SUN/90.10

Starlink Project
Starlink User Note 90.10

P C T Rees, M J Bly, P T Wallace

7 March 1995

SNX — Starlink Extensions to the NCAR Graphics Utilities
Version 1.1-1
Programmer's Manual

Contents

1 Introduction

The NCAR graphics suite (SUN/88) consists of a set of subprograms which can be used to produce complete graphs in a variety of formats. The package has been in wide use for some years; the latest version employs the ISO standard GKS interfaces for its low-level plotting, giving it access to all Starlink graphics devices, present and future.

The NCAR routines themselves are thoroughly documented, and just a few simple calls will produce graphs of excellent appearance. The package also provides a high level of flexibility, with dozens of different details of the plot independently controllable to give exactly the result required. However, beginners may be daunted by the mass of features offered, and unless they take the extreme step of reading the manual may give up before they realise what the package can do for them. This document describes minor extensions which provide more convenient access to certain features without sacrificing flexibility.

The AUTOGRAPH part of the NCAR suite, used in conjunction with the Starlink NCAR extensions and the Starlink low level plotting package SGS (SUN/85), offers an alternative high level system to PGPLOT (SUN/15) for producing graphs of one variable plotted against another. All of the Starlink extensions provided within SNX enhance the power of the facilities provided by AUTOGRAPH and make it more accessible to the beginner.

Here are some of the things you can do using the facilities described in this document:

- Open AUTOGRAPH/SGS/GKS with a single call.
- Plot a finished graph, with annotation, by means of a single call.
- Include special characters in graph labels (Greek, subscripts, mathematical symbols, *etc.*).
- Plot character strings anywhere on the graph.
- Manage the graphics device display surface via the SGS system of zones while using AUTOGRAPH to do the plotting.
- Align the AUTOGRAPH and SGS coordinate systems.
- Save and restore the state of AUTOGRAPH so that multiple active plots can be managed.
- Perform cursor input.

See Appendices C–E for example programs demonstrating some of these facilities.

2 Summary of Calls

CALL SNX_AGOP
Make AUTOGRAPH/SGS/GKS ready for plotting.

CALL SNX_EZRXY (XDRA, YDRA, NPTS, XLAB, YLAB, GLAB)
Plot (x,y) data, with labelling, using AUTOGRAPH.

CALL SNX_AGLAB (LNAME, TEXT)
Set up an AUTOGRAPH label.

CALL SNX_CHSET (I)
Select one of the two NCAR Roman fonts.

CALL SNX_AGSAV (HEAP)
Save the state of AUTOGRAPH.

CALL SNX_AGRES (HEAP)
Restore the state of AUTOGRAPH.

CALL SNX_TO (SORN)
Switch between NCAR and SGS plotting.

CALL SNX_AGWV
Make the AUTOGRAPH graph window match the current viewport (= zone if using SGS).

CALL SNX_AGCS
Make the current SGS zone world coordinates match the AUTOGRAPH grid coordinate system.

CALL SNX_CURS (X, Y, N)
Read a cursor position.

UX = **SNX_AGGUX** (GX)
Transform grid X coordinate to user X coordinate.

UY = **SNX_AGGUY** (GY)
Transform grid Y coordinate to user Y coordinate.

GX = **SNX_AGUGX** (UX)
Transform user X coordinate to grid X coordinate.

GY = **SNX_AGUGY** (UY)
Transform user Y coordinate to grid Y coordinate.

CALL SNX_WRTST (XG, YG, STRING, HG, IOR, ICTR)
Plot a character string.

3 Linking

3.1 Use with the UNIX operating system

Assuming all Starlink directories have been added to the environment variables `PATH` and `LD_LIBRARY_PATH` (see SUN/118), then to link a non-ADAM program with SNX and NCAR the command line would be:

```
% f77 program.o -L/star/lib ‘snx_link‘ -o program.out
```

This command line will link with the SNX, NCAR and SGS libraries.

Two subroutines provided by SNX exist in other forms elsewhere and require special handling. The first gives access to the special fonts provided by the NCAR routine PWRITX: it is a Starlink version of the NCAR subroutine AGPWRT. The second intercepts and changes the numeric labels produced by AUTOGRAPH, turning “.1" and “−.5" into “0.1" and “−0.5" *etc.*: it is a Starlink version of the NCAR subroutine AGCHNL. These routines will be automatically linked by using ‘`snx_link`‘ if you have called the SNX routine `SNX_AGOP`. If you are using SNX routines as part of an NCAR/SGS mix, and have not called `SNX_AGOP` to initialise the graphics system, you must link with the objects directly:

```
% f77 program.o /star/lib/agpwritx.o /star/lib/agchnlz.o \
       -L/star/lib ‘snx_link‘ -o program.out
```

(It is important to realise that the use of the special fonts will naturally incur a speed penalty. Advice on this is given in §7.)

Compiling and linking ADAM applications is discussed in Appendix A.

3.2 Use with the VAX/VMS operating system

Non-ADAM programs may be linked with the SNX and NCAR libraries by

```
$ LINK program, NCAR_DIR:SNX_LINK/OPT, STAR_LINK/OPT
```

This command line will link with the SNX, NCAR and SGS libraries.

Compiling and linking ADAM applications is discussed in Appendix A.

4 Initialising AUTOGRAPH/SGS/GKS

To make AUTOGRAPH/SGS/GKS ready for plotting, use the Fortran statement

```
CALL SNX_AGOP
```

The routine asks for the SGS/GKS workstation name via the standard input device used by NCAR (normally the terminal). If <CR> or an invalid name is entered, a list of all the current names is displayed and the request is repeated. Once SGS/GKS and the workstation have all been successfully opened, AUTOGRAPH is configured to use the full display surface (in other words the base SGS zone). If you don't want the AUTOGRAPH plot to fill the display surface, use the SGS zone creation routines to select the desired region and then call SNX_AGWV (see §9).

5 Plotting X against Y with AUTOGRAPH

To plot a labelled graph of the curve joining (x,y) points given as separate x and y arrays, the following can be used:

```
CALL SNX_EZRXY( XDRA, YDRA, NPTS, XLAB, YLAB, GLAB )
```

XDRA and YDRA are real 1-dimensional arrays containing the x and y points respectively; the integer NPTS is the number of (x,y) values; XLAB, YLAB and GLAB are character strings giving the three labels (x-axis, y-axis and graph). Trailing blanks in the labels are ignored. Greek letters and special symbols can be used by including PWRITX function codes in the strings (see Appendix F) by virtue of linking with the Starlink version of AGPWRT (§3 and §7).

When using SNX, the format of numeric labels is "0.4", "−0.25" instead of the default style used by AUTOGRAPH, ".4", "−.25", by virtue of linking with the Starlink version of AGCHNL (§3). Other control over labelling (character sizes, multiple lines, character orientation, *etc.*) can be achieved by setting the appropriate AUTOGRAPH parameters and then calling the NCAR routine EZXY directly, and by writing bespoke versions of the user-replaceable routines AGCHNL and AGUTOL.

NCAR can plot other forms of data: *e.g.* y array only, multiple curves, *etc.* These are discussed in the NCAR AUTOGRAPH document – see EZY, EZMY, EZMXY.

Appendices C–E contain several examples of the use of SNX_EZRXY.

6 AUTOGRAPH Labels

To set up one of the four predefined AUTOGRAPH labels (see §2.30 *et seq.* in the AUTOGRAPH document), the following call may be used:

```
CALL SNX_AGLAB( LNAME, TEXT )
```

LNAME and TEXT are both character strings. The first character of LNAME is the label name ('R', 'L', 'B' or 'T' for right, left, bottom, top respectively) and TEXT is the label text. Trailing

blanks are ignored so that the text proper is correctly centred – for deliberate trailing blanks, use '$' as an endmark. Also, PWRITX function codes can be inserted for special characters (see §3 and §7). For LNAME = 'T' or 'L', the AUTOGRAPH line number is set to 100; for 'B' or 'R', the line number is −100.

