Simpleplot 2-15

SIMPLEPLOT
ViSualization

BUSS Ltd.

SIMPLEPLOT ViSualization

Copyright © 2000 by

BUSS Ltd.
Business and Innovation Centre
Angel Way
Bradford
West Yorkshire
BD7 1BX
United Kingdom

Telephone	01274 841396
	+44 1274 841396
Fax	01274 841388
	+44 1274 841388

Web Site	http://www.buss.co.uk/buss
Email	info@buss.co.uk

All rights reserved.

Seventh Edition, October 2000

Contents

List of Figures

Preface

SIMPLEPLOT is a library of subroutines for plotting graphs. A wide variety of graphs can be drawn as well as more general pictures and diagrams. Facilities are biased towards the graphical representation of data; in particular, scientific data.

SIMPLEPLOT was originally designed for programmers who wanted to draw pictures of their data with minimum programming effort. Although it still achieves this goal, SIMPLEPLOT has developed into a much more powerful tool which can also be used for professional software applications.

Six separate sections constitute the complete SIMPLEPLOT Mark 2:

1. The basic package for conventional graph plotting – *x*-*y* plots and polar plots.

2. Additional subroutines for 3-dimensional plotting – contour maps and surface pictures of 3-D data

4. Additional subroutines for presentation graphics – bar charts, histograms and pie charts.

5. Additional subroutines for plotting functions of three variables – perspective pictures of 4-dimensional data.

6. The SIMPLEPLOT mapping module – for representing data based on geographical coordinate systems.

7. The SIMPLEPLOT ViSualization module – for full colour modelling of functions of 2, 3 and 4 variables.

SIMPLEPLOT-PLUS refers to a SIMPLEPLOT library which is made up from Sections *1*, *2*, and *4* and contains many additional facilities.

This manual refers to subroutines available from Section *7*, for full colour surface plotting. The *SIMPLEPLOT Reference manual* (8^{th} edition) contains full specifications of other general purpose routines.

Graphics device interface

The SIMPLEPLOT library is independent of any single graphics system and is device sensitive with a device independent interface for the user. This means that the user is protected from having to know about the features of the target output device, but SIMPLEPLOT makes as much use of underlying features as possible.

SIMPLEPLOT has already been interfaced to a large number of graphics devices, and the range of validated device drivers is continually being extended. It can address graphics devices directly, or through separate low-level graphics systems (*eg.* GKS, X Window System) or graphics languages (*eg.* PostScript, CGM).

In conjunction with the Motif device driver, an interactive Motif library, SVT can be combined with *SIMPLEPLOT ViSualization* to facilitate the development of interactive Motif applications. SVT provides an interactive Motif environment into which *SIMPLEPLOT ViSualization* software may be integrated to display data. For further information, consult *SVT: SIMPLEPLOT ViSualization Tool.*

Types of pictures

This manual describes SIMPLEPLOT's advanced graphics facilities:

- ViSualizing functions of 2, 3 and 4 variables
- Surface representation of data
- Comparing two dependent functions
- Stacked contour plots
- Drawing objects
- Applying 3-dimensional transformations

The full SIMPLEPLOT library provides a much wider range of facilities but only a subset is described in this manual.

1. Introduction

1.1 Overview

Overview]

The *SIMPLEPLOT ViSualization* manual is a tutorial manual for the SIMPLEPLOT library of graph drawing subroutines. It can be used as a manual in its own right, but for detailed explanations of other SIMPLEPLOT subroutines please refer to the *SIMPLEPLOT Reference manual* (8th edition).

1.2 Software version

The software described in this manual is based on SIMPLEPLOT Mark 2, version 2-15.

1.3 Target audience

Target audience]

The *SIMPLEPLOT ViSualization* manual has been written with the following readers in mind:

- New users of SIMPLEPLOT should be able to use this manual to produce high–quality pictures of data even without previous experience of using the SIMPLEPLOT graphing library. The *SIMPLE-PLOT Primer* may also be referenced to provide a fuller explanation of basic concepts.
- Experienced users probably do not need to read it all, but may choose to skip to the examples and formal specifications.

1.4 Related documents

Related documents]

Related documents include:

- The *SIMPLEPLOT Reference manual* (8th edition) in which *Host Specific Information* is available.
- The *SIMPLEPLOT Primer* which provides an introduction to SIMPLEPLOT, especially those facilities available for plotting 2-D data.
- *SVT: SIMPLEPLOT ViSualization Tool*, which describes the Motif application for the interactive display of 3-D data.

1.5 How to report problems

How to report problems]

Any problems with SIMPLEPLOT software or its associated products and services should be reported to BUSS Ltd on a Software Performance Report (SPR) form. One of these is sent out with every software kit – please photocopy it or contact BUSS Ltd if you would like extra copies.

1.6 How to use this manual

How to use this manual]

If you are a newcomer to SIMPLEPLOT you may find it easier to get started if you use this manual in the following way:

1. Look through the chapter introductions.

   Then either:

   - Convert an existing program into a plotting program by adding calls of subroutines from one of these chapters alone, or
   - Find an example program which is the closest to what you want to produce and adapt it for your data.

2. Execute your program according to your host computer's requirements for a SIMPLEPLOT program.
3. When the program works and produces basic graphs, it can be enhanced by using the comprehensive general-purpose layout and annotation routines as specified in the *SIMPLEPLOT Reference manual* and *SIMPLEPLOT Primer*.

Having gained confidence in using the subroutines, a new user should be able to go on and use any combination of subroutines for different applications.

1.7 Conventions

Conventions]

The following conventions are used in this manual for example programs:

- The example programs have been designed to be as brief as possible.
- In example programs in which data sets are read from files, these are included as part of the software distribution kit.
- Each example program is accompanied by an explanation of those subroutines which have not occurred in any previous example or which are being used in a new context.

1.8 Illustrations

Figures and output from example programs are produced using SIMPLEPLOT version 2-15 with the Colour PostScript device driver.

2. Getting Started

This chapter covers the following topics:

2.1 The raster image

2.2 *SIMPLEPLOT ViSualization* programming

- Before `VSNEW`
- Auxiliary subroutines
- Drawing attributes
- Palette setting subroutines
- Simple drawing
- Surface drawing
- IsoSurface drawing
- Transformation subroutines
- 3-D barcharts
- Annotations

2.3 Data

- Gridded z data
- Gridded x-y-z data
- Data on a 3-D grid
- Data in equal-node elements
- Data in non-equal-node elements

2.4 Missing data

2.1 The raster image

SIMPLEPLOT ViSualization plots representations of 3-dimensional data with hidden line and surface removal on any type of graphics device. The hidden line and surface removal is achieved using Z buffer algorithms to draw into an internal buffer which holds a representation of a raster image. As the image is built up in the buffer, parts of the image can be overdrawn repeatedly; only the final drawing is output to the device when the user triggers it. The use of this method has several consequences:

- there is nothing to see until the buffer is purged.
- it is possible that something drawn may appear to be absent because it has been overdrawn
- the amount of device plotting per image is not dependent on the complexity of the image
- output on pen plotters is possible, but not ideal

2.2 *SIMPLEPLOT ViSualization* programming

Every *SIMPLEPLOT ViSualization* picture is started by `VSNEW` which sets up a raster buffer, and is finished by `VSOUT` which transfers the picture from the raster buffer to the graphics device.

