1

This example illustrates a simple use of EZY. Note that AUTOGRAPH chose to use exponential left-axis numeric labels, since non-exponential labels would have required more characters.

```
PROGRAM EXMPL1

C
C Define the data array.
C
```
REAL YDRA(1001)

C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data array.
C
DO 101 I=1,1001
X=FLOAT(I)/20.
YDRA(I)=10.*((X-1.)*(X-11.)*(X-21.)*(X-31.)*(X-41.)*
 + (X-51.)+2.E7*(FRAN()-0.5)
101 CONTINUE
C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Draw the graph, using EZY.
C
CALL EZY(YDRA,1001,'EXAMPLE 1 (EZY)' )
C
C Close GKS.
C
CALL CLSGKS
C
STOP
C
END

FUNCTION FRAN()
C
C Random-number generator.
C
DATA X / 2.7182818 /
SAVE X
X=AMOD(9821.*X+.211327,1.)
FRAN=X
RETURN
END
EXAMPLE 2

This example illustrates a simple use of EZXY. Note that x coordinates used need not be monotonically increasing.
PROGRAM EXMPL2 C C Define the data arrays. C REAL XDRA(4001),YDRA(4001) C C Initialize GKS. C CALL OPNGKS C C Fill the data arrays. C DO 101 I=1,4001
   THETA=.0015707963267949*FLOAT(I-1)
   RHO=SIN(2.*THETA)+.05*SIN(64.*THETA)
   XDRA(I)=RHO*COS(THETA)
   YDRA(I)=RHO*SIN(THETA)
101 CONTINUE C C Draw a boundary around the edge of the plotter frame. C CALL BNDARY C C Draw the graph, using EZXY. C CALL EZXY (XDRA,YDRA,4001,'EXAMPLE 2 (EZXY)$') C C Close GKS. C CALL CLSGKS C STOP C END
EXAMPLE 3

This example illustrates a simple use of EZMY. A remarkably uninteresting graph.
PROGRAM EXMPL3
C
C Define the data array.
C
REAL YDRA(100,2)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data array.
C
DO 101 I=1,100
   YDRA(I,1)=COS(3.14159265358979*FLOAT(I)/25.)*FLOAT(I)**2
   YDRA(I,2)=COS(3.14159265358979*FLOAT(I)/25.)*10.**(.04*FLOAT(I))
101 CONTINUE
C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Draw the graph, using EZMY.
C
CALL EZMY(YDRA,100,2,100,'EXAMPLE 3 (EZMY)'
C
C Close GKS.
C
CALL CLSGKS
C
STOP
C
END
EXAMPLE 4

This example illustrates a simple use of EZMXY - nested ellipses.
PROGRAM EXMPL4
C
C Define the data arrays.
C
REAL XDRA(201),YDRA(201,10)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data arrays.
C
DO 102 I=1,201
   XDRA(I)=-1.+0.02*FLOAT(I-1)
   IF (I.GT.101) XDRA(I)=2.-XDRA(I)
   DO 101 J=1,10
      YDRA(I,J)=FLOAT(J)*
      SQRT(1.000000000001-XDRA(I)**2)/10.
      IF (I.GT.101) YDRA(I,J)=-YDRA(I,J)
  101 CONTINUE
  102 CONTINUE
C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Draw the graph, using EZMXY.
C
CALL EZMXY (XDRA,YDRA,201,10,201,'EXAMPLE 4 (EZMXY)$')
C
C Close GKS.
C
CALL CISGKS
C
STOP
C
END

August 1987 151
EXAMPLE 5

This example illustrates a relatively simple use of EZMXY; generating the data requires using some really fascinating numeric constants. Essentially, six circles are drawn; two small portions of each are blanked out by salting in some "null 1's" (1.E36's). The result is a possibly recognizable commercial logo.
PROGRAM EXMPL5

C Define the data arrays.
C
REAL XDRA(401,6),YDRA(401,6)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Compute required constants.
C
PI=3.14159265358979
PID200=PI/200.
PITWO=2.*PI
PIT2D3=2.*PI/3.
PIT4D3=4.*PI/3.
RADOSC=SQRT(3.)/3.
RADOLC=SQRT(3.)/2.
BSSCLL=ATAN(SQRT(12.)/6.)
BSSCUL=ATAN(SQRT(143.)/7.)
BSSLCL=ATAN(SQRT(143.)/17.)
BSSLCL=ATAN(SQRT(2.0))
C
C Fill the data arrays.
C
DO 101 I=1,401
  THETA=PID200*FLOAT(I-1)
  XDRA(I,1)= -.5+RADOSC*COS(THETA)
  YDRA(I,1)= RADOSC*SIN(THETA)
  IF (ABS(THETA ).GE.BSSCLL.AND.
      + ABS(THETA-PITWO).GE.BSSCLL.AND.
      + ABS(THETA-PIT2D3).LE.BSSCUL) XDRA(I,1)=1.E36
  IF (ABS(THETA-PITWO).GE.BSSCLL.AND.
      + ABS(THETA-PIT2D3).LE.BSSCUL) XDRA(I,1)=1.E36
  XDRA(I,2)= .5+RADOSC*COS(THETA)
  YDRA(I,2)= RADOSC*SIN(THETA)
  IF (ABS(THETA-PIT2D3).GE.BSSCLL.AND.
      + ABS(THETA-PIT4D3).LE.BSSCUL) XDRA(I,2)=1.E36
  IF (ABS(THETA-PIT4D3).GE.BSSCLL.AND.
      + ABS(THETA-PIT2D3).LE.BSSCUL) XDRA(I,2)=1.E36
  XDRA(I,3)= RADOSC*COS(THETA)
  YDRA(I,3)=RADOLC+RADOSC*SIN(THETA)

August 1987
IF (ABS(THETA-PIT4D3).GE.BSSCLL.AND.
+ ABS(THETA-PIT4D3).LE.BSSCUL) XDRA(I,3)=1.E36
XDRA(I,4)= -.5+RADOLC*COS(THETA)
YDRA(I,4)= RADOLC*SIN(THETA)

IF (ABS(THETA ).GE.BSCLL.AND.
+ ABS(THETA ).LE.BSCUL) XDRA(I,4)=1.E36

IF (ABS(THETA-PIT2D3).GE.BSCLL.AND.
+ ABS(THETA-PIT2D3).LE.BSCLUL) XDRA(I,5)=1.E36
XDRA(I,6)= RADOLC*COS(THETA)
YDRA(I,6)=RADOLC+RADOLC*SIN(THETA)

C Specify subscripting of XDRA and YDRA.
C
CALL AGSETI ('ROW.',2)

C Set up grid shape to make 1 unit in x = 1 unit in y.
C
CALL AGSETF ('GRID/SHAPE.',2.)

C Turn off background, then turn labels back on.
C
CALL AGSETF ('BACKGROUND.',4.)
CALL AGSETI ('LABEL/CONTROL.',2)

C Turn off left label.
C
CALL AGSETC ('LABEL/NAME.','L')
CALL AGSETI ('LABEL/SUPPRESSION FLAG.',1)

C Change text of bottom label.
C
CALL AGSETC ('LABEL/NAME.','B')
CALL AGSETI ('LINE/NUMBER.',-100)
CALL AGSETC ('LINE/TEXT,' 'PURITY, BODY, AND FLAVOR$')
C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Draw the graph, using EZMXY.
C
CALL EZMXY (XDRA,YDRA, 401,6,401,'EXAMPLE 5 (EZMXY)$')
C
C Close GKS.
C
CALL CLSGKS
C
STOP
C
END
EXAMPLE 6

This example illustrates the use of the graph window and the four principal types of backgrounds one can use: perimeter, grid, half-axis, and "none". Note that I have cheated a bit and turned the labels back on for the last of these. This example also illustrates the use of linear and logarithmic x and y mappings.
PROGRAM EXMPL6
C
C Define the data arrays.
C
REAL XDRA(501),YDRA(501)
C
C Define the graph-window parameter array.
C
REAL GWND (4,4)
C
DATA (GWND(I,1),I=1,4) / 0.0, 0.5, 0.5, 1.0 /
DATA (GWND(I,2),I=1,4) / 0.5, 1.0, 0.5, 1.0 /
DATA (GWND(I,3),I=1,4) / 0.0, 0.5, 0.0, 0.5 /
DATA (GWND(I,4),I=1,4) / 0.5, 1.0, 0.0, 0.5 /
C
C Define variables used in setting up informational labels.
C
CHARACTER*35 GLAB
CHARACTER*23 BACK(4)
CHARACTER*12 INLG(4)
C
DATA BACK(1) / '(PERIMETER BACKGROUND)$' /
DATA BACK(2) / '(GRID BACKGROUND)$' /
DATA BACK(3) / '(HALF-AXIS BACKGROUND)$' /
DATA BACK(4) / '(NO BACKGROUND)$' /
C
DATA INLG(1) / 'LINEAR$' /
DATA INLG(2) / 'LOGARITHMIC$' /
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data arrays.
C
DO 101 I=1,501
   THETA=.031415926535898*FLOAT(I-1)
   XDRA(I)=500.+ .9*FLOAT(I-1)*COS(THETA)
   YDRA(I)=500.+ .9*FLOAT(I-1)*SIN(THETA)
101 CONTINUE
AUTOGRAPH: A Graphing Utility