Illegal LNAME values cause the top label to be set to

```
'*** AGLAB LABEL ERROR ***'.
```

The SNX_AGLAB routine can be used in conjunction with the AUTOGRAPH facilities. For example, the following calls will create a 2-line 'B' label (beneath the x-axis), with the second line (numbered −200) of character height 0.03 grid units. Here, one grid unit is the height of the box within which the curve is plotted.

```
CALL SNX_AGLAB( 'B', 'This is the first line of text' )
CALL AGSETC( 'LABEL/NAME.', 'B' )
CALL AGSETI( 'LINE/NUMBER.', -200 )
CALL AGSETF( 'LINE/CHARACTER.', 0.03 )
CALL AGSETC( 'LINE/TEXT.', 'This is the second line of text' )
```

It is permissible to omit the AGSETC call which specifies the 'B' label as this will have been done already by SNX_AGLAB.

7 Control over Text Font and Character Set

An ordinary AUTOGRAPH application program linked only with the NCAR library, *e.g.*

```
$ LINK PROGRAM, NCAR_DIR:NCAR_LINK/OPT, STAR_LINK/OPT
```

on VAX/VMS, will include the standard version of the routine AGPWRT which uses the GKS text-drawing facilities. Extremely flexible control over font, colour, *etc.*, is available via GKS (see §2.43 of the AUTOGRAPH document), but NCAR itself contains some facilities for controlling the text font which many users will find more accessible, particularly when Greek letters, subscripts and other special characters are to be included in labels.

To use the NCAR fancy fonts, the application program must be linked with a special Starlink version of the AUTOGRAPH routine AGPWRT, which has access to the NCAR PWRITX character drawing routine (§3). When using the Starlink version of AGPWRT, PWRITX function codes can be included in the character strings in order to switch between various sets of special characters. Lower case characters and common punctuation marks are automatically handled; apostrophe (which is used as the delimiter for the PWRITX codes) is specified by two consecutive apostrophes. A choice of two native NCAR character sets is available. The selection can be made as follows:

```
CALL SNX_CHSET( I )
```

The integer argument I may be either 1 or 2. I=1 selects the “duplex" font set and I=2 the “complex" font. The “complex" characters (the default) are more ornate and fussier than the comparatively plain “duplex" characters. Only the Roman characters are changed by the SNX_CHSET call – not the Greek, *etc.*

For precise details of how to construct the PWRITX strings, refer to Appendix F, which includes the program preamble comments describing the AGPWRITX routine, followed by a plot of the available characters. The principle is that within the string of ordinary characters, function codes can be embedded describing changes of font, size and position. The function codes are delimited by apostrophes. For example, a label meaning r^2 could be written using the string “`r'S'2'N'`". The “`r`" is simply drawn as a lower case “r"; the “`'S'`" is a function code switching to superscript; the “2" is then drawn as a small numeral in the superscript position; and finally the “`'N'`" switches back from superscript to normal. Though daunting at first, the strings are quite easy to construct and to debug by trial and error.

As a more complicated example, the string “`Area = 'GL'P'R'r'S'2`" will generate a label equivalent to “Area $= \pi r^2$". (The apostrophes make for messy Fortran literals; this one would be “`'Area = ''GL''P''R''r''S''2''`".) The first PWRITX function code ‘GL’ selects the Greek Lowercase font; P happens to be the source character which generates the pi; the next function code ‘R’ switches back to Roman for the r; the final code ‘S’ means superscript and draws the power of 2 symbol. More examples are included in the XYPLOT input file listed in Appendix C.

Inevitably, use of the special fonts costs plotting time – each character is formed from many line segments, all of which have to be transmitted to the device. This is compounded by the relatively high density of numeric labelling which AUTOGRAPH provides by default. This speed penalty is an acceptable cost where a publication-quality plot is being produced on a hardcopy device, but not for interactive applications requiring frequent and rapid plotting. Appendix D describes a demonstration program which switches between three different character drawing methods and graph styles to achieve different trade-offs between speed and appearance. This involves a special form of the call to AGPWRT, available only with the AGPWRITX implementation of this routine. The call is:

```
CALL AGPWRT( 0.0, 0.0, ' ', 0, 0, 0, -100 )
```

or

```
CALL AGPWRT( 0.0, 0.0, ' ', 0, 0, 0, 100 )
```

(All the arguments except the last one, ICEN, which must be ± 100, are ignored.) The -100 ICEN value selects character drawing via the ordinary PWRIT routine, which uses the GKS character drawing facilities and thus allows control of font, precision, *etc.* The +100 ICEN value selects character drawing via the fancy PWRITX routine.

8 AUTOGRAPH Save and Restore

It can sometimes be useful to draw several graphs on one display frame and then to plot additional material to (or perform input from) one or more of them later. This can be accomplished by using the routines SNX_AGSAV and SNX_AGRES, which respectively save and restore the AUTOGRAPH context. Thus the sequence:

```
Plot something with AUTOGRAPH
CALL SNX_AGSAV( HEAP )
Plot something else with AUTOGRAPH
CALL SNX_AGRES( HEAP )
```

will leave AUTOGRAPH in the same state as it was after the first plot. HEAP is a 1-dimensional real array in which SNX_AGSAV stores all of AUTOGRAPH's important variables. HEAP must currently be at least 2606 elements long; check the SNX_AGSAV source for the latest value. If you wish to save several different AUTOGRAPH states in one program, several HEAP arrays will of course be required.

When using this facility it will very often be necessary to switch SGS zone (with SGS_SELZ) as well as saving and restoring AUTOGRAPH. When this is done, a call to SNX_AGWV (§9.1) will be needed as well. See also §11.

The NCAR software includes its own AUTOGRAPH save/restore facility (AGSAVE, AGRSTR). This uses a scratch file, which means that a state saved in one program can be restored in another, but does not save the plotting context. It is thus most useful for plotting successive data sets with the same AUTOGRAPH parameters rather than (for example) making additions to a graph plotted earlier, something which is for that matter hard to do with GKS itself.

9 AUTOGRAPH and SGS Coordinate Systems

9.1 Making AUTOGRAPH use the current SGS zone

AUTOGRAPH can be made to plot within the current GKS viewport as follows:

```
CALL SNX_AGWV
```

This will cause the AUTOGRAPH graph window (the rectangle which contains the labels, axes, grid, *etc.*) to be coincident with the GKS viewport. A specially convenient way to set up a GKS viewport is by means of SGS; a call to SNX_AGWV following any zone creation or selection call will cause AUTOGRAPH to arrange its plot within that zone. The zone may be of any aspect ratio, with the proviso that labels may overflow very narrow ones.

9.2 Setting the SGS coordinates to match an AUTOGRAPH plot

Once AUTOGRAPH has established its plotting strategy (usually after drawing the graph, or either the background or curve) it may be necessary to make additions to the plot or to perform input. This can be done conveniently as follows:

```
Plot the graph with AUTOGRAPH
CALL SNX_AGCS
```

The call to SNX_AGCS sets the extent (in world coordinates) of the current SGS zone, presumed to cover the AUTOGRAPH graph window (the rectangular area within which the whole AUTOGRAPH plot, including labels *etc.*, is drawn), so that the AUTOGRAPH grid window (the

rectangular area within which the curve itself is plotted) has bounds (0.0,1.0,0.0,1.0). Thus the world coordinates of the SGS zone are now AUTOGRAPH grid coordinates. Clipping requirements may then make it necessary to create another zone covering just the grid window, but still with grid coordinates. This is easily accomplished by:

```
CALL SGS_ZONE( 0.0, 1.0, 0.0, 1.0, IZGW, J )
CALL SGS_SW( 0.0, 1.0, 0.0, 1.0, J )
```

9.3 SGS/AUTOGRAPH coordinate conversions

For data-related plotting and input, four functions are provided to perform the necessary coordinate conversions. The two sets of coordinates are respectively the grid coordinates (GX,GY) and the user (*i.e.* data) coordinates (UX,UY).

To convert a grid x coordinate to a user x coordinate:

```
UX = SNX_AGGUX( GX )
```

To convert a grid y coordinate to a user y coordinate:

```
UY = SNX_AGGUY( GY )
```

To convert a user x coordinate to a grid x coordinate:

```
GX = SNX_AGUGX( UX )
```

To convert a user y coordinate to a grid y coordinate:

```
GY = SNX_AGUGY( UY )
```

The grid coordinates are such that the grid window part of the AUTOGRAPH graph window has bounds (0.0,1.0,0.0,1.0); they match the world coordinates for a zone set up with SNX_AGCS (see the previous section).

The user coordinates relate to the actual data values supplied, even though the plotting may have been done with logarithmic scaling or reversed directions.