2.2.1 Before `VSNEW`

Some subroutines may be called before `VSNEW` to specify requirements which are brought together and used by `VSNEW`:

Name	*Action*	*Default*
`VS3DLM`	x-y-z scales	-1 to 1 for x, y and z
`VSFITP`	scale length proportions	proportional to data
`VSFULL`	scaling mode	Maximize image
`VSINIT(2)`	restore default scales	
`VSLBOX`	draw Limiting Box	Do not draw
`VSVRTP`	(R, θ, ϕ) viewing position	`VSVRTP(0.0,30.0,15.0)`
`VSVXYZ`	(x, y, z) viewing position	`VSVRTP(0.0,30.0,15.0)`

2.2.2 Auxiliary subroutines

Some subroutines do not contribute directly to drawing, but provide useful auxiliary facilities:

Name	*Action*
`KVXYD`	convert 3-D (x, y, z) to 2-D (x, y) and depth
`QV3DLM`	inquire x-y-z scales
`QVBRFL`	inquire z scale for 3-D barcharts
`QVDPLM`	inquire depth range
`QVSCAL`	inquire scale
`VSINIT`	initialize

2.2.3 Drawing attributes

Drawing attributes can be specified by calling appropriate subroutines before performing the drawing; when no values have been specified for attributes, defaults are used.

Name	*Action*	*Default*
`VSEDGC`	specify edge colour index	1
`VSEDGI`	specify offset and increment of drawn edges	0, 0, 1
`VSEDGV`	specify edge visibility	0
`VSFILC`	specify fill colour index	1
`VSLINC`	specify line colour index	1
`VSPMMG`	specify marker magnification	1.0
`VSPMC`	specify marker colour index	1

2.2.4 Palette setting subroutines

Colour attributes are specified by colour indices; the colours assigned to particular indices can be the default colours, or can be changed using the following:

Name	*Action*
`VSCDEF`	specify colour value from list of defaults
`VSCGS`	specify a grey scale palette for a range of colour indices
`VSCHLS`	specify colour value by HLS
`VSCIM`	specify method of interpolating colour sequences
`VSCRGB`	specify colour value by RGB
`VSCRNG`	specify graded range of colour values

2.2.5 Simple Drawing

A few subroutines are offered for performing simple plotting tasks from (x, y, z) coordinates:

Name	Action
`VSBLFL`	draw a single 3-D block
`VSBLTN`	tint a single 3-D block
`VSPGFE`	fill an edge-flagged polygon
`VSPGFL`	fill polygon
`VSPGTE`	tint an edge-flagged polygon
`VSPGTN`	tint polygon
`VSPLDR`	draw polyline
`VSPLTN`	tint polyline
`VSPMDR`	draw polymarker

2.2.6 Surface plotting

Complete surfaces with coloured contours can be drawn from several data configurations:

Name	*Action*
`VSEQX`	specify x grid values
`VSEQY`	specify y grid values
`VSRG`	draw surface from 2-D z array with regular x and y
`VSRGU`	tint surface from 2-D z and u arrays with regular x and y
`VSUTOC`	specify how the data range maps to colour indices
`VSXYZ`	draw surface from 2-D x, y and z arrays
`VSXYZU`	tint surface from 2-D x, y z and u arrays
`VSZ`	draw surface from x, y and z arrays with element structure
`VSZU`	tint surface from x, y, z and u arrays with element structure

2.2.7 IsoSurface plotting

Complete IsoSurfaces with or without contours of a second variable can be drawn:

Name	*Action*
`VSEQZ`	specify z grid values
`VSIS`	draw IsoSurface from 3-D v data array with regular x, y, and z
`VSISND`	specify whether IsoSurfaces should exclude the six end planes of the data
`VSISU`	draw IsoSurface from 3-D v data array and overlay contours from u data array with regular x, y, and z
`VSLSLD`	specify the direction of a light source
`VSLSSM`	specify the shading method for light-sourcing

2.2.8 Transformation subroutines

The (x, y, z) coordinates supplied as data can be plotted on the x-y-z scales of the picture; it is also possible to specify transformations to be performed on the coordinates before plotting:

Name	*Action*
`VSMAG`	magnify
`VSROT`	rotate
`VSTRAN`	translate

2.2.9 3-D barcharts

3-D barcharts can be drawn from arrays of data:

Name	*Action*
`QVBRFL`	inquire z scale for `VSBRFL`
`VSBRBZ`	specify base level for 3-D barcharts
`VSBRFC`	specify colour mode for `VSBRFL`
`VSBRFL`	fill 3-D barchart
`VSBRSZ`	specify bar widths for 3-D barcharts
`VSBRTN`	tint 3-D barchart
`VSBRUP`	specify stacking direction for stacked 3-D barcharts

2.2.10 Annotations

Ordinary SIMPLEPLOT subroutines such as `TITLE7` may be used to annotate *SIMPLEPLOT ViSualization* pictures; some additional *SIMPLEPLOT ViSualization* specific annotations are also available:

Name	*Action*
`VSAXDR`	draw 3-D axes
`VSAXFC`	specify axis plane fill colour indices
`VSAXGC`	specify axis plane grid colour indices
`VSK7H`	draw horizontal key
`VSK7V`	draw vertical key

`VSAXDR` draws axes annotating the 3-D scales of the current picture. It must be called after `VSNEW` in order to pick up the scaling details, and is best called after `VSOUT`. `VSAXDR` does not draw into the raster image, and the axes may be overdrawn by the image if it is called before `VSOUT`.

The axes are drawn by the same processes which draw 2-D axes from SIMPLEPLOT, and can be modified by the same `AX*` subroutines, specifying axis types as `'X3'`, `'Y3'` and `'Z3'`.

The user data scale drawn with a key is drawn as a `'U3'` axis, and may be modified by appropriate `AX*` subroutines.

This table shows the effect of axis subroutines on axis types `'X3'`, `'Y3'`, `'Z3'` and `'U3'`. A fuller version of this table can be found in Appendix T of the *SIMPLEPLOT Supplement*.

Subroutine	X3	Y3	Z3	U3
AXCLR	√	√	√	√
AXCRSS	×	×	×	×
AXGRID	×	×	×	×
AXIS7	√	√	√	×
AXLAB7	√	√	√	×
AXLBAN	$\surd^U$	$\surd^U$	$\surd^U$	×
AXLBGP	√	√	√	√
AXLBJS	×	×	×	×
AXLBLV	×	×	×	×
AXLBSL	×	×	×	×
AXLBSP	√	√	√	×
AXLBTM	√	√	√	×
AXLBTP	√	√	√	×
AXLOCN	√	√	√	√
AXMAJ	√	√	√	×
AXMIN	√	√	√	×
AXRNGE	√	√	√	√
AXSBDV	√	√	√	√
AXSBMN	√	√	√	√
AXSBTK	$\surd^U$	$\surd^U$	$\surd^U$	$\surd^U$
AXSUBS	√	√	√	√
AXTXT7	√	√	√	×

U: Preceding, Following and Inside unavailable

`VSAXFC` specifies colour indices for subsequent calls of `VSNEW` to fill the planes at the rear of the picture, where −1 (default) means leave the planes unfilled. The first index is for the rear plane(s) with x constant, the second index is for rear plane(s) with y constant, and the third index is for rear plane(s) with z constant.