C Suppress the frame advance.
CALL AGSETI ('FRAME.',2)

C Do four graphs on the same frame, using different backgrounds.
DO 102 IGRF = 1,4

C Position the graph window.
CALL AGSETP ('GRAPH WINDOW.',GWND(1,IGRF),4)

C Declare the background type.
CALL AGSETI ('BACKGROUND TYPE.',IGRF)

C Setting the background type may have turned informational labels off. In that case, turn them back on.
IF (IGRF.EQ.4) CALL AGSETI ('LABEL/CONTROL.',2)

C Set up parameters determining linear/log nature of axes.
ILLX=(IGRF-1)/2
ILLY=MOD(IGRF-1,2)

C Declare the linear/log nature of the graph.
CALL AGSETI ('X/LOGARITHMIC.',ILLX)
CALL AGSETI ('Y/LOGARITHMIC.',ILLY)

C Change the x- and y-axis labels to reflect the linear/log nature of the graph.
CALL AGSETIC ('LABEL/NAME.','B')
CALL AGSETI ('LINE/NUMBER.',-100)
CALL AGSETIC ('LINE/TEXT.',INLG(ILLX+1))

August 1987
CALL AGSETC ("LABEL/NAME.",'L')
CALL AGSETI ("LINE/NUMBER.",100)
CALL AGSETC ("LINE/TEXT.",LNLG(ILLY+1))

C
C Set up the label for the top of the graph.
C
WRITE (GLAB,1001) IGRF,BACK(IGRF)

C
C Draw the graph, using EZXY.
C
CALL EZXY (XDRA,YDRA,501,GLAB)

C
102 CONTINUE
C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Advance the frame.
C
CALL FRAME
C
C Close GKS.
C
CALL CLSGKS
C
STOP
C
C Format for encode.
C
1001 FORMAT ('EXAMPLE 6-',I1,' ',A23)
END
EXAMPLE 7

This example illustrates several features. It shows how to define informational labels. It shows how label coordinate systems along an axis are defined. Windowing is used to prevent the curves from running wild. The "user" set of dashed-line patterns is employed. ("Incrudescence" is a word that I invented, by the way.)
PROGRAM EXMPL7

C Define the data arrays and the dash-pattern array.
C
REAL XDRA(101),YDRA(101,9)
CHARACTER*28 DSHP(9)

C Declare the type of the dash-pattern-name generator.
C
CHARACTER*16 AGDSHN

C Initialize GKS.
C
CALL OPNGKS

C Fill the data arrays and the dash pattern array.
C
DO 101 I=1,101
   XDRA(I)=-90. +1.8*FLOAT(I-1)
101 CONTINUE

DO 103 J=1,9
   WRITE (DSHP(J),1001) J, FJ=J
   DO 102 I=1,101
      YDRA(I,J)=3.*FJ-(FJ/2700.)*XDRA(I)**2
102 CONTINUE
103 CONTINUE

C Turn on windowing.
C
CALL AGSETI ('WINDOWING.',1)

C Move the edges of the curve window (grid).
C
CALL AGSETF ('GRID/LEFT.' ,.10)
CALL AGSETF ('GRID/RIGHT.' ,.90)
CALL AGSETF ('GRID/BOTTOM.' ,.10)
CALL AGSETF ('GRID/TOP.' ,.85)

August 1987 161
C Set the x and y minimum and maximum.
C
CALL AGSETF ('X/MINIMUM.', -90.)
CALL AGSETF ('X/MAXIMUM.', +90.)
CALL AGSETF ('Y/MINIMUM.', 0.)
CALL AGSETF ('Y/MAXIMUM.', 18.)
C
C Set left axis parameters.
C
CALL AGSETI ('LEFT/MAJOR/TYP\text{}E.', 1)
CALL AGSETF ('LEFT/MAJOR/BASE.', 3.)
CALL AGSETI ('LEFT/MINOR/SPACING.', 2)
C
C Set right axis parameters.
C
CALL AGSETI ('RIGHT/FUNCTION.', 1)
CALL AGSETF ('RIGHT/NUMERIC/TYP\text{}E.', 1.E36)
C
C Set bottom axis parameters.
C
CALL AGSETI ('BOTTOM/MAJOR/TYP\text{}E.', 1)
CALL AGSETF ('BOTTOM/MAJOR/BASE.', 15.)
CALL AGSETI ('BOTTOM/MINOR/SPACING.', 2)
C
C Set top axis parameters.
C
CALL AGSETI ('TOP/FUNCTION.', 2)
CALL AGSETF ('TOP/NUMERIC/TYP\text{}E.', 1.E36)
C
C Set up the dash patterns to be used.
C
CALL AGSETI ('DASH/SELE\text{}CTOR.', 9)
CALL AGSETI ('DASH/LENGTH.', 28)
DO 104 I=1,9
   CALL AGSETC (AGDSHN(I), DSHP(I))
104 CONTINUE
C
C Set up the left label.
C
CALL AGSETC ('LABEL/NAME.', 'L')
CALL AGSETI ('LINE/NUMBER.', 100)
CALL AGSETC ('LINE/TEXT.', 'HEIGHT (KILOMETERS)$')
C
C Set up the right label.
C
CALL AGSETC ('LABEL/NAME.', 'R')
CALL AGSETI ('LINE/NUMBER.', -100)
CALL AGSETC ('LINE/TEXT.',
+ 'PRESSURE (TONS/SQUARE FURLONG)$')
C
C Set up the bottom labels.
C
CALL AGSETC ('LABEL/NAME.', 'B')
CALL AGSETI ('LINE/NUMBER.', -100)
CALL AGSETC ('LINE/TEXT.', 'LATITUDE (DEGREES)$')
C
CALL AGSETC ('LABEL/NAME.', 'SP')
CALL AGSETF ('LABEL/BASEPOINT/X.', .000001)
CALL AGSETF ('LABEL/BASEPOINT/Y.', 0.)
CALL AGSETF ('LABEL/OFFSET/Y.', -.015)
CALL AGSETI ('LINE/NUMBER.', -100)
CALL AGSETC ('LINE/TEXT.', 'SP$')
C
CALL AGSETC ('LABEL/NAME.', 'NP')
CALL AGSETF ('LABEL/BASEPOINT/X.', .999999)
CALL AGSETF ('LABEL/BASEPOINT/Y.', 0.)
CALL AGSETF ('LABEL/OFFSET/Y.', -.015)
CALL AGSETI ('LINE/NUMBER.', -100)
CALL AGSETC ('LINE/TEXT.', 'NP$')
C
C Set up the top label.
C
CALL AGSETC ('LABEL/NAME.', 'T')
CALL AGSETI ('LINE/NUMBER.', 80)
CALL AGSETC ('LINE/TEXT.',
+ 'DISTANCE FROM EQUATOR (MILES)$')
CALL AGSETI ('LINE/NUMBER.', 90)
CALL AGSETC ('LINE/TEXT.', '$')
CALL AGSETI ('LINE/NUMBER.', 100)
CALL AGSETC ('LINE/TEXT.',

August 1987

163
AUTOGRAPH: A Graphing Utility

\[ \text{\'LINES OF CONSTANT INCRUDESCENCE\'} \]
CALL AGSETI ('LINE/NUMBER.',110)
CALL AGSETC ('LINE/TEXT.','EXAMPLE 7 (EZMXY)\$')

C
C Set up centered (box 6) label.
C
CALL AGSETC ('LABEL/NAME.','EQUATOR')
CALL AGSETI ('LABEL/ANGLE.',90)
CALL AGSETI ('LINE/NUMBER.',0)
CALL AGSETC ('LINE/TEXT.','EQUATOR\$')