10 Plotting Character Strings

AUTOGRAPH provides for multiline labels in one of several predefined regions of the plot, but can be awkward to use for placing character strings in arbitrary places. The SPPS routine WTSTR is more suitable for this purpose, but operates in user (*i.e.* data) coordinates, not appropriate for plotting outside the grid window, and expresses the character size in low level “plotter units". Character strings can, of course, be plotted directly through GKS or SGS, though access to the special NCAR fonts will not be available.

The Starlink extension routine SNX_WTRST is a further method of plotting strings. The call is as follows:

```
CALL SNX_WRTST( XG, YG, STRING, HG, JOR, JCTR )
```

XG,YG (real) is the position of the string in grid coordinates. STRING is the character string to be plotted, and may contain PWRITX codes (see §7) if special characters are required. This option requires linking with the AGPWRITX version of AGPWRT (see §3). HG (real) is the character height in grid Y units. JOR (integer) is the string orientation, in degrees anticlockwise from the usual left-to-right. JCTR (integer) describes the justification; -2 and -1 = left, 0 = centred, +1 and +2 = right, where "left" means "the middle of the left edge of the leftmost character is at XG,YG" *etc.* The values 0 and ± 2 produce proportionally-spaced strings and allow fancy characters to be used; ± 1 produces monospaced strings consisting of ordinary letters (including lowercase), numbers and punctuation characters. You would use ± 1 for plotting a table – the AUTOGRAPH numeric labels are plotted using this value. There is no provision for centred monospaced strings.

11 Mixing GKS, SGS and NCAR Plotting

Most plotting packages are designed to be used on their own, and cannot be used in conjunction with any other package. In contrast, SGS has been designed to permit direct use of GKS, and it is also possible, with care, to use the NCAR routines in combination with SGS and GKS very effectively. However, NCAR and SGS/GKS are separate plotting systems and some care is necessary when switching back and forth between them.

Two packaged save/restore facilities exist within SNX to help do this. The most elaborate is the pair of routines SNX_AGSAV and SNX_AGRES (see §8), which are a complete save and restore for AUTOGRAPH. A simpler facility, which performs the necessary buffer flushing operations and saves/restores the SGS zone, is the SNX_TO routine. This is used as follows:

```
Plot with NCAR
CALL SNX_TO( 'SGS' )
Plot with SGS
CALL SNX_TO( 'NCAR' )
More plotting with NCAR
```

Only the first character of the argument is significant. The sequence must begin with the switch to SGS and then alternate strictly.

When you have just plotted something using an NCAR routine (or at least a low level one – the high level utilities do it for you) and wish to switch to SGS or GKS, make the following call:

```
CALL PLOTIT( 0, 0, 2 )
```

or if you are switching from SGS:

```
CALL SGS_FLUSH
```

or from GKS:

```
CALL GUWK( WKID, REGFL )
```

After plotting with AUTOGRAPH, further GKS or SGS work will require the coordinate systems to be re-established. In SGS this can be done as follows:

```
CALL SGS_ICURZ( IZ )
CALL SGS_SELZ( IZ )
```

This can be omitted if the routine SNX_AGCS is being called. Conversely, the NCAR coordinate systems will need to be saved and restored if there is intervening SGS or GKS plotting. This can be done by means of the SPPS routines GETSET and SET.

12 Cursor Input

To input coordinates from a graph plotted with AUTOGRAPH, use the call

```
CALL SNX_CURS( X, Y, N )
```

X, Y (real) are user (*i.e.* data) coordinates. Their value when the call is made is used to preset the cursor position (if this is possible on the GKS workstation concerned). On exit, they are set to the position indicated by the cursor input operation. The integer N is the SGS choice, and will generally indicate the button that has been pressed.

Variations of choice device, cursor visibility, echo type, *etc.*, may be made by direct SGS/GKS calls before SNX_CURS is called.

Appendix E contains a demonstration program for SNX_CURS.

13 References

SGP/16 Starlink Application Programming Standard.
SUN/15 PGPLOT — Graphics Subroutine Library.
SUN/83 GKS — Graphical Kernel System (7.2).
SUN/85 SGS — Simple Graphics System.
SUN/88 NCAR — NCAR – Graphics Utilities.
SUN/118 Starlink Software on UNIX.
SUN/144 ADAM — UNIX Version.

A Using SNX within ADAM Applications

A.1 Use with the UNIX operating system

Assuming all Starlink directories have been added to the environment variables **PATH** and **LD_LIBRARY_PATH** (see SUN/118), then to link an ADAM application with SNX the command line would be

```
% alink application.o ‘snx_link_adam‘
```

SUN/144 gives further details of compiling and linking ADAM applications with the UNIX operating system.

A.2 Use with the VAX/VMS operating system

When developing ADAM applications, all logical names and symbols used during compiling and linking are defined by executing the procedures

```
$ ADAMSTART
$ ADAM_DEV
```

The application may then be compiled using the FORTRAN command and linked using the ALINK and MLINK procedures, *e.g.*

```
$ ALINK application,NCAR_DIR:SNX_LINK_ADAM/OPT
```

B Portability

B.1 Overview

This section discusses the portability of SNX, including the coding standard adopted and a list of those Starlink packages which need to be ported to the target machine before a port of SNX can proceed.

B.2 Coding and porting prerequisites

The standard of Fortran used for the coding of SNX is fundamentally Fortran 77, using the Starlink Fortran coding conventions described in SGP/16. Several common extensions to the Fortran 77 standard are used in the SNX source code; they are as follows:

- symbolic subprogram names may be longer than six characters (but are always shorter than ten characters);

- symbolic subprogram names include the "_" symbol;
- the full ASCII character set is assumed in character constants.

There are no operating system-specific routines in SNX. SNX requires NCAR, SGS and GKS Vn. 7.2 or GKS Vn. 7.4 to be available.

C Demonstration – the XYPLOT Program

The following pages contain a listing of the Fortran source for a program called XYPLOT, plus a sample input file and the plot it produces. As well as demonstrating several of the facilities described in this document, XYPLOT may be useful as a template for developing other simple graphics utilities, and can be used in its own right as a convenient way of producing a presentable plot of a list of (x,y) data. The Fortran source may be found in the file NCAR_DIR:XYPLOT.FOR on VAX/VMS and in the archive `/star/starlink/lib/snx/examples/snx-examples_source.tar` on UNIX. On VAX/VMS, this program may be compiled, linked and run using the command sequence

```
$ FORTRAN NCAR_DIR:XYPLOT
$ LINK XYPLOT, NCAR_DIR:SNX_LINK/OPT, STAR_LINK/OPT
$ RUN XYPLOT
```

On UNIX, this program may already be installed in directory `/star/bin/examples/snx`. If it has been deinstalled and removed to save space, you can copy the entire source directory to a scratch dircetory, and with the SYSTEM environment variable set appropriately, build and run it thus:

```
% setenv SYSTEM alpha_OSF/1
% ./mk xyplot
% ./xyplot
```

XYPLOT reads an input file and plots a graph. The input file consists of a sequence of alphanumeric records, the first three of which are the labels (title, x-axis, y-axis) with the rest each containing an (x,y) point (in free format). The example data may be found in the file NCAR_DIR:XYPLOT.DAT on VAX/VMS and `/star/bin/examples/snx/xyplot.dat` on UNIX (if installed). The resulting graph consists of a line joining the points in the order in which they are given.

C.1 Fortran source

```
      PROGRAM XYPLOT
*+
*     - - - - - - -
*      X Y P L O T
*     - - - - - - -
```

```
*
*   Plot xy data from a file using NCAR package.
*
*   Input is from LU 1.  The format is:
*
*           graph label text
*           x label text
*           y label text
*           x y
*           x y
*           x y
*            :
*            :
*
*   P T Wallace   Starlink   May 1987
*
*+
      IMPLICIT NONE

      INTEGER N,NP,NPMAX
      PARAMETER (NPMAX=10000)
      CHARACTER FNAME*80,GLAB*80,XLAB*80,YLAB*80
      REAL X(NPMAX),Y(NPMAX)


*  Open input file
      PRINT *,'Filename?'
      READ (*,'(Q,A)') N,FNAME
      OPEN (UNIT=1,STATUS='OLD',FILE=FNAME(:N),READONLY)

*  Read label text
      READ (1,'(A)',END=100) GLAB
      READ (1,'(A)',END=100) XLAB
      READ (1,'(A)',END=100) YLAB

*  Read x,y data
      DO NP=1,NPMAX
         READ (1,*,END=100) X(NP),Y(NP)
      END DO
      NP=NPMAX+1

*  Adjust number of points
 100  CONTINUE
      NP=NP-1

*  Plot the graph
      IF (NP.GE.2) THEN
         CALL SNX_AGOP
         CALL SNX_EZRXY(X,Y,NP,XLAB,YLAB,GLAB)
         CALL SGS_CLOSE
      ELSE
         PRINT *,'*** Insufficient Data ***'
      END IF
```

```
      END
```

Notice that most of the program is devoted to reading in the data, and that the plotting itself requires just three calls – to open, plot and close respectively.