`VSAXGC` specifies colour indices for subsequent calls of `VSNEW` to draw axis grids on the planes at the rear of the picture, where −1 (default) means no grid. The first index is for the rear plane(s) with x constant, the second index is for rear plane(s) with y constant, and the third index is for rear plane(s) with z constant.

Any `AX*` subroutines which affect the position of grid lines must be called before `VSNEW` to ensure that the grids on the back planes mask the tick marks on axes drawn by `VSAXDR` or `AXIS7`.

`VSK7H` and `VSK7V` draw keys to the current mapping between user data values and colours, which can be controlled by `VSUTOC`. In the present release, the key is drawn using the same raster buffer as used for pictures, therefore these subroutines must not be called between `VSNEW` and `VSOUT`.

The following subroutines must be called before `VSNEW`:

Name	*Action*
AXCLR	specify level of axis annotation
AXRNGE	specify the subrange over which an axis is to be drawn
AXSBDV	specify the interval of axis subdivisions and annotation
AXSBMN	specify minor axis subdivisions independently
AXSUBS	specify the number of major and minor axis subdivisions
VSAXFC	specify axis plane fill colour indices
VSAXGC	specify axis plane grid colour indices

The following subroutines must be called after `VSNEW`, and are best called after `VSOUT`:

Table 2.1 Relative positions of titles, keys and captions

	`'W'`est	`'P'`receding `'L'`eft	`'C'`entre	`'F'`ollowing `'R'`ight	`'E'`ast
`'N'`orth	*page*	*group*	*page*	*group*	*page*
`'H'`igher	*page*	*group*	*group*	*group*	*page*
`'O'`ver	*page*	*picture*	*picture*	*picture*	*page*
`'T'`op	*page*	*picture*	*picture*	*picture*	*page*
`'C'`entre	*page*	*picture*	*picture*	*picture*	*page*
`'B'`ottom	*page*	*picture*	*picture*	*picture*	*page*
`'U'`nder	*page*	*picture*	*picture*	*picture*	*page*
`'L'`ower	*page*	*group*	*group*	*group*	*page*
`'S'`outh	*page*	*group*	*page*	*group*	*page*

Name	*Action*
`AXIS7`	draw an axis
`AXLAB7`	add an axis label and tick mark
`AXMAJ`	draw a single major axis subdivision
`AXMIN`	draw a single minor axis subdivision
`AXTXT7`	draw a set of axis labels and tick marks
`VSAXDR`	draw 3-D axes

The following subroutines do not affect the axis grids, and may therefore be called any time before `VSAXDR`/`AXIS7`:

Name	*Action*
`AXLBAN`	specify the style of axis annotation labels
`AXLBGP`	specify the level of axis annotation near an intersection
`AXLBSP`	specify the separators between *time-date* based axis labels
`AXLBTM`	specify the components of *time-date* based axis labels
`AXLBTP`	specify numeric or *time-date* annotation
`AXLOCN`	specify location of an axis
`AXSBTK`	specify the style of axis tick marks

The following subroutines must be called after `VSOUT`

Name	*Action*
`VSK7H`	draw horizontal key
`VSK7V`	draw vertical key

Keys are positioned relative to a page, picture or group, according to the values of `VCHAR` and `HCHAR`, the first two arguments. Table 2.1 shows the effect of `VCHAR` and `HCHAR` on positions.

Keys which are positioned relative to a picture must be drawn after `VSOUT` has been called. Those positioned relative to a page or group of pictures may also be drawn before `CALL VSNEW`.

2.3 Data

Data]

SIMPLEPLOT ViSualization can directly interpret five forms of data.

2.3.1 Gridded z data

Gridded z data]

`REAL` array `Z2ARR(NX,NY)` contains a regular grid of function values (*eg.* surface heights) at every combination of `NX` equally spaced x values and `NY` equally spaced y values. `Z2ARR` is a single valued function of x and y – for every x-y pair there is only a single corresponding `Z2ARR` value (see Figure 2.1). Corresponding data structures are covered by the `RG*` subroutines in the main SIMPLEPLOT library. This data configuration can be plotted using `VSRG` and `VSRGU`.

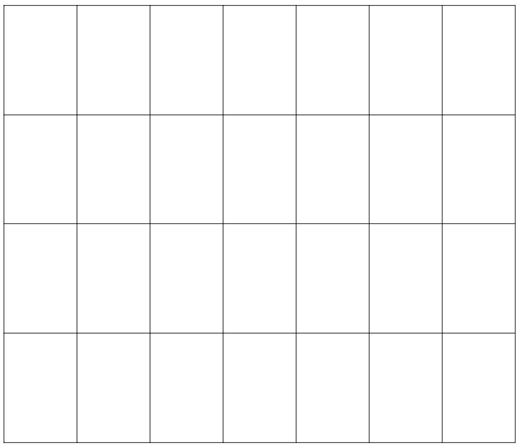


Figure 2.1 Gridded z data

Most examples in chapter 9 draw the surface corresponding to a 2-dimensional array z. By default the data range is mapped onto colour indices 1 to 15.

2.3.2 Gridded x-y-z data

Gridded x-y-z data]

`REAL` arrays `X2ARR(NX,NY)`, `Y2ARR(NX,NY)` and `Z2ARR(NX,NY)` contain the (x, y, z) coordinates of a grid of points. The grid is constructed by joining adjacent points in the arrays, to define a set of planar quadrilateral surfaces.

The coordinates

```
(X2ARR(I,J), Y2ARR(I,J), Z2ARR(I,J))
(X2ARR(I+1,J), Y2ARR(I+1,J), Z2ARR(I+1,J))
(X2ARR(I+1,J+1), Y2ARR(I+1,J+1), Z2ARR(I+1,J+1))
(X2ARR(I,J+1), Y2ARR(I,J+1), Z2ARR(I,J+1))
```

define the four corners of a planar surface (see Figure 2.2).

A simple example of this sort of data is (x, y, z) coordinates of points on latitudinal and longitudinal grids on the surface of the earth. This data configuration can be plotted using `VSXYZ` and `VSXYZU`.