C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Draw the graph, using EZMXY.
C
CALL EZMXY (XDRA,YDRA,101,9,101,0)
C
C Close GKS.
C
CALL CLSGKS
C
STOP
C
C Format for encode above.
C
1001 FORMAT ('$$$$$$$$$$$$$$$$$$$$$$J'=",Il,"')
C
END

SUBROUTINE AGUTOL (IAXS,FUNS,IDMA,VINP,VOTP)
C
C Mapping for the right axis.
C
IF (FUNS.EQ.1.) THEN
   IF (IDMA.GT.0) VOTP=ALOG10(20.-VINP)
   IF (IDMA.LT.0) VOTP=20.-10.**VINP
C
C Mapping for the top axis.
C
ELSE IF (FUNS.EQ.2.) THEN
   IF (IDMA.GT.0) VOTP=70.136*VINP
   IF (IDMA.LT.0) VOTP=VINP/70.136
   C
   C Default (identity) mapping.
   C
   ELSE
      VOTP=VINP
   END IF
   C
   C Done.
   C
   RETURN
   C
   END
EXAMPLE 8

This example is a somewhat ugly graph demonstrating that one can plot "x as a function of y", that label values can run "backwards" along an axis, and that axes need not have major tick marks at their ends. The "alphabetic" set of dashed-line patterns is used. Major tick marks on the x axis are extended in both directions.
PROGRAM EXMPL8
C
C Define the data arrays.
C
REAL XDRA(101), YDRA(4,101)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data arrays.
C
    DO 101 I=1,101
        XDRA(I)=-3.14159265358979+
        + .062831853071796*FLOAT(I-1)
    101 CONTINUE
C
    DO 103 I=1,4
        FLTI=I
        BASE=2.*FLTI-1.
        DO 102 J=1,101
            YDRA(I,J)=BASE+.75*SIN(-3.14159265358979+
            + .062831853071796*FLTI*FLOAT(J-1))
        102 CONTINUE
    103 CONTINUE
C
C Change the line-end character to a period.
C
CALL AGSETC ('LINE/END.', '.')
C
C Specify labels for x and y axes.
C
CALL ANOTAT ('SINE FUNCTIONS OF T.', 'T.', 0, 0, 0, 0)
C
C Use a half-axis background.
C
CALL AGSETI ('BACKGROUND.', 3)
C
C Move x axis to the zero point on the y axis.
C
August 1987

167
AUTOGRAPH: A Graphing Utility

CALL AGSETF ('BOTTM/INTERSECTION/USER.', 0.)

C Specify base value for spacing of major ticks on x axis.
C
CALL AGSETF ('BOTTM/MAJOR/BASE.', 1.)

C Run major ticks on x axis to edge of curve window.
C
CALL AGSETF ('BOTTM/MAJOR/INWARD.', 1.)
CALL AGSETF ('BOTTM/MAJOR/OUTWARD.', 1.)

C Position x axis minor ticks.
C
CALL AGSETI ('BOTTM/MINOR/SPACING.', 9)

C Run the y axis backward.
C
CALL AGSETI ('Y/ORDER.', 1)

C Run plots full-scale in y.
C
CALL AGSETI ('Y/NICE.', 0)

C Have AUTOGRAPH scale x and y data the same.
C
CALL AGSETF ('GRID/SHAPE.', .01)

C Use the alphabetic set of dashed-line patterns.
C
CALL AGSETI ('DASH/SELECTOR.', -1)

C Tell AUTOGRAPH how the data arrays are dimensioned.
C
CALL AGSETI ('ROW.', -1)

C Reverse the roles of the x and y arrays.
C
CALL AGSETI ('INVERT.', 1)

C Draw a boundary around the edge of the plotter frame.
Examples

C
C     CALL BNDARY
C
C Draw the curves.
C
C     CALL EZMXY (XDRA,YDRA,4,4,101,'EXAMPLE 8. ')
C
C Close GKS.
C
C     CALL CLSGKS
C
C STOP
C
C END
EXAMPLE 9

This example shows how to create a hybrid logarithmic axis with positive values on one end and negative values on the other; a line of discontinuity separates the two. The routine AGCHNL is replaced by a version which forces the numeric label in the middle of the left axis to be blank.
PROGRAM EXMPL9

C Define the data arrays.
C
DIMENSION XDAT(400),YDAT(400)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data arrays.
C
DO 101 I=1,400
   XDAT(I)=(FLOAT(I)-1.)/399.
101 CONTINUE
C
CALL GENDAT (YDAT( 1),200,200,1,3,3,+ .01,+ 10.)
CALL GENDAT (YDAT(201),200,200,1,3,3,-10.,-.01)
C
C The y data ranges over both positive and negative values. It is desired that both ranges be represented on the same graph and that each be shown logarithmically, ignoring values in the range −.01 to +.01, in which we have no interest. First we map each y datum into its absolute value (.01 if the absolute value is too small). Then we take the base-10 logarithm, add 2.0001 (so as to be sure of getting a positive number), and re-attach the original C sign. We can plot the resulting y data on a linear y axis.
C
DO 102 I=1,400
   YDAT(I)=SIGN(ALOG10(AMAX1(ABS(YDAT(I)),.01))+2.0001,
       + YDAT(I))
102 CONTINUE
C
C In order that the labels on the y axis should show the original values of the y data, we change the user-system–to-label-system mapping on both y axes and force major ticks to be spaced logarithmically in the label system (which will be defined by the subroutine AGUTOL in such a way as to re-create numbers in the
AUTOGRAPH: A Graphing Utility

C original range).
C
CALL AGSETI ( 'LEFT/FUNCTION.', 1)
CALL AGSETI ( 'LEFT/MAJOR/TYPE.', 2)
C
CALL AGSETI ( 'RIGHT/FUNCTION.', 1)
CALL AGSETI ( 'RIGHT/MAJOR/TYPE.', 2)
C
C Change the left-axis label to reflect what's going on.
C
CALL AGSETC ( 'LABEL/NAME.', 'L')
CALL AGSETI ( 'LINE/NUMBER.', 100)
CALL AGSETC ( 'LINE/TEXT.',
            + 'LOG SCALING, POSITIVE AND NEGATIVES')
C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Draw the curve.
C
CALL EZXY (XDAT,YDAT,400,'EXAMPLE 9$')
C
C Close GKS.
C
CALL CLSGKS
C
STOP
C
END
SUBROUTINE AGUTOL (IAXS,FUNS,IDMA,VINP,VOTP)
C
C Left or right axis.
C
IF (FUNS.EQ.1.) THEN
  IF (IDMA.LT.0) THEN
    VOTP=SIGN(ALOG10(AMAX1(ABS(VINP),.01))+2.0001,VINP)
  ELSE
    VOTP=SIGN(10.**(ABS(VINP)-2.0001),VINP)
  END IF
Examples

C All others.
C
ELSE
VOTP=VINP
END IF
C
C Done.
C
RETURN
C
END
SUBROUTINE AGCHNL (IAXS,VILS,CHRM,MCIM,NCIM,IPXM,
+ CHRE,MCIE,NCIE)
C
CHARACTER(*) CHRM,CHRE
C
C Modify the left-axis numeric label marking the value "0."
C
IF (IAXS.EQ. 1.AND.VILS.EQ. 0.) THEN
  CHRM(1:1)=''
  NCIM=1
  IPXM=0
  NCIE=0
END IF
C
C Done.
C
RETURN
C
END
SUBROUTINE GENDAT (DATA,IDIM,M,N,MLOW,MHIGH,DLOW,DHIGH)
C
C This is a routine to generate test data for two-dimensional
C graphics routines. Given an array "DATA", dimensioned
C "IDIM,1", it fills the sub-array ((DATA(I,J),I=1,M),J=1,N)
C with a two-dimensional field of data having approximately
C "MLOW" lows and "MHIGH" highs, a minimum value of exactly
C "DLOW" and a maximum value of exactly "DHIGH".
C
August 1987 173
"MLOW" and "MHGH" are each forced to be greater than or equal to 1 and less than or equal to 25.

The function used is a sum of exponentials.