The reasons SNX_EZRXY expects you to do the SGS/GKS open and close rather than providing truly one-call plotting by doing them itself are:

- you may not want to use SGS
- you may want to plot within a selected region of the display surface
- you may want to do more graphics I/O later on in the same program.

The trade-off is fractionally less convenience versus greatly increased versatility.

C.2 Input file

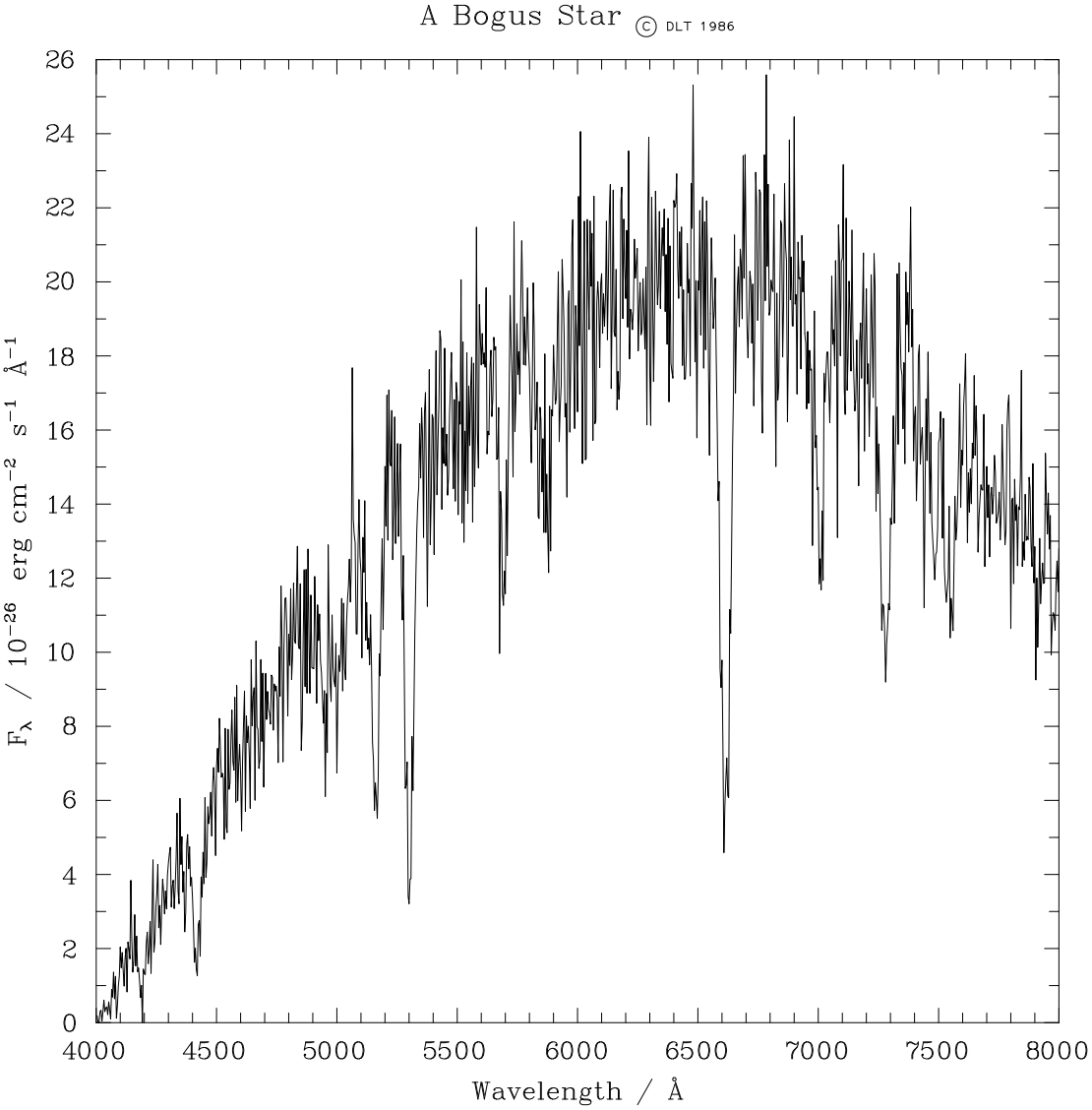
The beginning of the file which was used to produce the graph on the next page is given below. Note the use of PWRITX function codes in the three label records; this assumes the program has been linked with the special version of the AGPWRT routine (see §3 and 7). Deliberately complicated example labels have been contrived; most real applications would require just plain text without PWRITX codes.

```
A Bogus Star ’V:-50IGL’Y’H:-80KRU’C  DLT 1986
Wavelength / ’.A’
F’GLB’K’NRU’ / 10’S’-26’N’ erg cm’S’-2’N’ s’S’-1’N’ ’.AS’-1
   4004.000      0.1888006
   4008.000      0.0000000E+00
   4012.000      0.0000000E+00
   4016.000      0.3046547
   4020.000      0.3328641
   4024.000      3.8437031E-02
   4028.000      0.2980890
             :
             :
             :
             :
```

XYPLOT is a convenient tool for quickly debugging complicated label strings. With a text editor, set up a dummy input file consisting of the label string you want to try out, then two more arbitrary labels and two arbitrary data points. Run the XYPLOT program and inspect the x-axis label; if wrong, edit the data file and try again.

C.3 The plot

XYPLOT produced the following graph when the example input file given in the previous section was used. Further copies can be obtained by running XYPLOT and replying with the name of the example input file to the ‘Filename?’ prompt.


A Bogus Star © DLT 1986
F_λ / 10^{-26} erg cm^{-2} s^{-1} Å^{-1}
0
2
4
6
8
10
12
14
16
18
20
22
24
26
4000
4500
5000
5500
6000
6500
7000
7500
8000
Wavelength / Å

D Demonstration – the SPEED Program

The following pages contain a listing of the Fortran source for a program called SPEED, plus a sample input file. It produces a plot similar to the one from XYPLOT (Appendix C). The SPEED program demonstrates three different compromises between plotting speed and quality of result. The source and a suitable data file (which is the same format as for XYPLOT) are distributed with the SNX software: in the directory NCAR_DIR on VAX/VMS and in the archive `/star/starlink/lib/snx/examples/snx-examples.tar` on UNIX.

On VAX/VMS, this program may be compiled, linked and run using the command sequence

```
$ FORTRAN NCAR_DIR:SPEED
$ LINK SPEED, NCAR_DIR:SNX_LINK/OPT, STAR_LINK/OPT
$ RUN SPEED
```

On UNIX, this program may already be installed in directory `/star/bin/examples/snx`. If it has been deinstalled and removed to save space, you can copy the entire source directory to a scratch dircetory, and with the SYSTEM environment variable set appropriately, build and run it thus:

```
% setenv SYSTEM alpha_OSF/1
% ./mk speed
%./speed
```

The program will prompt “Filename?" (respond NCAR_DIR:SPEED.DAT on VAX/VMS and **/star/starlink/lib/snx/speed.dat** on UNIX) and “Workstation?" (give the SGS workstation name). The prompt:

```
Precision/font?  H=h/w, S=GKS, N=NCAR, E=exit
```

will then appear. The three font options are as follows:

- A response of “N" selects the fancy NCAR fonts and allows AUTOGRAPH to determine the numeric labelling density. The result is of high quality but takes a long time to plot.
- A response of “S" selects the GKS default font. The result is much quicker, and still presentable.
- A response of “H" selects the hardware font and requests a reduced density of numeric labelling. On devices which have hardware character generation (most do), the plot is very rapid but may not be especially attractive. (One common defect is that the y-label may appear reversed, because the device is unable to plot characters on their sides and yet the positions of the individual characters are unchanged.)