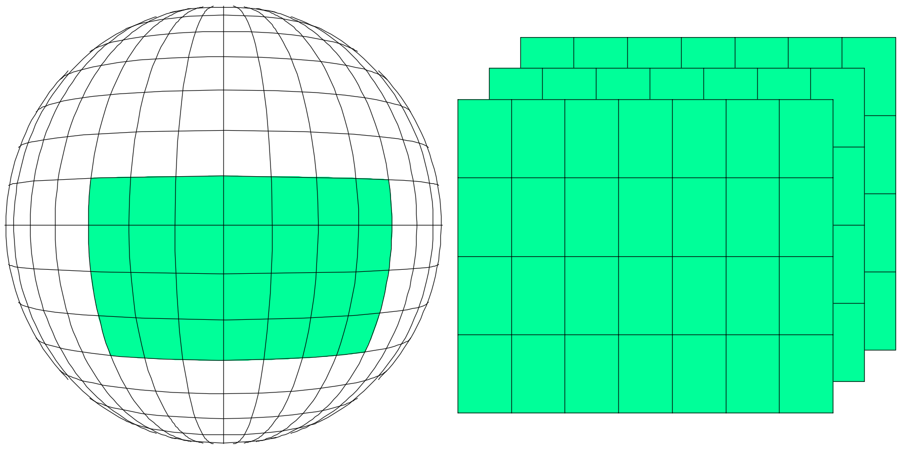


Figure 2.2 Gridded x-y-z data

2.3.3 Data in equal-node elements

`REAL` arrays `XARR(NPTS)`, `YARR(NPTS)` and `ZARR(NPTS)` contain the (x, y, z) coordinates of `NPTS` nodes, and `INTEGER` array `I2ARR(NNODES,NELS)` contains indices of `XARR`, `YARR` and `ZARR` which define a structure of `NELS` elements, each containing `NNODES` nodes (see Figure 2.3). This data configuration can be plotted using `VSZ` and `VSZU`.

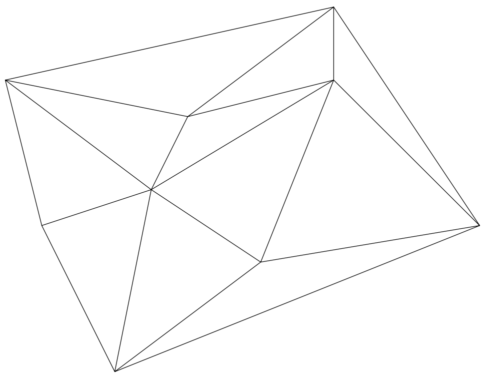


Figure 2.3 Data structured into equal-node elements

2.3.4 Data in non-equal-node elements

(x, y, z) coordinates can be structured into planar elements containing different numbers of nodes. *SIMPLEPLOT ViSualization* does not recognise any general data structure to represent such data, but pictures can be constructed using polygon fill subroutines, `VSPGFL` and `VSPGTN` for each separate element of the structure (see Figure 2.4).

2.3.5 Data on a 3-D grid

`REAL` array `V3ARR(NX,NY,NZ)` contains a regular grid of values (*eg.* density) at every combination of `NX` equally spaced x values, `NY` equally spaced y values, and `NZ` equally spaced z values. `V3ARR` is a function of x, y, and z – for every x-y-z triple there is a corresponding `V3ARR` value (see Figure 2.5). Corresponding data structures are covered by the `PV*` subroutines in the SIMPLEPLOT Volumes section. An IsoSurface through this data configuration can be plotted using `VSIS` and `VSISU`.

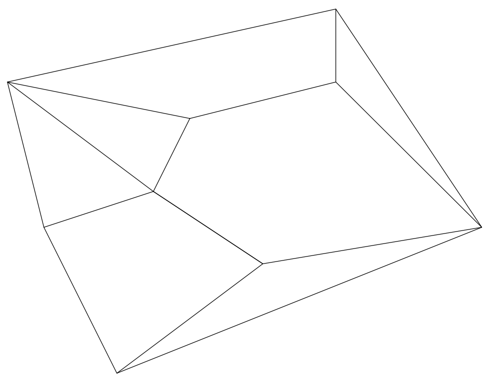


Figure 2.4 Data structured into unequal-node elements

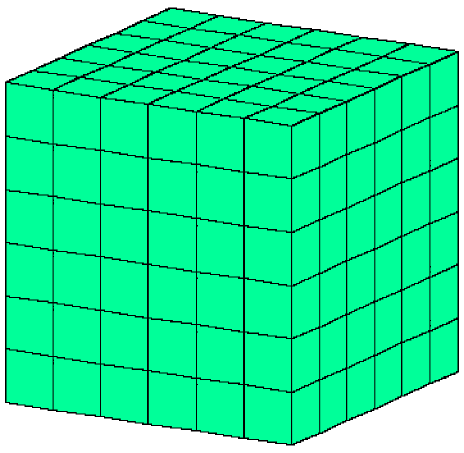


Figure 2.5 Data on a 3-D grid

2.4 Missing data

Missing data]

Sometimes data are not available at all points in the plotting area; for example, a population graph of an island would produce meaningless figures for areas of sea, or there are areas which are so desolate that reliable statistics concerning their physical properties are not available. When these gaps in data arise, it is preferable to show them as gaps in any graphical representation.

A special value can be used in a data set to indicate to SIMPLEPLOT that there are no valid data at that point; this is then represented in the picture as a hole (or gap) in the surface. In Figure 2.6 the rectangular hole represents missing data.

Before any holes can be defined, the program must establish what value to use as a 'no-data' value – a value can be specified using `NODATA`, or `QNODAT` can be called to inquire the current 'no-data' value:

`NODATA(RVAL)` specifies the value, `RVAL`, which SIMPLEPLOT is to interpret as a 'no-data' value. By default, 1.0×10^{-20} is interpreted as this value.

`QNODAT(RVAL)` inquires the value which has been set internally as a 'no-data' value (either by default or a prior call to `NODATA`). The argument of `QNODAT` must be the name of a `REAL` variable which receives the 'no-data' value.

'No data' values affect all plotting and related activities – all plotting routines ignore coordinates which correspond to the 'no-data' value. If `NODATA` is called to specify your own choice of the 'no-data' value,

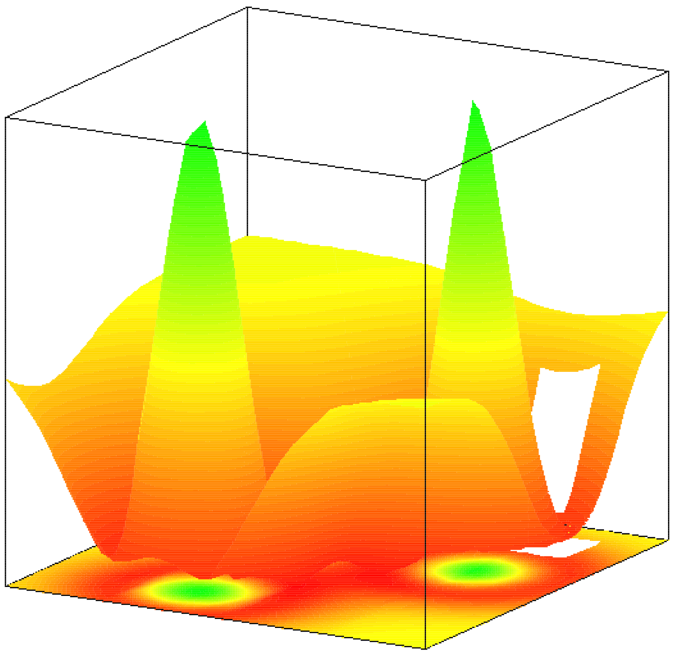


Figure 2.6 Missing data values on a surface

care should be taken to choose a value outside the range of normal coordinates.