DIMENSION DATA(IDIM, 1), CCNT(3, 50)

FOVM = 9. / FLOAT(M)
FOVN = 9. / FLOAT(N)

NLOW = MAX0(1, MIN0(25, MLOW))
NHGH = MAX0(1, MIN0(25, MHGH))
NCNT = NLOW + NHGH

DO 101 K = 1, NCNT
   CCNT(1, K) = 1. + (FLOAT(M) - 1.) * FRAN(
   CCNT(2, K) = 1. + (FLOAT(N) - 1.) * FRAN(
   IF (K .LE. NLOW) THEN
      CCNT(3, K) = -1.
   ELSE
      CCNT(3, K) = +1.
   END IF
101 CONTINUE

EMIN = +1. E36
DMAX = -1. E36
DO 104 J = 1, N
   DO 103 I = 1, M
      DATA(I, J) = .5 * (DLOW + DHGH)
   DO 102 K = 1, NCNT
      DATA(I, J) = DATA(I, J) + .5 * (DHGH - DLOW) * CCNT(3, K) * EXP(-((FOVM * (FLOAT(I) - CCNT(1, K)))**2 + (FOVN * (FLOAT(J) - CCNT(2, K)))**2))
102 CONTINUE
   DMIN = AMIN1(DMIN, DATA(I, J))
   DMAX = AMAX1(DMAX, DATA(I, J))
103 CONTINUE
104 CONTINUE

DO 106 J = 1, N
Examples

DO 105 I=1,M
    DATA(I,J) = (DATA(I,J) - DMIN) / (DMAX - DMIN) * (DHGH - DLW) + DLOW
105 CONTINUE
106 CONTINUE
C
    RETURN
C
    END
FUNCTION FRAN()
C
    DATA X / 2.7182818 /
    SAVE X
    X = AMOD(9821. * X + .211327, 1.)
    FRAN = X
    RETURN
END

Random-number generator.

DATA X / 2.7182818 /
SAVE X
X = AMOD(9821. * X + .211327, 1.)
FRAN = X
RETURN
END
EXAMPLE 10

This example shows how to create non-standard numeric labels on the axes. The bottom x axis is labelled with the names of the months, and the left y axis with Roman numerals.
PROGRAM XMPL10
C
C Define the data arrays.
C
REAL XDRA(1201),YDRA(1201)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data arrays. The independent variable represents
time during the year (a hypothetical year with equal-length
months) and is set up so that minor ticks can be lengthened
to delimit the months; the major ticks, though shortened to
invisibility, still determine where the labels go.
C
DO 101 I=1,1201
   XDRA(I)=FLOAT(I-51)
   YDRA(I)=COSH(FLOAT(I-601)/202.)
101 CONTINUE
C
C Change the labels on the bottom and left axes.
C
CALL ANOTAT ('MONTHS OF THE YEAR$','ROMAN NUMERALS$',0,0,0,0)
C
C Fix the minimum and maximum values on both axes and prevent
AUTOGRAPH from using rounded values at the ends of the axes.
C
CALL AGSETF ('X/MIN.',-50.)
CALL AGSETF ('X/MAX.',1150.)
CALL AGSETI ('X/NICE.',0)
C
CALL AGSETF ('Y/MIN.',1.)
CALL AGSETF ('Y/MAX.',10.)
CALL AGSETI ('Y/NICE.',0)
C
C Specify the spacing between major tick marks on all axes.
C Note that the AUTOGRAPH dummy routine AGCHNL is supplanted
C (below) by one which supplies dates for the bottom axis and

August 1987
AUTOGRAPH: A Graphing Utility

C Roman numerals for the left axis in place of the numeric
C labels one would otherwise get.
C
CALL AGSETI (' LEFT/MAJOR/TYPE.',1)
CALL AGSETI (' RIGHT/MAJOR/TYPE.',1)
CALL AGSETI ('BOTTOM/MAJOR/TYPE.',1)
CALL AGSETI ('TOP/MAJOR/TYPE.',1)

CALL AGSETF (' LEFT/MAJOR/BASE.', 1.)
CALL AGSETF (' RIGHT/MAJOR/BASE.', 1.)
CALL AGSETF ('BOTTOM/MAJOR/BASE.',100.)
CALL AGSETF (' TOP/MAJOR/BASE.',100.)

C Suppress minor ticks on the left and right axes.

CALL AGSETI ('LEFT/MINOR/SPACING.',0)
CALL AGSETI ('RIGHT/MINOR/SPACING.',0)

C On the bottom and top axes, put one minor tick between each
C pair of major ticks, shorten major ticks to invisibility,
C and lengthen minor ticks. The net effect is to make the
C minor ticks delimit the beginning and end of each month,
C while the major ticks, though invisible, cause the names of
C the months to be where we want them.
C
CALL AGSETI ('BOTTOM/MINOR/SPACING.',1)
CALL AGSETI ('TOP/MINOR/SPACING.',1)

CALL AGSETF ('BOTTOM/MAJOR/INWARD. ',0.)
CALL AGSETF ('BOTTOM/MINOR/INWARD. ',.015)
CALL AGSETF ('TOP/MAJOR/INWARD. ',0.)
CALL AGSETF ('TOP/MINOR/INWARD. ',.015)

C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Draw the graph, using EZXY.
C
CALL EZXY (XDRA,YDRA,1201,
EXAMPLE 10 (MODIFIED NUMERIC LABELS)

C Close GKS.
C
CALL CLSGKS
C
STOP
C
END
SUBROUTINE AGCHNL (IAXS, VILS, CHRM, MCIM, NCIM, IPXM,
+ CHRE, MCIE, NCIE)
C
CHARACTER*(*) CHRM, CHRE
C
C Define the names of the months for use on the bottom axis.
C
CHARACTER*3 MONS(12)
DATA MONS /
+ 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN',
+ 'JUL', 'AUG', 'SEP', 'OCT', 'NOV', 'DEC' /
C
C Modify the numeric labels on the left axis.
C
IF (IAXS.EQ.1) THEN
    CALL AGCORN (IFIX(VILS), CHRM, NCIM)
    IPXM=0
    NCIE=0
C
C Modify the numeric labels on the bottom axis.
C
ELSE IF (IAXS.EQ.3) THEN
    IMON=IFIX(VILS+.5)/100+1
    CHRM(1:3)=MONS(IMON)
    NCIM=3
    IPXM=0
    NCIE=0
END IF
C
C Done.
C
RETURN
AUTOGRAPH: A Graphing Utility

END
SUBROUTINE AGCORN (NTGR, BCRN, NCRN)

CHARACTER*(*) BCRN

C This routine receives an integer in NTGR and returns its
C Roman-numeral equivalent in the first NCRN characters of
C the character variable BCRN. It only works for integers
C within a limited range and it does some rather unorthodox
C things (like using zero and minus).
C
C ICH1, ICH5, and IC10 are character variables used for the
C single-unit, five-unit, and ten-unit symbols at a given
C level.
C
CHARACTER*1 ICH1, ICH5, IC10

C Treat numbers outside the range (-4000,+4000) as infinites.
C
IF (IABS(NTGR).GE.4000) THEN
  IF (NTGR.GT.0) THEN
    NCRN=5
    BCRN(1:5)='(INF)'
  ELSE
    NCRN=6
    BCRN(1:6)='(−INF)'
  END IF
  RETURN
END IF

C Use a '0' for the zero. The Romans never had it so good.
C
IF (NTGR.EQ.0) THEN
  NCRN=1
  BCRN(1:1)=0'
  RETURN
END IF

C Zero the character counter.
C
NCRN=0
C
Handle negative integers by prefixing a minus sign.
C
IF (NTGR.LT.0) THEN
   NCRN=NCRN+1
   BCRN(NCRN:NCRN)='-'
END IF
C
C Initialize constants. We'll check for thousands first.
C
IMOD=10000
IDIV=1000
ICH1='M'
C
C Find out how many thousands (hundreds, tens, units) there
C are and jump to the proper code block for each case.
C
101 INTG=MOD(IABS(NTGR),IMOD)/IDIV
C
   GO TO (107,104,104,104,102,103,103,103,103,106),INTG+1
C
C Four - add ICH1 followed by ICH5.
C
102 NCRN=NCRN+1
   BCRN(NCRN:NCRN)=ICH1
C
C Five through eight - add ICH5, followed by INTG-5 ICH1's.
C
103 NCRN=NCRN+1
   BCRN(NCRN:NCRN)=ICH5
C
   INTG=INTG-5
   IF (INTG.LE.0) GO TO 107
C
C One through three - add that many ICH1's.
C
104 DO 105 I=1,INTG
   NCRN=NCRN+1
   BCRN(NCRN:NCRN)=ICH1
C
   GO TO (107,104,104,104,104,102,103,103,103,103,106),INTG+1
C
   NCRN=NCRN+1
   BCRN(NCRN:NCRN)=ICH5
AUTOGRAPH: A Graphing Utility

BCRN(NCRN:NCRN)=ICH1

105 CONTINUE

C

GO TO 107

C

C Nine - add ICH1, followed by IC10.