D.1 Fortran source

```
      PROGRAM SPEED

*+
*
*  Demonstration of different plotting style/speed
*  tradeoffs in NCAR/SGS/GKS
*
*  P T Wallace   Starlink   10 June 1987
*  P C T Rees    Starlink   19 May 1992
*     Replaced FLUSH calls with PLOTIT calls.
*
*+

      IMPLICIT NONE

      INTEGER NPMAX
      PARAMETER (NPMAX=10000)
      INTEGER N,NP,IPREC,NCOUNT
      REAL RNULL1,TICK
      CHARACTER FNAME*80,GLAB*80,XLAB*80,YLAB*80,K*1
      REAL X(NPMAX),Y(NPMAX)
      LOGICAL PLOTED,NTBAD



*  Open input file
      PRINT *,'Filename?'
      READ (*,'(Q,A)') N,FNAME
      OPEN (UNIT=1,STATUS='OLD',FILE=FNAME(:N),READONLY)

*  Read label text
      READ (1,'(A)',END=9000) GLAB
      READ (1,'(A)',END=9000) XLAB
      READ (1,'(A)',END=9000) YLAB

*  Read x,y data
      DO NP=1,NPMAX
         READ (1,*,END=100) X(NP),Y(NP)
      END DO
      NP = NPMAX+1

*  Adjust number of points
 100  CONTINUE
      NP = NP-1

*  Check enough data
      IF (NP.LT.2) GO TO 9000

*  Prepare to plot the graph
      CALL snx_AGOP
      CALL AGGETF('NULL/1.',RNULL1)
      PLOTED = .FALSE.
```

```
*  Select character precision and font
 200  CONTINUE
      PRINT *,'Precision/font?  H=h/w, S=GKS, N=NCAR, E=exit'
      READ (*,'(A)') K
      NTBAD = .FALSE.
      IF (K.EQ.'H'.OR.K.EQ.'h') THEN

*     Hardware characters - fast but tacky
         CALL AGPWRT(0.0,0.0,' ',0,0,0,-100)
         IPREC = 0
         NCOUNT = 1
         TICK = 1.0

      ELSE IF (K.EQ.'S'.OR.K.EQ.'s') THEN

*     GKS software characters - reasonably fast and attractive
         CALL AGPWRT(0.0,0.0,' ',0,0,0,-100)
         IPREC = 2
         NCOUNT = 6
         TICK = RNULL1

      ELSE IF (K.EQ.'N'.OR.K.EQ.'n') THEN

*     PWRITX characters - fancy but slow
         CALL AGPWRT(0.0,0.0,' ',0,0,0,100)
         IPREC = 2
         NCOUNT = 6
         TICK = RNULL1

      ELSE IF (K.EQ.'E'.OR.K.EQ.'e') THEN

*     Exit requested - wrap up
         CALL sgs_CLOSE
         GO TO 9999

      ELSE

*     Unrecognised command
         NTBAD = .TRUE.
         PRINT *,'?'

      END IF

*  Repeat prompt if problems
      IF (NTBAD) GO TO 200

*  Clear the zone if necessary
      IF (PLOTED) CALL sgs_CLRZ

*  Setup for plotting:
*
*  SGS/GKS text precision
      CALL sgs_SPREC(IPREC)
```

```
*  Density of tick marks and numeric labels
      CALL AGSETI('LEFT/MAJOR/COUNT.',NCOUNT)
      CALL AGSETI('RIGHT/MAJOR/COUNT.',NCOUNT)
      CALL AGSETI('BOTTOM/MAJOR/COUNT.',NCOUNT)
      CALL AGSETI('TOP/MAJOR/COUNT.',NCOUNT)
      CALL AGSETF('LEFT/MINOR/SPACING.',TICK)
      CALL AGSETF('RIGHT/MINOR/SPACING.',TICK)
      CALL AGSETF('BOTTOM/MINOR/SPACING.',TICK)
      CALL AGSETF('TOP/MINOR/SPACING.',TICK)

*  Plot the graph
      CALL snx_EZRXY(X,Y,NP,XLAB,YLAB,GLAB)
      CALL PLOTIT(0,0,2)
      CALL sgs_FLUSH

      PLOTED = .TRUE.

*  Next plot
      GO TO 200

*  Exits
 9000 CONTINUE
      PRINT *,'Insufficient data!'

 9999 CONTINUE

      END
```

D.2 Input file

The beginning of the file which is supplied for use with the SPEED program is given below. It is the same format as for the XYPLOT program, but note that the three label records should not contain PWRITX function codes as these will not work when the “S" or “H" options are used.

```
Simulated Stellar Spectrum
Wavelength
Flux
   5000.000       6.738085
   5004.000       8.849804
   5008.000       9.914771
   5012.000       9.484353
   5016.000       9.981673
   5020.000       11.45505
   5024.000       8.954852
   5028.000       11.32500
             :
             :
             :
             :
```

E Demonstration – the INDEMO Program

The following pages contain a listing of the Fortran source for a program called INDEMO. It produces a plot of a spiral, with a logarithmic y-axis and an x-axis with the scale running right to left. The objective is to demonstrate the SNX_CURS input routine, and to show that data coordinates can be read in and used. Again, the source is distributed with the SNX software: in the directory NCAR_DIR on VAX/VMS and **/star/starlink/lib/snx/examples** on UNIX.

On VAX/VMS, this program may be compiled, linked and run using the command sequence

```
$ FORTRAN NCAR_DIR:INDEMO
$ LINK INDEMO, NCAR_DIR:SNX_LINK/OPT, STAR_LINK/OPT
$ RUN INDEMO
```

On UNIX, this program may already be installed in directory `/star/bin/examples/snx`. If it has been deinstalled and removed to save space, you can copy the entire source directory to a scratch dircetory, and with the SYSTEM environment variable set appropriately, build and run it thus:

```
% setenv SYSTEM alpha_OSF/1
% ./mk indemo
% ./indemo
```

Give the SGS workstation name in response to the “Workstation?" prompt. The graph will be plotted, and the cursor will appear. Adjust the cursor position and press the “hit" key (normally numeric 1 and 2 on the keyboard). A cross will be drawn at the cursor position, and next to it a message giving the data coordinates at that point. The program will exit when a cursor “hit" is made outside the plot bounds, or a zero cursor choice entered.

E.1 Fortran source

```
      PROGRAM INDEMO

*+
*
*     - - - - - - -
*      I N D E M O
*     - - - - - - -
*
*   Demonstrate cursor input following NCAR AUTOGRAPH plot.
*
*   P T Wallace   Starlink   May 1987
*   P C T Rees    Starlink   April 1992
*      Added exit on cursor choice 0.
*
*+

      IMPLICIT NONE
```

```
      INTEGER I,N
      LOGICAL LOOP
      REAL V,T,X(1000),Y(1000),XU,YU,XG,YG,XA(1),YA(1)
      CHARACTER S*13

      REAL snx_AGUGX,snx_AGUGY



*  Fill the data arrays.
      XU = 300.0
      YU = 400.0

      DO I=1,1000
         V = REAL(I-1)
         T = V/50.0
         X(I) = XU + V*COS(T)/2.0
         Y(I) = YU + V*SIN(T)/2.5
      END DO

*  Open NCAR/SGS/GKS.
      CALL snx_AGOP

*  X axis is reversed linear.
      CALL AGSETI('X/ORDER.',1)

*  Y axis is logarithmic.
      CALL AGSETI('Y/LOG.',1)

*  Plot the graph.
      CALL snx_EZRXY(X,Y,1000,'Spiral','x','y')

*  Loop - mark positions on plot.
      LOOP = .TRUE.

      DO WHILE (LOOP)

*     Input a cursor position.
         CALL snx_CURS(XU,YU,N)

*     Check cursor choice and finish if zero.
         IF (N.LE.0) THEN

*        Finish.
            LOOP=.FALSE.
         ELSE

*        Transform to grid coordinates.
            XG = snx_AGUGX(XU)
            YG = snx_AGUGY(YU)

*        Inside or outside grid window?
            IF (XG.LE.1.0.AND.XG.GE.0.0.AND.
```

```
     :          YG.LE.1.0.AND.YG.GE.0.0) THEN

*           Inside - mark the selected position.
               WRITE (S,'(''('',F5.1,'','',F5.1,'')'')') XU,YU
               XA(1) = XU
               YA(1) = YU
               CALL POINTS(XA,YA,1,-2,0)
               CALL snx_WRTST(XG+0.02,YG,S,0.02,0,-2)
            ELSE

*           Outside - finished.
               LOOP = .FALSE.
            END IF
         END IF

*  Next position
      END DO

*  Wrap up
      CALL sgs_CLOSE

      END
```

F PWRITX

The special version of the AGPWRT routine described earlier uses the PWRITX routine to plot the required character string. A feature of this routine is that function codes can be embedded in the character string to select different fonts, draw subscripts, *etc.* These function codes are the ones you will need to use if you wish to have Greek letters, *etc.* in your labels. The AGPWRITX documentation is given below, and is followed by a plot showing all the available characters (the Roman characters being those in the “complex" font).