3. Scales

This chapter covers the following topics:

3.1 Data scales

All points are assumed to lie within a 3-dimensional Limiting Box — `XMIN` to `XMAX` on the x-axis, `YMIN` to `YMAX` on the y-axis, `ZMIN` to `ZMAX` on the z-axis (Figure 3.1). The scales are set so that all points within these limits can be plotted.

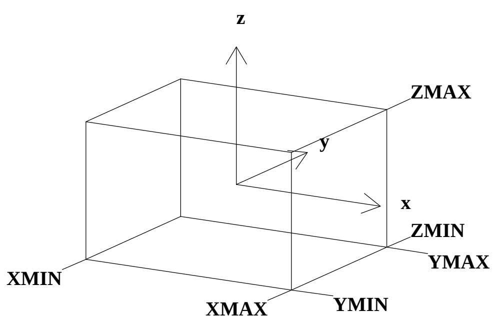


Figure 3.1 3-D limits for 3-D data

The Limiting Box is set relative to 3-D limits defined by the user. 3-D limits can be specified by subroutine `VS3DLM` and inquired by `QV3DLM`. If no limits have been specified, the data scales are defined as being -1.0 to $+1.0$ on each axis.

All points within the Limiting Box can be plotted. However, if *Translation*, *Magnification* or *Rotation* is applied to a point, its (x, y, z) coordinates change. (For more information on *Affine Transformation*, see Appendix T).

Thus, although a point lies within the specified 3-D limits, once `VSTRAN`, `VSMAG` or `VSROT` has been called, it may no longer be plottable.

3.2 Plotting scales

Plotting scales]

The 2-D plotting scales are set so that the Limiting Box touches each edge of the picture. Figure 3.2 shows the plotting area which contains a Limiting Box.

All 3-D points which can be represented within the 2-D plotting scales can be plotted, even those which lie outside the Limiting Box.

Two-dimensional scales are defined in *Normalized coordinates*, relative to the *Bounding Sphere*. The Bounding Sphere is the area which contains all possible rotations of the Limiting Box. It has a radius of $\sqrt{3}$. In other words, all points within the Limiting Box have x and y values between $-\sqrt{3}$ and $+\sqrt{3}$. These limits apply in all cases except following a call to `VSFULL(2)` (see chapter 5).

The current 2-D scales can be inquired by a call to `QVSCAL`. By default, the limits always lie between $\pm\sqrt{3}$, although the range is usually far less than this.

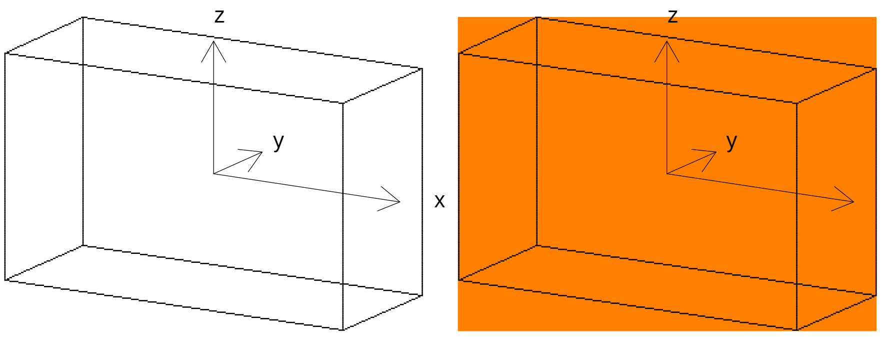


Figure 3.2 2-D limits for 3-D data

3.3 Fit type

Fit type]

By default, the length of each axis is in proportion to the user scales. A box with limits 0 to 4 in x and y and 0 to 2 in z has a z-axis which is half the length of the other axes. Figure 3.3 shows such a box.

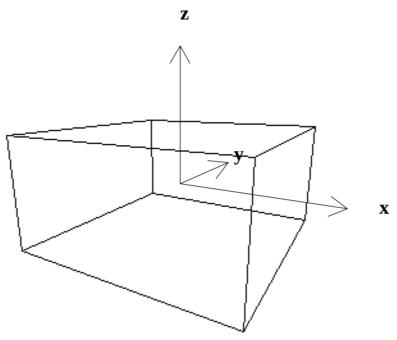


Figure 3.3 x-y-z scales all comparable (default)

This is acceptable as long as the ranges have similar orders of magnitude. However, it is not suitable for a graph which represents 0 to 100 decibels against 0° to 0.001° against 100,000 to 200,000 km. Similarly, a surface which showed x and y ranges of 0 to 10 miles and spot heights of 10,000 to 20,000 metres would not be meaningful if all scales were directly comparable.

Either `VSFITP` or `VSFIT` may be called to specify the relative lengths of the axes.

- Subroutine `VSFITP` specifies that the physical lengths of axes are in proportion to each other. For example, following a call to `VSFITP(2.0, 2.0, 1.0)`, the Limiting Box would be like that in Figure 3.3, regardless of the data limits.
- Subroutine `VSFIT` allows the user to define which scales use comparable units. If different units are used for each scale, the Limiting Box is defined as a cube. If two scales are comparable, the length of the third is equal to the larger of the other two.

Figure 3.4 shows the effect of specifying different fit types on a cuboid. The dotted box is a cube which

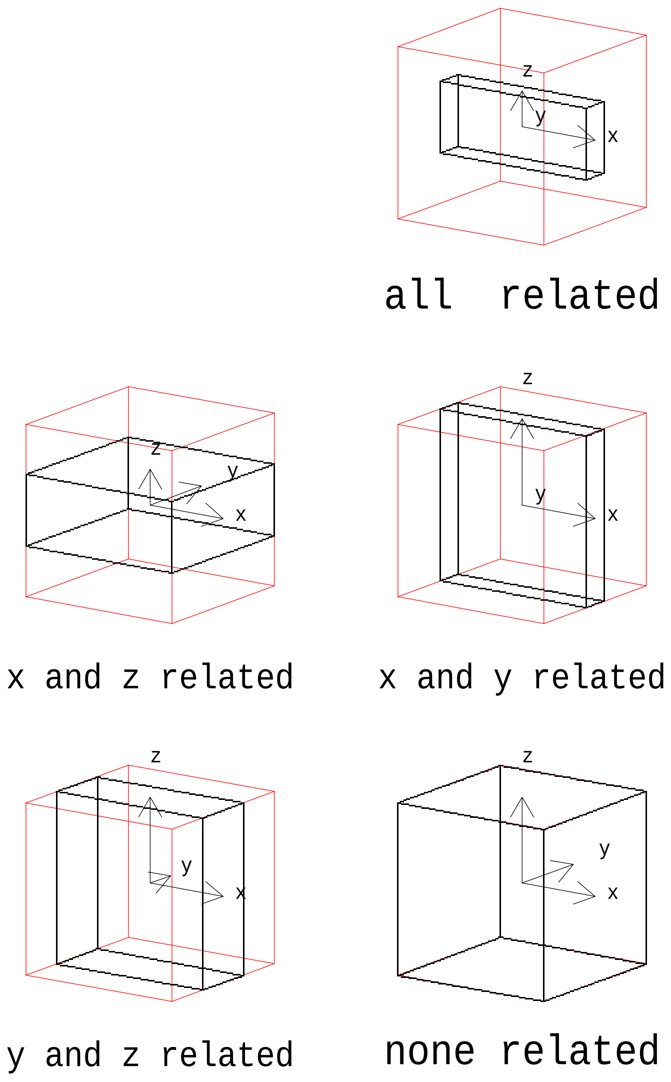


Figure 3.4 Changing the relationship between scales with`VSFIT`

usually has the coordinates -1.0 to $+1.0$ along each axis (the exception to this is following a call to `VSFULL(2)`, see chapter 5).