C

106 NCRN=NCRN+1
    BCRN(NCRN:NCRN)=ICH1
    NCRN=NCRN+1
    BCRN(NCRN:NCRN)=IC10

C

C If we're done, exit.

C

107 IF (IDIV.EQ.1) RETURN

C

C Otherwise, tool up for the next digit and loop back.

C

    IMOD=IMOD/10
    IDIV=IDIV/10
    IC10=ICH1

C

    IF (IDIV.EQ.100) THEN
        ICH5='D'
        ICH1='C'
    ELSE IF (IDIV.EQ.10) THEN
        ICH5='L'
        ICH1='X'
    ELSE
        ICH5='V'
        ICH1='I'
    END IF

C

    GO TO 101

C

END
EXAMPLE 11

This example shows how to create a scattergram.
PROGRAM XMPL11
C
C Create a scattergram.
C
REAL XDRA(500),YDRA(500)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data arrays.
C
DO 101 I=1,500
   XDRA(I)=.5+(2.*(FRAN()-0.5))**5
   YDRA(I)=.5+(2.*(FRAN()-0.5))**5
101 CONTINUE
C
C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Suppress the frame advance.
C
CALL AGSETI ('FRAME.',2)
C
C Suppress the drawing of curves by the EZ... routines.
C
CALL AGSETI ('SET.',-1)
C
C Draw the background, using EZXY.
C
CALL EZXY (XDRA,YDRA,500,'EXAMPLE 11 (SCATTERGRAM) $')
C
C Put a plus sign at each of the x-y positions.
C
CALL POINTS (XDRA,YDRA,500,-2,0)
C
C Advance the frame.
C
CALL FRAME
C
C Close GKS.
C
CALL CLSGKS
C
STOP
C
END
FUNCTION FRAN()
C
C Random-number generator.
C
DATA X / 2.7182818 /
SAVE X
X=AMOD(9821.*X+.211327,1.)
FRAN=X
RETURN
END
EXAMPLE 12

This example shows how to create the effect of a histogram. Shading is done by an XLIB utility called FILL.
PROGRAM XMPL12

C Create a sort of histogram.
C
REAL XDRA(249), YDRA(249), WORK(204), IWRK(204)
C
C Initialize GKS.
C
CALL OPNGKS
C Fill the data arrays. First, we define the histogram outline. This will be used in the call to FILL which fills in the area under the histogram.
C
XDRA(1)=0.
YDRA(1)=0.
C
DO 101 I=2,100,2
   XDRA(I )=XDRA(I-1)
   YDRA(I )=EXP(-16.*(FLOAT(I/2)/50.-.51)**2)+.1*FRAN()
   XDRA(I+1)=XDRA(I-1)+.02
   YDRA(I+1)=YDRA(I)
101 CONTINUE
C
XDRA(102)=1.
YDRA(102)=0.
C
C Define lines separating vertical boxes from each other.
C
NDRA=102
C
DO 102 I=3,99,2
   XDRA(NDRA+1)=1.E36
   YDRA(NDRA+1)=1.E36
   XDRA(NDRA+2)=XDRA(I)
   YDRA(NDRA+2)=0.
   XDRA(NDRA+3)=XDRA(I)
   YDRA(NDRA+3)=AMIN1(YDRA(I),YDRA(I+1))
   NDRA=NDRA+3
102 CONTINUE
C
AUTOGRAPH: A Graphing Utility

C Draw a boundary around the edge of the plotter frame.
C
   CALL BNDARY
C
C Suppress the frame advance.
C
   CALL AGSETI ('FRAME.',2)
C
C Draw the graph, using EZXY.
C
   CALL EZXY (XDRA,YDRA,249,'EXAMPLE 12 (HISTOGRAM)$')
C
C Use the XLIB routine FILL to fill the area defined by the
C data. Note that FILL is not part of the AUTOGRAPH package.
C
   CALL FILLOP ('AN',45)
   CALL FILLOP ('SP',128)
   CALL FILL (XDRA,YDRA,102,WORK,204,IWRK,204)
C
C Advance the frame.
C
   CALL FRAME
C
C Close GKS.
C
   CALL CLSGKS
C
   STOP
C
   END
FUNCTION FRAN()
C
C Random-number generator.
C
   DATA X / 2.7182818 /
   SAVE X
   X=AMOD(9821.*X+.211327,1.)
   FRAN=X
   RETURN
END
A FINAL EXAMPLE

The final example conveys an appropriate message. It demonstrates two of the peculiar types of numeric labels which may be generated. (There are others.) It also demonstrates the use of dashed minor ticks (a hellishly time-consuming business which sometimes produces a nice effect).
AUTOGRAPH: A Graphing Utility

PROGRAM EXMPLF

C Define the data array.
C
DIMENSION XYCD(226)
C
C Initialize GKS.
C
CALL OPNGKS
C
C Fill the data array.
C
READ 1001 , XYCD
C
DO 101 I=1, 226
   IF (XYCD(I).EQ.1.E36) GO TO 101
   XYCD(I)=2.**((XYCD(I)-15.)/2.5)
101 CONTINUE
C
C Specify log/log plot.
C
   CALL DISPLA (0,0,4)
C
C Bump the line-maximum parameter past 42.
C
   CALL AGSETI ('LINE/MAXIMUM. ', 50)
C
C Specify x- and y-axis labels, grid background.
C
   CALL ANOTAT('LOGARITHMIC, BASE 2, EXPONENTIAL LABELS$',
               + 'LOGARITHMIC, BASE 2, NO-EXPONENT LABELS$',
               + 2,0,0,0)
C
C Specify the graph label.
C
   CALL AGSETC ('LABEL/NAME.', 'T')
   CALL AGSETI ('LINE/NUMBER.', 100)
   CALL AGSETC ('LINE/TEXT.', 'FINAL EXAMPLE$')
C
C Specify x-axis ticks and labels.
CALL AGSETI ("BOTTOM/MAJOR/TYPe.", 3)
CALL AGSETF ("BOTTOM/MAJOR/BASE.", 2.)
CALL AGSETI ("BOTTOM/NUMERIC/TYPe.", 2)
CALL AGSETI ("BOTTOM/MINOR/SPACING.", 4)
CALL AGSETI ("BOTTOM/MINOR/PATTERN.", 125252B)

C Specify y-axis ticks and labels.
C
CALL AGSETI ("LEFT/MAJOR/TYPe.", 3)
CALL AGSETF ("LEFT/MAJOR/BASE.", 2.)
CALL AGSETI ("LEFT/NUMERIC/TYPe.", 3)
CALL AGSETI ("LEFT/MINOR/SPACING.", 4)
CALL AGSETI ("LEFT/MINOR/PATTERN.", 125252B)

C Compute secondary control parameters.
C
CALL AGSTUP (XYCD(1),1,0,113,2,XYCD(2),1,0,113,2)
C
C Draw the background.
C
CALL AGBACK
C
C Draw the curve twice to make it darker.
C
CALL AGCURV (XYCD(1),2,XYCD(2),2,113,1)
CALL AGCURV (XYCD(1),2,XYCD(2),2,113,1)

C Draw a boundary around the edge of the plotter frame.
C
CALL BNDARY
C
C Advance the frame.
C
CALL FRAME
C
C Close GKS.
C
CALL CLSGKS

August 1987
AUTOGRAPH: A Graphing Utility

STOP
C
C Format.
C
1001 FORMAT (14E5.0)
C
END

August 1987
Section V: Error Messages
ERROR MESSAGES

AUTOGRAPH routines detect certain errors and, in response, call the routine SETER, which is an adapted version of a PORT error handler. Currently, all such errors are treated as being fatal and cause termination of the job.

An error message is logged before the job is terminated. Each such message includes the name of the routine which detected the error and may be accompanied by supplementary information aimed at allowing the user to easily identify the call that caused the error. The possible error messages are as follows (in alphabetical order):

AGEXAX (CALLED BY AGSTUP) - USER-SYSTEM-TO-LABEL-SYSTEM MAPPING IS NOT MONOTONIC

This probably means that you have replaced the default routine AGUTOL with a version of your own, and you've blown it.

AGGETC - PARAMETER TO GET IS NOT INTRINSICALLY OF TYPE CHARACTER

The argument TPGN specifies a parameter which is not intrinsically of type character. See the description of AGGETC, in the section "ROUTINES".