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

This routine is a substitute for the NCAR routine of the same name, but allowing the PWRITX character drawing routine to be used instead of the usual PWRIT, giving access to special fonts, *etc.*

The character string accepted by the PWRITX routine includes control codes to switch font, *etc.* This implementation of AGPWRT accepts and passes PWRITX codes and also inserts the appropriate codes where lowercase and certain other characters are supplied. Some special characters not directly available from PWRITX are produced by means of control sequences selected through additional codes which are peculiar to this implementation of AGPWRT.

Other special values of the ICEN parameter are used to select plotting via either the PWRITX routine or the PWRIT routine. This feature allows an application to plot using GKS character drawing facilities on some occasions while using PWRITX on other occasions. The PWRITX option gives high quality characters from a rich set, but slowly; the GKS option, on the other

hand, provides access to other GKS fonts and allows the use of lower text precision on graphics devices where this will reduce the plotting time.

The NCAR utilities use values of the justification argument ICEN of ± 1 where left or right justification is required, and in these cases the string is plotted artificially monospaced to avoid irregular axis labelling, subject to the restriction that the string must not contain PWRITX function codes (though lowercase and common punctuation symbols are acceptable). To enable left and right justified strings to be proportionally spaced and to contain PWRITX function codes, this implementation of AGPWRT also supports the non-NCAR values ICEN $= \pm 2$. There is no provision for centred monospaced strings.

Given:

XPOS,YPOS	r,r	string position in SPPS user coordinates
CHRS,NCHS	c,i	string and length, inclusive of codes, *etc.*
ISIZ	i	character size
IORI	i	orientation
ICEN	i	justification, or PWRIT/PWRITX selection

Where the special values of ICEN are used to select plotting via either PWRIT or PWRITX, the other arguments are ignored.

Plotter units:

> Displacements and character sizes are specified in “plotter units". (N.B. Do not confuse plotter units with device coordinates. Plotter units are not related to true device resolution.) Normally the longest dimension of the display surface is 1024 plotter units. Should the resolution limitation that this implies be unacceptable, it can be changed by means of the SPPS routine SETI.

PWRITX fonts:

> Twelve fonts are provided, each containing 47 symbols and indexed by the standard Fortran characters (`A-Z 0-9 +-*/()$=,.` and space). A given font is specified by three letters which, in broad terms, select which combination of size (Principal, Indexical, Cartographic), alphabet (Roman, Greek), and case (upper, lower) is required.

ISIZ argument:

> ISIZ specifies an overall magnification factor for all the characters drawn.
>
> Each of the three size classifications in the set of twelve fonts provided (Principal, Indexical and Cartographic) has a nominal character width (in spite of proportional spacing), height and total height. The space character is exactly one nominal width wide, and a carriage control code causes a downwards increment equal to the nominal total height, which thus includes white space.

The nominal width (W), height (H), and total height (T) for the Principal (P), Indexical (I), and Cartographic (K) fonts are as follows (plotter units):

	W	H	T
P	16	21	32
I	12	13	20
K	8	9	14

ISIZ values of 3 or less select standard magnifications of the above sizes. Values of 4 or more are proportional to the resulting magnification. The overall magnification for different ISIZ values is as follows:

ISIZ	mag
< 1	8/21
1	12/21
2	18/21
3	24/21
> 3	ISIZ/21

Thus, for ISIZ>3, Principal characters will be drawn nominally ISIZ plotter units high, and ISIZ can simply be thought of as the nominal character height in plotter units.

IORI argument:

IORI is the string orientation in degrees anticlockwise from the normal left-to-right.

ICEN argument (normal use to specify positioning):

ICEN	Meaning
–2	(XPOS,YPOS) is the centre of the left edge of the first character. The string is proportionally spaced.
–1	(XPOS,YPOS) is the centre of the left edge of the first character. The string is monospaced. Lowercase and punctuation characters are permitted, but not function codes.
0	(XPOS,YPOS) is the centre of the entire string. The string is proportionally spaced.
+1	(XPOS,YPOS) is the centre of the right edge of the last character. The string is monospaced. Lowercase and punctuation characters are permitted, but not function codes.
+2	(XPOS,YPOS) is the centre of the right edge of the last character. The string is proportionally spaced.

ICEN argument (special use to select string drawing routine):

ICEN	Meaning
–100	Selects PWRIT routine, which gives access to GKS fonts, precision, em etc.
+100	Selects PWRITX routine, which gives access to special fonts and other features.

Notes:

- For ICEN = $\pm$100 the other arguments are ignored.
- Initially, PWRITX is selected.
- This inelegant use of the ICEN argument is a consequence of having to work within the standard AGPWRT call, which is made directly by the AUTOGRAPH utilities.

Characters available without using function codes:

- All uppercase and lowercase Roman characters, and most common punctuation symbols are available by including them literally in the string CHRS.
- Two consecutive apostrophes causes a single apostrophe to be drawn – for example "`Murphy''s Law`".
  Both of these are features of this implementation of AGPWRT and are not available when directly calling PWRITX.

Function codes:

Function codes are sequences of characters, enclosed within apostrophes, which may (except when ICEN= ± 1) be included in the character string CHRS to change font, case, *etc.* within the plotted string. No punctuation is needed between functions except for a comma between adjacent numbers; however, commas may be used between functions to improve readability. The following are the only legal function codes. Any other characters in a function string will be ignored except that an error report will be issued and, if more than 10 errors occur within a string, control will be returned to the main program without further plotting. At the start of the string, size, type, and case are Principal, Roman, and Upper.

FONT ALPHABET

R Roman characters, *etc.*

G Greek characters, *etc.*

FONT SIZE (see table, above, under ISIZ)

P Principal fonts

I Indexical fonts

K Cartographic fonts

FONT CASE

U or Un Upper case. If U is followed by a number n (not separated by a comma) then n characters will be drawn in uppercase and subsequent characters will be in lowercase. The U1 option is thus particularly useful for capitalizing sentences.

L or Ln Lower case. If L is followed by a number n, then n characters will be drawn in lower case and subsequent characters will be in upper case.

SUBSCRIPTS AND SUPERSCRIPTS

S or Sn Superscript level.

B or Bn Subscript level.

N or Nn Normal level.

When switching from some “base" character size to super- or subscripting, the character size will change depending on the base character size. Principal base characters will be subscripted or superscripted with indexical characters, with a 10 plotter unit shift (scaled in accordance with ISIZ) up or down. Indexical and cartographic base characters will be sub- or superscripted with cartographic characters with a 7 plotter unit shift.

If multiple S or B functions are used, the original base character is forgotten, and the base is simply the one preceding the latest S or B.

The case of the indexing characters will generally be the same as that of the base character unless otherwise specified. There is one exception: a lower case indexical base will be super- or subscripted with upper case cartographic, as the lowercase cartographic font is composed of special characters rather than letters and numbers.

If S, B or N is followed by a number n, then n characters will be drawn as specified above, after which character size, case and position will be reset to that of the base characters. If n is negative, its absolute value will be used instead; n cannot be zero. Do not overlap level definitions given for a specified number of characters.

The N option returns character, case and size to that of the base but maintains the current character position.

```
For example:  'U1'T'S1'EST    will be drawn   e
                                             Tst

        and   'U1'T'S'E'N'ST  will be drawn   e
                                             T st
```

DIRECT CONTROL OVER POSITION

Hn, HnQ Increment horizontally in the frame. Hn will shift the present X position by n plotter units. HnQ will shift the present X position by n nominal character widths. Positive n shifts to the right, and negative to the left. If n is omitted, a value of 1 is assumed.