The Limiting Box fits within the dotted cube, so that at least one axis has the scale -1.0 to $+1.0$. The other axes are in the range $-R$ to R, where $0.0 < R \leq 1.0$.

The first picture (all scales related) is the default. On subsequent pictures, the scale which is not in proportion has the same length as the longer of the other two. This means that the Limiting Box is a cuboid. In the final picture, with no scales related, the Limiting Box is a cube.

4. Viewing

This chapter covers the following topics:

4.1 Viewing direction
- Horizontal Viewing Angle
- Vertical Viewing Angle

4.2 Depth

4.3 Perspective

4.4 Viewing from an (x, y, z) point

4.5 Limiting Viewing Position

4.1 Viewing direction

The angle from which an object is viewed is called the *Viewing Angle*. It has 2 components: the Horizontal Viewing Angle θ (theta), and the Vertical Viewing Angle ϕ (phi).

Figure 4.1 shows a typical Limiting Box. The first picture is the default view, with $\theta = 30°$ and $\phi = 15°$. Although the x, y and z axes are, as always, perpendicular to each other, they appear at an angle to the viewer.

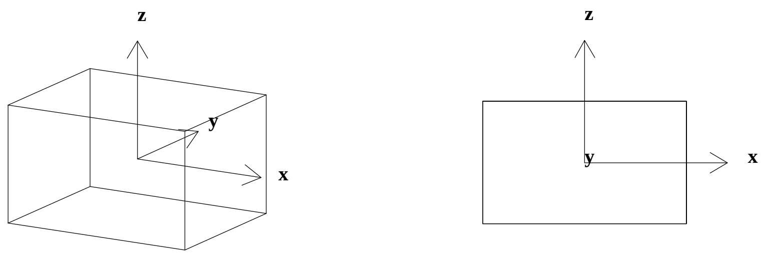


Figure 4.1 Changing the Viewing Angle

The second picture has a Viewing Angle of ($\theta = 0°$, $\phi = 0°$), representing the box viewed along the y-axis. The x and z axes appear as a horizontal and a vertical line. The y-axis goes into the page.

Figure 4.2 shows the position of θ and ϕ relative to the x-y-z axes. At $\theta = 0°$ or $180°$, $x = 0.0$; at $\theta = 90°$ or $270°$, $y = 0.0$; at $\phi = 0°$, $z = 0.0$.

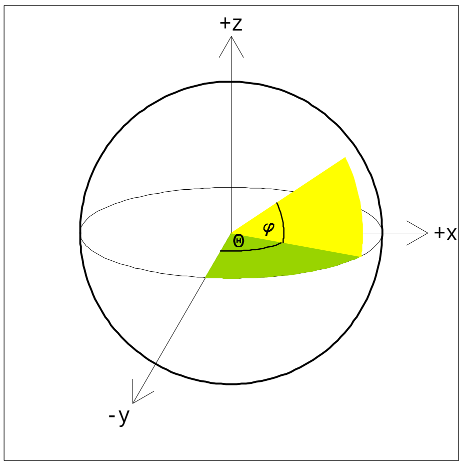


Figure 4.2 Viewing Angle (θ and ϕ)

4.1.1 Horizontal Viewing Angle

Figure 4.3 shows the effect of changing the *Horizontal Viewing Angle* (θ), which is analogous to passing over the Earth along the Equator. The x, y and z axes remain perpendicular to each other.

In these figures, a constant value of ϕ has been chosen which shows the box slightly tilted.

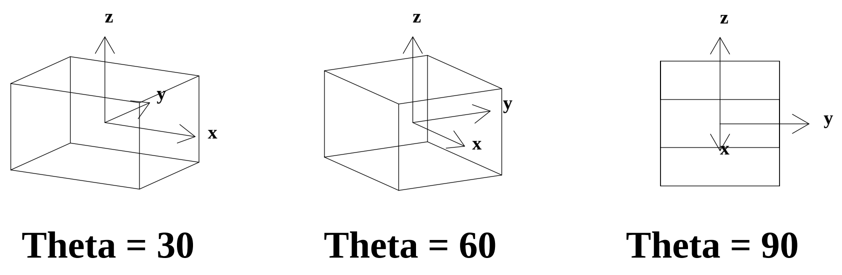


Figure 4.3 Changing the Horizontal Viewing Angle (θ)

4.1.2 Vertical Viewing Angle

Figure 4.4 shows the effect of changing the *Vertical Viewing Angle* (ϕ), which is analogous to passing over the Earth along a line of longitude.

In these figures, a constant value of θ has been chosen which shows the box slightly turned.

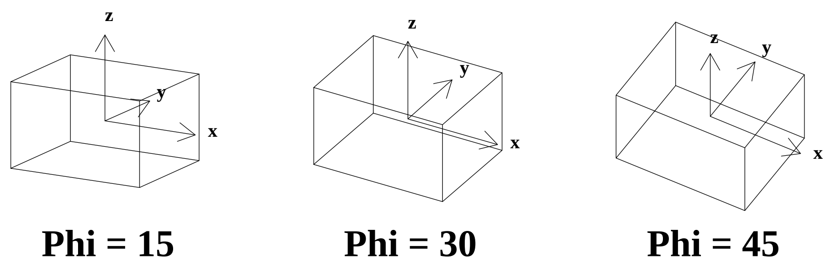


Figure 4.4 Changing the Vertical Viewing Angle (ϕ)

4.2 Depth

Depth]

The Viewing Plane is the plane onto which a 2-D image is projected. It is perpendicular to the Viewing Angle, which it meets at the Origin (*ie.* the centre of the Limiting Box). The depth of a point is its distance from the Viewing Plane, positive beyond the Viewing Plane, and negative towards the viewer. (x, y, z) data coordinates of a point can be converted to 2-D plotting coordinates and depth by subroutine `KVXYD`.

In order to separate the concept of depth from other parameters, consider the case when both the Horizontal Viewing Angle and the Vertical Viewing Angle are zero as in the second picture in Figure 4.1. In this case, the 2-D plotting coordinates are equal to the equivalent 3-D data coordinates x and z, and the Viewing Plane is the same as the $x - z$ plane.