AGGETP OR AGSETP - ATTEMPT TO ACCESS LABEL ATTRIBUTES BEFORE SETTING LABEL NAME

The parameter 'LABEL/NAME.' must be set prior to the call which gave the error message, specifying which label's attributes are being referenced.

AGGETP OR AGSETP - ATTEMPT TO ACCESS LINE ATTRIBUTES BEFORE SETTING LINE NUMBER

The parameter 'LINE/NUMBER.' must be set prior to the call which gave the

August 1987
error message, specifying which label line's attributes are being referenced.

AGGETP OR AGSETP - ILLEGAL KEYWORD USED IN PARAMETER IDENTIFIER

The argument TPGN contains an unrecognizable keyword.

AGKURV - NUMBER OF POINTS IS .LE. 0

The argument NEXY, in a call to AGCURV, is less than or equal to zero. The routine AGKURV is called by AGCURV to draw un-windowed curves.

AGNUMB - MANTISSA TOO LONG
AGNUMB - EXPONENT TOO LARGE
AGNUMB - ZERO-LENGTH MANTISSA

AGNUMB is called by AGAXIS to generate a character string expressing the value of a floating-point number. You should not be able to generate any of AGNUMB's error messages. If you do, see the AUTOGRAPH specialist.

AGQURV - NUMBER OF POINTS IS .LE. 0

The argument NEXY, in a call to AGCURV, is less than or equal to zero. The routine AGQURV is called by AGCURV to draw windowed curves.

AGRSTR - ERROR ON READ
AGRSTR - END-OF-FILE ON READ

Probably the unit specified by IFNO was not positioned properly.
AGSAVE - ERROR ON WRITE

A system error has occurred as a result of the attempted "WRITE".

AGSETC - PARAMETER TO SET IS NOT INTRINSICALLY OF TYPE CHARACTER

This means that the argument TPGN specifies some parameter other than one of the acceptable possibilities. See the description of AGSETC in the section "ROUTINES".

AGSETP - ATTEMPT TO DEFINE LINE OF NON-EXISTENT LABEL

The user has attempted to define a line of a label without first specifying which label; 'LABEL/NAME.' must be set prior to the call which gave the error message.

AGSETP - LABEL LIST OVERFLOW - SEE AUTOGRAPH SPECIALIST

The user has attempted to define more labels than AUTOGRAPH can handle; a modification of AUTOGRAPH is required. See the paragraph "INFORMATIONAL LABELS", in the section "OVERVIEW", or talk to the AUTOGRAPH specialist.

AGSETP - LINE LIST OVERFLOW - SEE AUTOGRAPH SPECIALIST

The user has attempted to define more label lines than AUTOGRAPH can handle; a modification of AUTOGRAPH is required. See the paragraph "INFORMATIONAL LABELS", in the section "OVERVIEW", or talk to the AUTOGRAPH specialist.

AGSTCH - CHARACTER-STRING BUFFER OVERFLOW - SEE CONSULTANT

August 1987
The routine AGSTCH is called by AGSETC to stash the character string in AUTOGRAPH's character storage space. The available storage space has been exhausted. See the consultant.

AGSTCH - CHARACTER-STRING INDEX OVERFLOW - SEE CONSULTANT

The routine AGSTCH is called by AGSETC to stash the character string in AUTOGRAPH's character storage space. Too many such strings have been stored. See the consultant.

AGSTUP - GRAPH WINDOW IMPROPERLY SPECIFIED

The parameters in the group named 'GRAPH.' have improper values.

AGSTUP - GRID WINDOW IMPROPERLY SPECIFIED

The parameters in the group named 'GRID.' have improper values. This is most likely to occur when 'SET.' has the value "2." or "4.", specifying that the edges of the grid window are to be as implied by the last call to the plot package routine SET. Check to make sure that the portion of the plotter frame specified by the last SET call is within the current graph window.

AGSTUP - s LABELS IMPROPERLY SPECIFIED

where "s" = "LEFT", "RIGHT", "BOTTOM", "TOP", or "INTERIOR".

Re-read the paragraph "THE LABEL BOXES", in the section "OVERVIEW". You have defined a label with a basepoint on one edge of the grid window and an offset vector pointing outward, some part of which extends inside the grid window (or vice-versa). This is not allowed.
Appendix A: Conversion from FORTRAN 66 AUTOGRAPH to FORTRAN 77 AUTOGRAPH
APPENDIX A:
CONVERSION FROM FORTRAN 66 AUTOGRAPH TO FORTRAN 77 AUTOGRAPH

This article is reprinted from the RECORD, vol. 5(10), October 1, 1984.

AUTOGRAPH TO BE UPDATED

A FORTRAN-77 version of AUTOGRAPH has been written. Character strings are handled differently in the updated version. All references to NCAR's plot-package support routines GETCHR, SETCHR, ULIBER, and LOC have been removed. The new code should be easily portable to any machine with a FORTRAN-77 compiler and enough memory. More importantly, the new AUTOGRAPH and code which uses it should continue to run on NCAR's CRAY-1 computers as we move to COS 1.13 and later systems. The new version will soon be made available for interested users. Sometime thereafter, it will be made the default version, and eventually the old version will be withdrawn. Each step of this process will be announced in the Daily Bulletin.

The new version required changes in the user interface. The author regrets the resulting inconvenience; conversion problems have been reduced to the minimum amount consistent with putting AUTOGRAPH on a healthy footing. It is hoped that the code written according to the new rules will not need to be changed significantly with the advent of new versions of FORTRAN and new operating systems.

This article is intended to serve as a guide for the conversion of programs from the old AUTOGRAPH to the new one. New versions of the AUTOGRAPH write-ups are being prepared and will be announced when ready.

Effects of the Change in FORTRAN Character Handling

The fact that the Hollerith data type of FORTRAN-66 was not carried over into FORTRAN-77 has necessitated a number of changes. Although Hollerith constructions such as the use of "nH..." and the storing of character strings in variables of type REAL or type INTEGER may still work on some machines (the CRAY-1 machines, for example), use of them must be discouraged, since they are not officially included in FORTRAN-77.

All character-string arguments must now be FORTRAN-77 character expressions. This includes the following:
AUTOGRAPH: A Graphing Utility

- the first two arguments of ANOTAT (axis labels)
- the last argument of EZY, EZXY, EZMY, and EZMXY (graph labels)
- the last argument of ANOTAT (an array of dash patterns)
- the first argument of AGSETF, AGSETI, AGSETP, AGGETF, AGGETI, and AGGETP
- both arguments of the new routines AGSETC and AGGETC, which are used to access (set/reset) the values of AUTOGRAPH control parameters representing character strings.

The FORTRAN-77 function reference "CHAR(0)" (the character whose integer equivalent is zero) or any character string beginning with that character may be used in calls to ANOTAT, EZY, EZXY, EZMY, and EZMXY to specify that no change is to be made in a current x-axis label, y-axis label, or graph label. On the CRAY-1 machines, a real or integer zero will still work, but should be avoided.

If the length of a character string defining a line of a label can be properly computed by use of the FORTRAN-77 function LEN, the character string need not have a 'LINE/END.' character (a "$", by default) following the last character of the label proper; if desired, however, that character may still be used. For example,

```fortran
CALL EZY (YDRA,NPTS,'DESIRED GRAPH LABEL')
```

will work and so will

```fortran
CHARACTER*19 GLAB
...
GLAB='DESIRED GRAPH LABEL'
CALL EZY (YDRA,NPTS,GLAB)
```

On the other hand, the code
would not work properly. AUTOGRAPH will conclude that the label is 40 characters long and won't center it properly. In this case, a $ is needed following the word LABEL. Of course, the $ may always be used.

Dash patterns defined by a call to ANOTAT may no longer be 16-bit binary quantities; character strings must be used. Lower-level calls do allow one to define 16-bit binary patterns. The nature of the lower-level calls to define dash patterns has changed completely and it has become necessary to impose a limit on the number of such dash patterns defined. For reasons of symmetry, 26 was the number chosen.