Vn, VnQ Increment vertically in the frame. Vn will shift the present Y position by n plotter units. VnQ will shift the present Y position by n nominal character total heights. Positive n shifts upwards, and negative downwards. If n is omitted, a value of 1 is assumed.

H:n Increment along the direction of the string by n percent of the nominal character height for Principal characters. If n is positive the shift is rightwards, and if negative the shift is leftwards. If n is absent there is no shift.

V:n Increment at right angles to the direction of the string by n percent of the nominal character height for Principal characters. If n is positive the shift is upwards, and if negative the shift is downwards. If n is absent there is no shift.

(Note: these two are nonstandard features peculiar to this implementation of AGPWRT and not available when PWRITX is called directly.)

X or Xn & Y or Yn Set X or Y. If the X or Y appears without a number n, the function will do nothing. Otherwise, the character coordinate in the X or Y direction will be set to the plotter coordinate n, so that the next character drawn will have this position in X or Y, subsequent characters will be drawn from this position.

(Note: within PWRITX, interactions with the proportional font and justification logic make these functions hard to use, and they are not recommended.)

C Carriage return: a carriage return and line feed will be done before the next character is plotted. N.B. The justification applies to the final line.

DIRECTION

D or Dn — Write down, rather than across the frame. If D appears without an n or if n=0, all characters will be written down until an 'A' function is encountered. If D is followed by a number n, n characters will be written down and subsequent characters will be written across the frame. If n is negative, the absolute value of n is used instead.

A — Write across: escape from the D option.

SPECIAL CHARACTERS

.A Angstrom unit

For example, to draw a string meaning "three Angstrom units per millimetre" we might use:

```
CALL AGPWRT( XPOS, YPOS, '3''.A''/mm', ...
```

DIRECT CHARACTER ACCESS

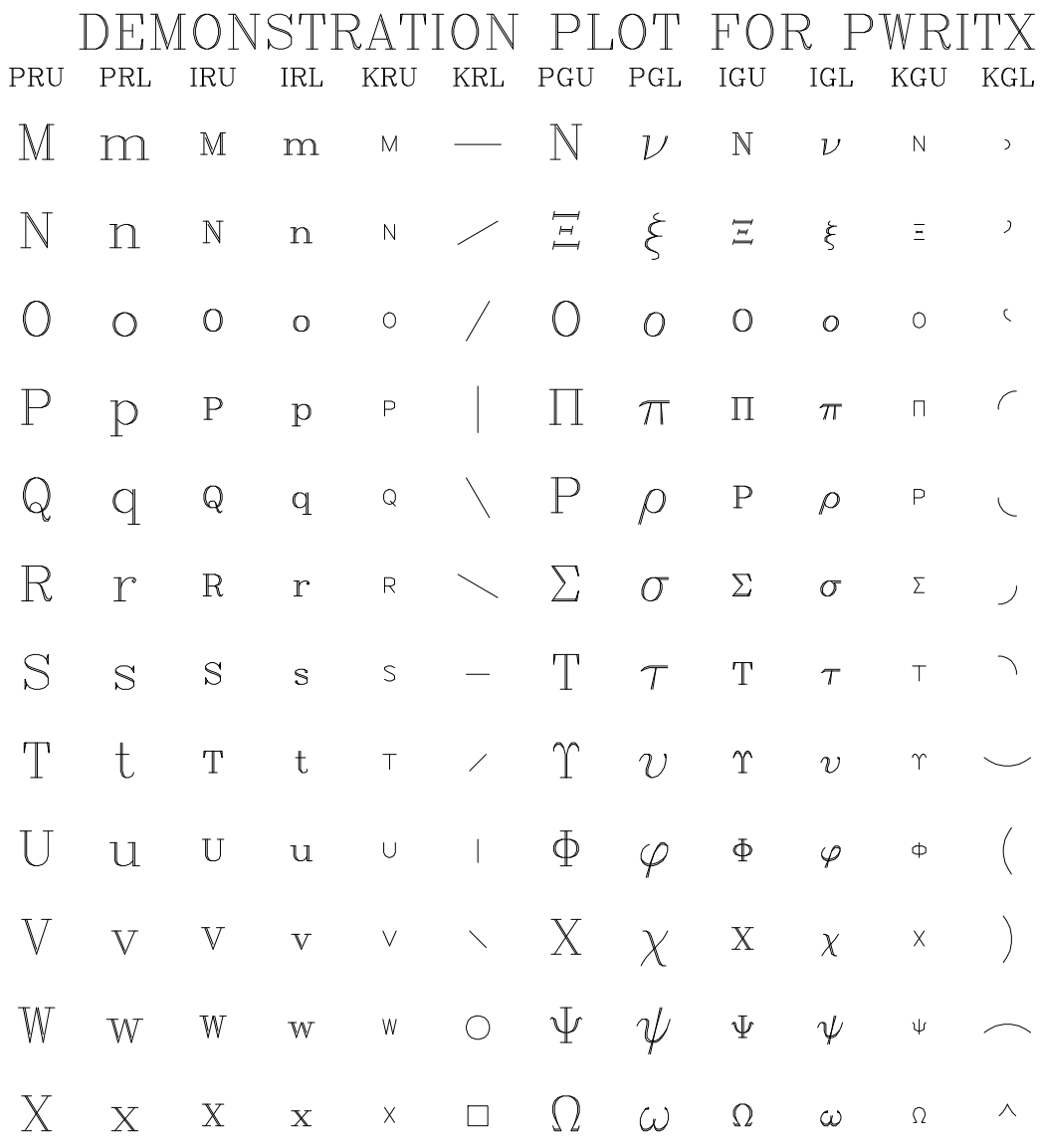
nnn Numeric character: character number nnn (octal) will be drawn.

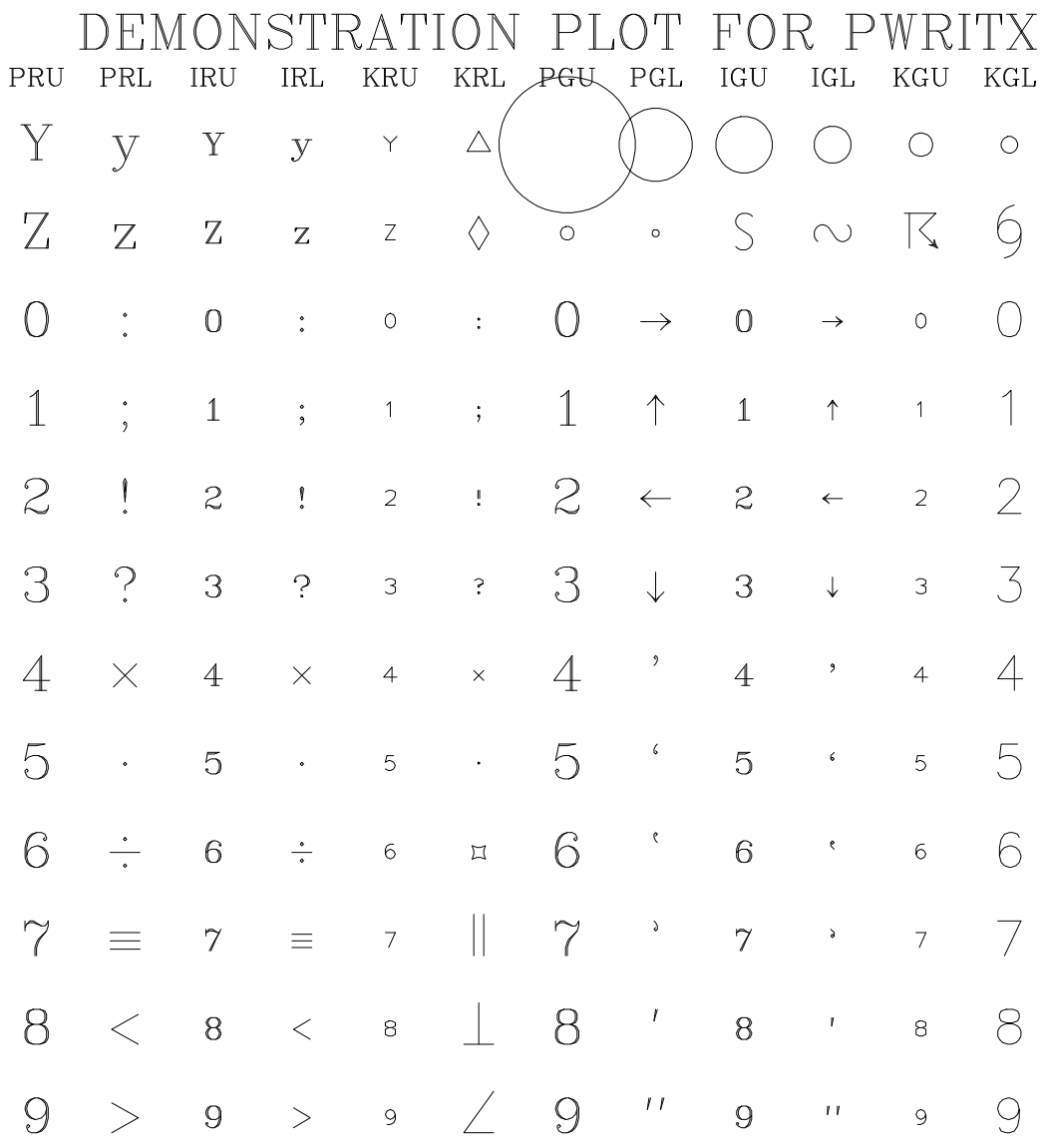
Called: AGGETI, AGSETI, SETER, PWRITX, PWRIT, GTNUM, KUPX, KUPY, CPUX, CPUY