Depth, like the 2-D scale, is measured in *Normalized Coordinates*, relative to the *Bounding Sphere*. All points within the Limiting Box have a depth between $-\sqrt{3}$ and $+\sqrt{3}$.

Figure 4.5 shows a Limiting Box which has been tilted at 45° angles in θ and ϕ. Each picture shows one plane of the box shaded according to depth values, *ie.* shading gets darker as depth increases. A side plane is shaded in the first picture, a front plane in the second, and the base plane in the third.

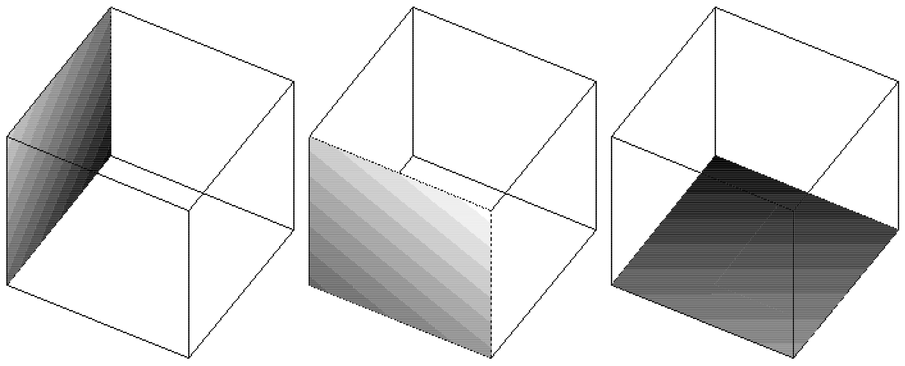


Figure 4.5 Shading the side, front and base according to depth

4.3 Perspective

Perspective]

When objects are viewed in *perspective*, close objects appear larger than similar sized objects which are further away. Perspective may be defined by setting a *Radius* measured in *Normalized coordinates* in the Viewing Direction. The combination of Radius with Viewing Angle is called the *Viewing Position*. The Viewing Position is specified by subroutine `VSVRTP`.

By default, all objects are viewed without perspective. This means that the Radius is infinite, and the size of an object is the same regardless of its depth.

For example, Figures 4.6 and 4.7 show the Limiting Boxes from Figure 4.1 drawn without and with perspective. In the first picture of Figure 4.6, the front face of a cuboid occupies the same area as the back face. In the second picture, the application of perspective means that the sides of the Limiting Box are no longer parallel. These lines, and all lines which are parallel to one of the axes meet at a point in the distance. This is called *3-point perspective*.

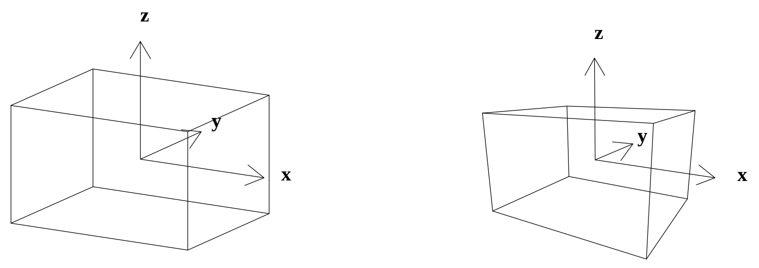


Figure 4.6 Limiting box, with and without perspective

In the first picture of Figure 4.7, the front and back faces line up. In the second picture, the application of perspective means that the nearer face appears considerably larger than the back face. This difference in size increases as the Radius gets smaller.

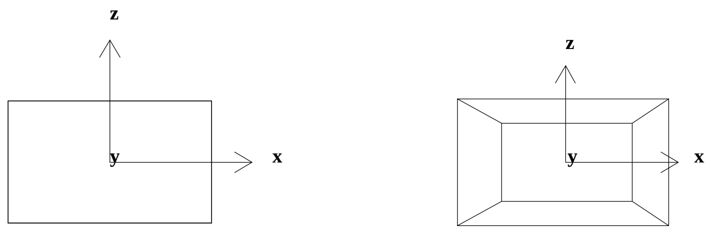


Figure 4.7 Viewing down the y-axis, with and without perspective

The size of a projected area depends on its depth. As the radius increases, the difference in size between areas with different depths decreases. At infinity, there is no difference at all.

4.4 Viewing from an (x, y, z) point

When viewing from a point with fixed coordinates, Viewing Angle and Radius are not appropriate parameters for specifying Viewing Position. It is also possible to specify the Viewing Position of an object from a specified (x, y, z) point (subroutine `VSVXYZ`).

4.5 Limiting Viewing Position

It is only possible to set scales which cover all points in the Limiting Box when the Viewing Position is further from the Origin (*ie.* the centre of the Limiting Box) than the furthest plottable point. When the Viewing Position is closer, some points within the Limiting Box are behind the viewer, and others are projected to infinity. To avoid unsatisfactory Viewing Positions, the Radius specified by subroutine `VSVRTP` should be greater than $\sqrt{3}$, or the (x, y, z) point specified by subroutine `VSVXYZ` should be chosen so that its distance from the origin exceeds

$$\sqrt{(\frac{\mathtt{XMAX-XMIN}}{2})^2 + (\frac{\mathtt{YMAX-YMIN}}{2})^2 + (\frac{\mathtt{ZMAX-ZMIN}}{2})^2}$$

where the range of data is `XMIN` to `XMAX`, `YMIN` to `YMAX` and `ZMIN` to `ZMAX`.

5. Relationship between the Limiting Box and the SIMPLEPLOT picture

This chapter covers the following topics:

5.1 Default Fill Type

5.2 Scaling independent of Viewing Angle

5.3 Transforming the Limiting Box

There are various strategies for setting scales to accommodate the Limiting Box within the current picture size:

- The scale is sufficient to accommodate the untransformed Limiting Box with the current Viewing Angle
- The scale is sufficient to contain the equivalent untransformed Limiting Box from all possible Viewing Angles
- The scale is sufficient to accommodate the Transformed Limiting Box with the current Viewing Angle

Subroutine `VSFULL` may be called to specify the required strategy.

5.1 Default Fill Type

Figure 5.1 shows the Limiting Box viewed from two different Viewing Angles. The shaded area shows the plotting limits. By default, or after `VSFULL(0)`, scales are set to maximize the size of the Limiting Box within the available picture size.

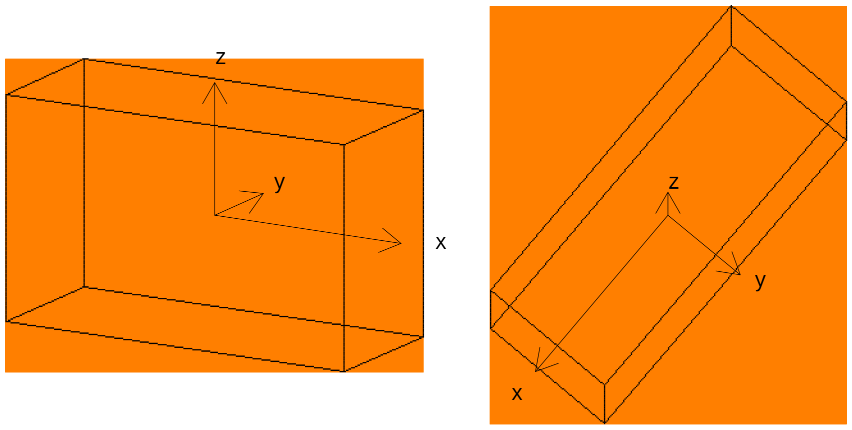


Figure 5.1 Plotting limits from a different angle

When each plot fills the available space, the scales change when the Viewing Angles changes. Figure 5.2 shows the effect of changing the Viewing Angle using the default fill type.