Effects of Losing the LOC Function

The function LOC, which returned the memory address of its argument, was non-standard and difficult or impossible to implement on some machines. It was used by AUTOGRAPH to recreate the address of a user-defined entity (either a character string, to be drawn, or an element in an array of character strings, to be used as a dash pattern, or a subroutine, to be called). It cannot be used by the FORTRAN-77 AUTOGRAPH. This creates the following problems:

The parameter-accessing scheme provided by AUTOGRAPH demands that the control parameters of AUTOGRAPH in the labelled common block AGCONP all be of type REAL. This poses no problem for parameters whose values are intrinsically real or integral, since the latter can be represented by an equivalent real number. Parameters which are intrinsically character strings, however, do present a problem. In the old AUTOGRAPH, this problem was solved by making the value of the control parameter be the real equivalent of the integer memory address of the variable containing the character string (as returned by the LOC function). When AUTOGRAPH needed to access the character string (which remained in the user's array), it could do so easily, using the LOC function to recreate the address of the array. This had the desirable effect of allowing the user to define a character string of any desired length without running into built-in limits. A new scheme must now be used. A character string passed to
AUTOGRAPH: A Graphing Utility

AUTOGRAPH by the user (for example, the text of a line label) is stashed in an internal array and assigned an identification number. The associated real control parameter preserves this number, so that AUTOGRAPH may use it to retrieve the character string when that becomes necessary (for example, when the label is to be drawn). This scheme has the disadvantage of imposing limits on the lengths of character strings used and on the total number of characters which can be used at one time; I see no standard-conforming way around that. A user who runs into one of these limits may need to increase the size of the array CHRA (which holds 2000 characters by default) in order to run; if this becomes necessary, contact the Consulting Office at (303) 497-1278.

The user-system-to-label-system mapping on each of the four axes may be modified by the user. With the old AUTOGRAPH, the user supplied the address of a subroutine defining the desired mapping by using the LOC function. The real equivalent of this address was saved as the value of the AUTOGRAPH control parameter 'AXIS/s/FUNCTION.' and then used at the appropriate time to recreate the address of the routine to be called. The new AUTOGRAPH requires a different scheme: a routine called AGUTOL does the mapping for all four axes; the user may replace the default routine with one which will do the desired mappings. This provides all the capabilities of the old scheme, but does require changes in the code.

Required Changes in Calls to EZ... Routines

The last argument in a call to EZY, EZXY, EXMY, and EXMXY must now be a character string. The values "0" and "0." should not be used; the character string "CHAR(0)" indicates that no change is to be made in the current graph label. Any other character string defines a new graph label. Such a string must either be of the exact length defined by the control parameter 'LINE/MAXIMUM.' (default - 40 characters) or shorter; if it is shorter, and if its length is not computable by AUTOGRAPH using the function LEN, the string should be terminated by the character defined by the control parameter 'LINE/END.' (default - a "$").

Required Changes in Calls to ANOTAT

The first two arguments of ANOTAT (defining new x- and y-axis labels) must now be character strings. The values "0" and "0." should not be used for either of these arguments; the character string "CHAR(0)" indicates that no
change is to be made in a current label. Any other character string defines a new label. Such a string must either be of the exact length defined by the control parameter 'LINE/MAXIMUM.' (default - 40 characters) or shorter; if it is shorter, and if its length is not computable by AUTOGRAPH using the function LEN, the string should be terminated by the character defined by the control parameter 'LINE/END.' (default - a "$").

The last argument of ANOTAT must be an array of character strings, defined by a statement of the form

\[
\text{CHARACTER}^m \text{ DSHC}(n)
\]

where \( m \) is greater than or equal to the value of the control parameter 'DASH/LENGTH.', and \( n \) is greater than or equal to the fifth argument of the ANOTAT call (IDSH, the number of dash patterns); \( n \) must be less than or equal to 26. If IDSH is zero, DSHC will be ignored, but, according to FORTRAN rules, should still be a character-string array.

Current calls to ANOTAT with the final argument an array of binary dash patterns must be modified. Consider the following code fragment:

```fortran
DIMENSION DSHL(3)
DATA DSHL / 177777B,125252B,114631B /
CALL ANOTAT (...,3,DSHL)
```

It may be replaced by the following:

```fortran
CHARACTER*16 DSHP(3)
DATA DSHP / '$$$$$$$$$$$$$$$$',
+ '$$$$$$'
+ '$$$$$$$$$$$$$$$$' /
CALL AGSETI ('DASH/LENGTH.',16)
CALL ANOTAT (...,3,DSHP)
```

Note that, in dash patterns 2 and 3, each apostrophe which is to be a part of the dash pattern must be written as two apostrophes. On the CRAY-1 machines, one can avoid this by using quotation marks to enclose the character string, but the result is not FORTRAN-77 standard.
Note that the parameter 'DASH/LENGTH.' must be set before ANOTAT is called, to specify how long the dash patterns are. Note also that the dash patterns are defined by character strings. To save the user some work, the following code may be used instead:

```fortran
INTEGER DSHL(3)
DATA DSHL / 177777B,125252B,114631B /
CHARACTER*16 AGBNCH
CHARACTER*16 DSHP(3)
...
DO 101 I=1,3
   DSHP(I)=AGBNCH(DSHL(I))
101 CONTINUE
CALL AGSETI ('DASH/LENGTH.',16)
CALL ANOTAT (...3,DSHP)
```

This is similar, except that the function AGBNCH is used to create the character-string dash patterns from the 16-bit binary patterns. AGBNCH is an AUTOGRAPH function of type CHARACTER*16 whose value, given a 16-bit integer dash pattern, is the character-string equivalent of that pattern. Another way is shown below:

```fortran
INTEGER DSHL(3)
DATA DSHL / 177777B,125252B,114631B /
CHARACTER*16 AGDSHN
...
DO 101 I=1,3
   CALL AGSETI (AGDSHN(I),DSHL(I))
101 CONTINUE
CALL AGSETI ('DASH/SELECTOR.',3)
```

In this case, ANOTAT is by-passed completely; the dash patterns are defined for AUTOGRAPH as 16-bit binary integers. (AGDSHN is an AUTOGRAPH function of type CHARACTER*16 whose value, given an integer n between 1 and 26, inclusive, is the name of the control parameter defining the nth dash pattern.) The above code therefore defines the first three dash patterns to have the required values, and then, by setting 'DASH/SELECTOR.', tells AUTOGRAPH to use that set of dash patterns.
Appendix A

Required Changes in Calls to DISPLA

Calls to DISPLA need not be changed - the three arguments are just as they always were.

Required Changes in Calls to AGGETx and AGSETx

The first argument in a call to a parameter-setting/retrieving routine must be a FORTRAN-77 character expression. Hollerith constants may be used on the CRAY-1 machines for the moment, but their continued use cannot be guaranteed.

Any call whose purpose was to set or retrieve parameters whose definition has changed will need to be changed appropriately. Parameter names involved are 'PRIMARY.', '[AXIS]/s/FUNCTION.', 'DASH/...', 'LABEL/NAME.', 'LINE/END.', and 'LINE/[DEFINITION]/TEXT.'. Subsequent sections discuss specific functional areas which will involve changes in parameter setting and retrieving calls.

Calls to AGGETP/AGSETP with the first argument 'PRIMARY.' were most likely intended to save/restore the state of AUTOGRAPH. For this purpose, the new routines AGSAVE and AGRSTR should be used instead.

An AGSETI call to set the parameter '[AXIS]/s/FUNCTION.' (for some "s") will have to be changed; the value of this parameter has been changed from the address of a subroutine to do the user-system-to-label-system mapping to a mapping selector.

Any call to set/retrieve a parameter whose name is of the form 'DASH/...' will have to be changed since all such parameters have been changed.

An AGSETF call to set either of the parameters 'LABEL/NAME.' or 'LINE/END.' must be changed to call AGSETC instead. This will set character-valued control parameters. For example, the call

```
CALL AGSETF ('LABEL/NAME.', 'L')
```

must be changed to

```
CALL AGSETP ('LABEL/NAME.', 'L')
```
Similarly, an AGSETP call to set the parameter 'LINE/TEXT.' must be changed to call AGSETC. For example, the call

```plaintext
CALL AGSETP ('LINE/TEXT.', 'THE TEXT OF A LABEL$', 1)
```

must be changed to

```plaintext
CALL AGSETC ('LINE/TEXT.', 'THE TEXT OF A LABEL$')
```

Note the absence of a third argument in the call to AGSETC.

Required Changes in Calls to AGSTUP, AGBACK, and AGCURV

Calls to AGSTUP, AGBACK, and AGCURV need not be changed.