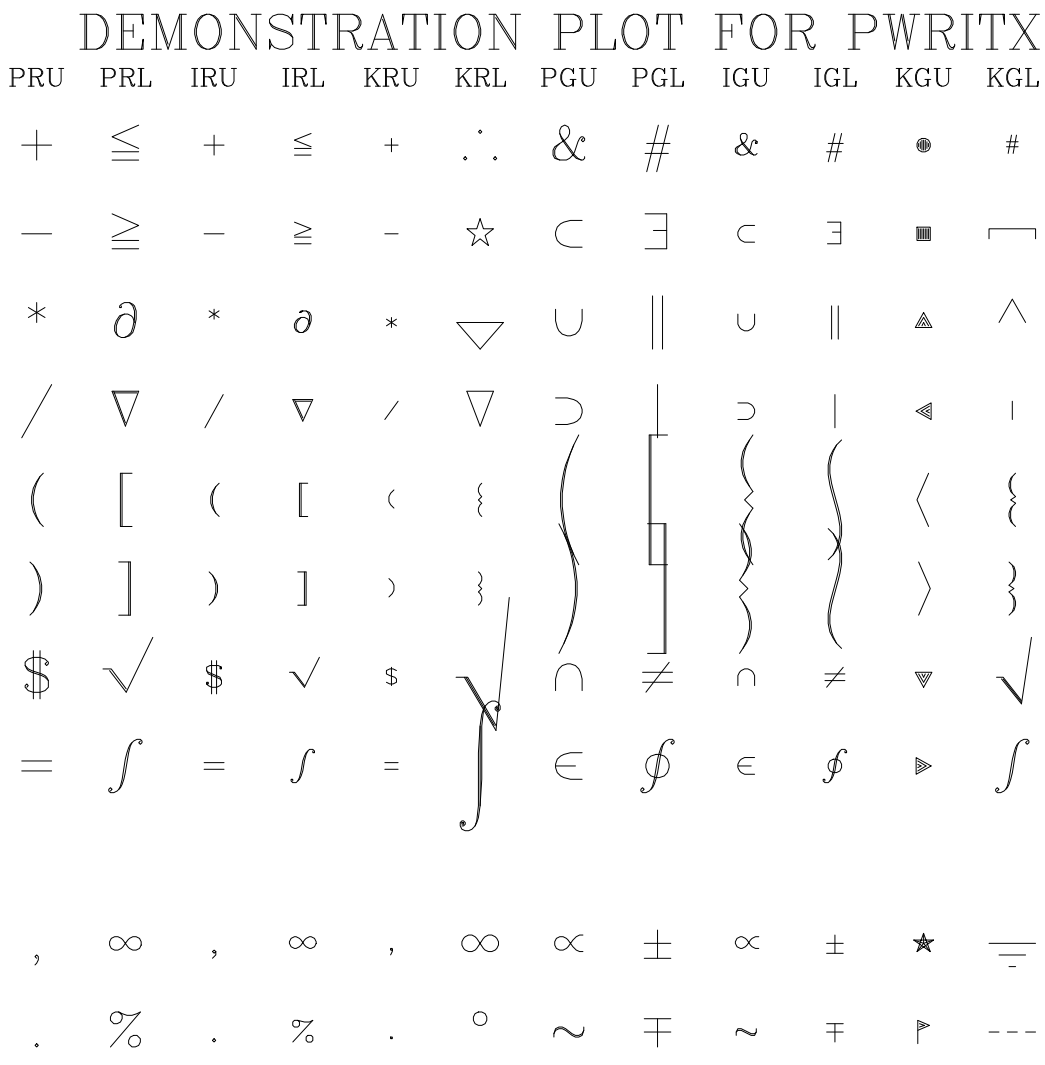
This routine uses characters outside the ANSI Fortran 77 character set.

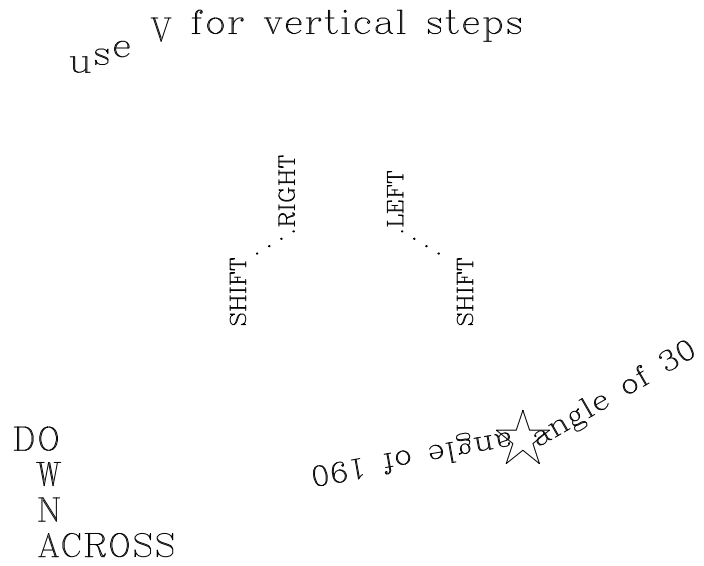
DEMONSTRATION PLOT FOR PWRITX

PRU	PRL	IRU	IRL	KRU	KRL	PGU	PGL	IGU	IGL	KGU	KGL
A	a	A	a	A	@	A	α	A	α	A	@
B	b	B	b	B	§	B	β	B	β	B	§
C	c	C	c	C	†	Γ	γ	Γ	γ	Γ	†
D	d	D	d	D	‡	Δ	δ	Δ	δ	Δ	‡
E	e	E	e	E	⊙	E	ε	E	ε	E	⊙
F	f	F	f	F	♀	Z	ζ	Z	ζ	Z	♀
G	g	G	g	G	⊕	H	η	H	η	H	⊕
H	h	H	h	H	♂	Θ	ϑ	Θ	ϑ	Θ	♂
I	i	I	i	I	—	I	ι	I	ι	I	◠
J	j	J	j	J	/	K	κ	K	κ	K	◠
K	k	K	k	K	|	Λ	λ	Λ	λ	Λ	◡
L	l	L	l	L	\	M	μ	M	μ	M	◡


DEMONSTRATION PLOT FOR PWRITX
PRU PRL IRU IRL KRU KRL PGU PGL IGU IGL KGU KGL


DEMONSTRATION PLOT FOR PWRITX
PRU PRL IRU IRL KRU KRL PGU PGL IGU IGL KGU KGL


DEMONSTRATION PLOT FOR PWRITX
PRU PRL IRU IRL KRU KRL PGU PGL IGU IGL KGU KGL



2 loWER, 3 UPPer

this is S^uperscripting
THIS IS S_UBSCRIPTING

SHOW U^se of NORMAL

(X99, Y99)

(X50, Y50)