Saving/Restoring the State of AUTOGRAPH

The new subroutines AGSAVE and AGRSTR should be used to save/restore the state of AUTOGRAPH. Each has a single argument – the number of a unit to which control parameters and character-string information may be written or from which they may be read. The user is responsible for positioning of this unit. For example, to save the default state of AUTOGRAPH on unit 19, execute the call

```plaintext
CALL AGSAVE (19)
```

before doing anything else. Later, to restore the default state of AUTOGRAPH, use the code

```plaintext
REWIND 19
CALL AGRSTR (19)
```

Old code using AGSETP/AGGETP to save/restore the parameter group 'PRIMARY.' should be replaced; it will not properly handle character-string information.

Defining DASH Patterns

The control-parameter group 'DASH.' originally contained the following parameters: 'DASH/ADDRESS.', which specified the address of an array of
"user" dash patterns; 'DASH/NUMBER.', which specified how many patterns there were in the array; 'DASH/INDEX.', which specified the index increment to get from one pattern in the array to the next; 'DASH/LENGTH.', which specified how many characters there were in each dash pattern; and 'DASH/CHARACTER.' and 'DASH/DOLLAR-QUOTE.', which specified the size of characters to be used. There was also a parameter called 'DASH/SELECTOR.', which specified whether the "alphabetic" set of dash patterns or the "user" set of dash patterns were to be used by EZMY and EZMXY (but was not, despite its name, a member of the group 'DASH.'). Code to define a set of 12-character dash patterns might have looked like this:

```
DIMENSION DSHP(2,3)
DATA DSHP / 8H$$$$$$$$, 4H$$$$,
+ 8H$'$'$'$', 4H$'$',
+ 8H$$''$$'', 4H$$'''' /
CALL AGSETI ('DASH/SELECTOR.',1)
CALL AGSETI ('DASH/ADDRESS.',LOC(DSHP))
CALL AGSETI ('DASH/NUMBER.',3)
CALL AGSETI ('DASH/INDEX.',2)
CALL AGSETI ('DASH/LENGTH.',12)
```

The new group 'DASH.' is quite different. It includes the following parameters: 'DASH/SELECTOR.', which both selects the set of dash patterns to be used and, if positive, specifies how many different dash patterns are to be used; 'DASH/LENGTH.', which specifies how many characters there are in each dash pattern; 'DASH/CHARACTER.' and 'DASH/DOLLAR-QUOTE.', which are as before; and the parameters 'DASH/PATTERN/1.', 'DASH/PATTERN/2.', and so on, through 'DASH/PATTERN/26.'. To achieve the effect of the old code above, the following new code would suffice:
AUTOGRAPH: A Graphing Utility

CHARACTER*12 DSHP(3)
DATA DSHP / '$$$$$$$$$$' ,
+ '$'$'$'$'$'$'$'$'$' ,
+ '$$'$'$'$'$'$'$'$' /

CALL AGSETI ('DASH/SELECTOR. ',3)
CALL AGSETI ('DASH/LENGTH. ',12)
CALL AGSETIC ('DASH/PATTERN/1. ',DSHP(1))
CALL AGSETIC ('DASH/PATTERN/2. ',DSHP(2))
CALL AGSETIC ('DASH/PATTERN/3. ',DSHP(3))

The last three lines may be rewritten as

DO 101 I=1,3
   CALL AGSETIC (AGDSHN(I),DSHP(I))
101 CONTINUE

but then the declaration

CHARACTER*16 AGDSHN

must be added to the code.

Note that, at most, 26 "user" dash patterns may be defined. They need not be character-string patterns. The statement

CALL AGSETF ('DASH/PATTERN/12. ',65535.)

or (on the CRAY-1 computers at least) the statement

CALL AGSETI ('DASH/PATTERN/12. ',177777B)

may be used to specify a binary pattern for the twelfth "user" dash pattern.

The new scheme is actually simpler than the old, but it imposes limits that the old one did not. Character strings used must not be longer than 504 characters each. Since strings are moved to storage locations inside AUTOGRAPH, there is the possibility of overflowing the 2000-position storage space for them. At most 26 different dash patterns may be defined.
Defining the Text of LABEL Lines

The old way to define the text of a line of a label used code like this:

```fortran
CALL AGSETF ('LABEL/NAME.', 'L')
CALL AGSETI ('LINE/NUMBER.', 100)
CALL AGSETP ('LINE/TEXT.', 'THE TEXT OF THE LABEL$', 1)
```

The new way looks like this:

```fortran
CALL AGSETC ('LABEL/NAME.', 'L')
CALL AGSETI ('LINE/NUMBER.', 100)
CALL AGSETC ('LINE/TEXT.', 'THE TEXT OF THE LABEL$')
```

User-System-to-LABEL-System Mapping

To change the user-system-to-label-system mapping on a given axis, a user of the old AUTOGRAPH wrote a routine - call it USRFUN - to do the mapping and provided AUTOGRAPH with the address of that routine by means of code like:

```fortran
EXTERNAL USRFUN

SUBROUTINE USRFUN (IDMA, VINP, VOTP)
  IF (IDMA.GT.0) VOTP=ALOG10(20.-VINP)
  IF (IDMA.LT.0) VOTP=20.-10.**VINP
  RETURN
END
```

To change the mapping on another axis, similar code was required. In the new scheme, the mapping for all four axes is done by an AUTOGRAPH routine called AGUTOL, with arguments IAXS, FUNS, IDMA, VINP, and VOTP. IAXS is a 1 if the left axis is being drawn, a 2 for the right axis, a 3 for the bottom axis, and a 4 for the top axis. FUNS is the value of 'AXIS/s/FUNCTION.'. The other arguments are as before. Thus, to change the mapping for both the left and bottom axes, the new code looks like:

```fortran
SUBROUTINE AGUTOL (IAXS, FUNS, IDMA, VINP, VOTP)
  IF (IAXS.EQ.1) VOTP=AGUTOL(2, FUNS, IDMA, VINP, VOTP)
  IF (IAXS.EQ.3) VOTP=AGUTOL(4, FUNS, IDMA, VINP, VOTP)
  RETURN
END
```
CALL AGSETF ('LEFT/FUNCTION.',1.)
CALL AGSETF ('BOTTOM/FUNCTION.',1.)

SUBROUTINE AGUTOL (IAXS,FUNS,IDMA,VINP,VOTP)
IF (IAXS.EQ.1.AND.FUNS.EQ.1.) THEN
  IF (IDMA.GT.0) VOTP=ALOG10(20.-VINP)
  IF (IDMA.LT.0) VOTP=20.-10.**VINP
ELSE IF (IAXS.EQ.3.AND.FUNS.EQ.1.) THEN
  IF (IDMA.GT.0) VOTP=70.136*VINP
  IF (IDMA.LT.0) VOTP=VINP/70.136
ELSE
  VOTP=VINP
END IF
RETURN
END

New Features and New Routines

The functions AGBNCH and AGDSHN are intended to make it easier for the user to define dash patterns. They have been described above.

The subroutine AGUTOL may be replaced by the user to change the user-system-to-label-system mappings on one or more axes. It has been described above.

With the advent of GKS and the possible acquisition of graphics devices with color capability, the number of user-changeable attributes of the various portions of a graph is increasing. The AUTOGRAPH user will need some way to specify these attributes. Rather than add a whole new set of control parameters to an already huge list, a new tack has been taken. Just before drawing an object, and again just after drawing that object, AUTOGRAPH calls one of three routines. The default versions of these routines do nothing; the user may supply his own versions, within which attributes may be set and reset as desired. The three routines are AGCHAX, which handles the various objects which together make up an axis, AGCHCU, which handles curves, and AGCHIL, which handles informational labels. The arguments with which these routines are called will be described in the new AUTOGRAPH write-ups.
There is also a new routine called AGCHNL, which is called just after the character form of a numeric label has been constructed and just before it is to be drawn. The user may supply a version of this routine to transform selected numeric labels in any desired fashion and return them to AUTOGRAPH. This feature may be used to label an axis with the names of the months, Roman numerals, and so on. The arguments with which this routine is called will also be described in the new AUTOGRAPH write-ups.

The new AUTOGRAPH does not call PWRIT directly at the points where it wishes to draw a character string. Instead, it calls a routine AGPWRT, whose arguments are exactly the same as those of PWRIT. AGPWRT, in turn, calls PWRIT with those arguments. Thus, to substitute a fancier character-drawing routine, one need only compile one's own version of AGPWRT; there is no need to modify the AUTOGRAPH source code. The XLIB file AGUPWRTX contains a version of AGPWRT which makes it possible to use PWRITX with AUTOGRAPH. The results are superior to those obtained with the obsolete XLIB file AGUSEPWRX.

by Dave Kennison