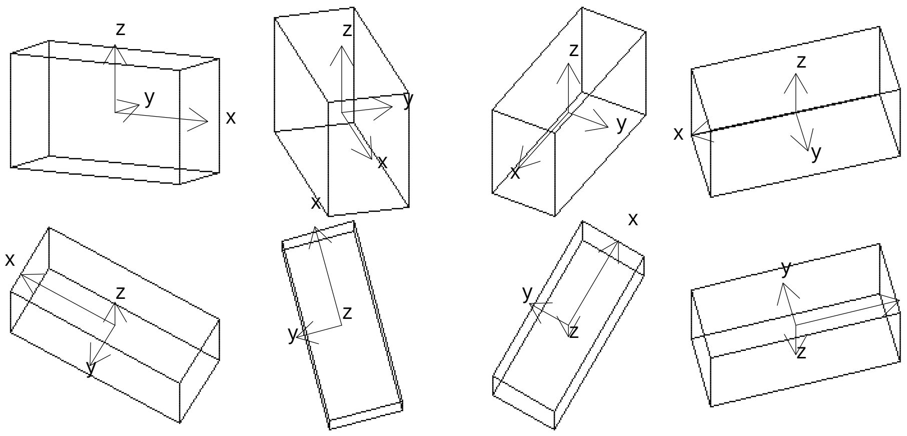


Figure 5.2 Different Viewing Angles with default filling

5.2 Scaling independent of Viewing Angle

`VSFULL(1)` requests that similar scales are maintained for all Viewing Angles given the current Radius. This is advisable when an animated sequence is required, as it ensures that the plotting scales are similar, and all objects within the Limiting Box will fit on the picture for all Viewing Angles.

Figure 5.3 shows the same sequence following a call to `VSFULL(1)`. The images are smaller than with the default fill type, but the scales used are the same for successive plots. When perspective is applied, a constant scale is still retained, but as the Viewing Angle changes, different parts of the plot appear to change size as their distance from the viewer changes.

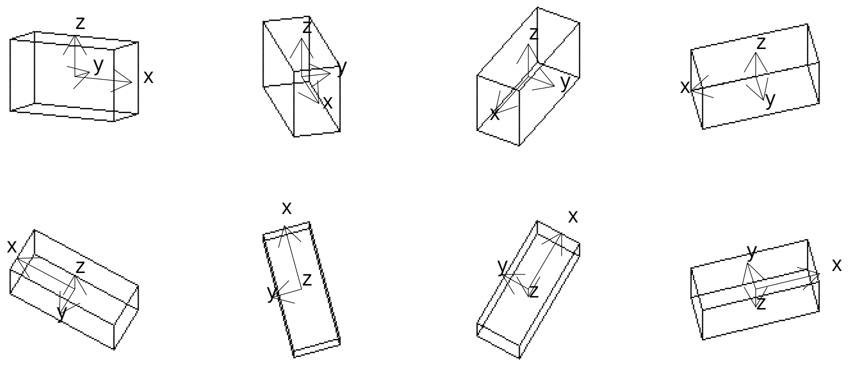


Figure 5.3 Different Viewing Angles with `VSFULL`

5.3 Transforming the Limiting Box

By default, points and objects are contained within the Limiting Box, but when Transformations are applied, points can go out of the Box.

This means that when scales relate to the Limiting Box, applying Transformations can make points within the x-y-z limits *before transformation* unplottable.

`VSFULL(2)` sets scales relative to the Transformed values of the limits set by `VS3DLM`. It is most useful for drawing objects like surfaces, when data lie within fixed limits, but Translation, Magnification and Rotation may be applied.

Like the default filling method, `VSFULL(2)` fills the SIMPLEPLOT picture as much as possible. This means that the scales of pictures with different Viewing Angles or Transformations will not be easily comparable.

Relationship between the Limiting Box and the SIMPLEPLOT picture

6. Colour and lighting

This chapter covers the following topics:

6.1 Colour representation

By default the full range of surface, contour or IsoSurface data are mapped on to colour indices 1 to 15 from the default palette. The range of data or number of colour indices can be changed by calling `VSUTOC`.

The target palette may be set by `VSCHLS` and `VSCRGB`. The palette is set as requested when `VSOUT` is called, if the device can support the number of colours requested. If a device has only a limited set of colours available, and cannot satisfy the requested palette, *SIMPLEPLOT ViSualization* renders the colours by using error diffusion algorithms.

6.2 Colour ranges

The *SIMPLEPLOT ViSualization* palette can contain at most 95 entries. These can be set individually or in blocks. A range of colour palette entries can be interpolated between start and end colours using `VSCRNG`. A range of grey palette entries can be interpolated between start and end intensities using `VSCGS`. The type of interpolation used can be controlled using `VSCIM`

.

6.3 IsoSurface lighting

The values displayed on IsoSurface pictures are determined in one of two ways.

For `VSIS` pictures, the values are light intensity from 0.0 (no light) to 1.0 (maximum light) calculated from the gradient of the surface relative to the direction of the light source. By default this intensity is mapped on to colour indices 1 to 15. A light-sourced picture can be achieved by setting these indices to a grey scale using `VSCGS`. The number of indices used can be controlled by specifying a different mapping using `VSUTOC`. The light source is located at the viewing position by default. It can be moved to a different position by calling `VSLSLD`.

For `VSISU`, pictures the values are interpolated from another data array `U`. By default the range of data in this array is mapped on to colour indices 1 to 15. A smoothly graded effect can be achieved by setting these indices to a colour scale using `VSCRNG`. The number of indices used can be controlled by specifying a different mapping using `VSUTOC`.

For both `VSIS` and `VSISU` the colours used are not interpolated across the surface by default. In order to achieve this effect `VSLSSM` should be called.

7. Polygon edges and surface grids

This chapter covers the following topics:

7.1 Edged polygons

7.2 Regular gridded surfaces

7.1 Edged polygons

By default polygons drawn with `VSPGFL` and `VSPGTN` have no distinct edge. `VSEDGV` and `VSEDGC` can be called to specify that edges should be drawn and to specify their colour index respectively. A colour index of `-1` specifies that subsequent edges should be omitted.

In order to have different coloured edges within one polygon two additional routines are available; `VSPGFE` and `VSPGTE`. These routines allow specification of the edge colour index for each edge, and are independent of `VSEDGV` and `VSEDGC`.

7.2 Regular gridded surfaces

Surface and IsoSurface pictures are, by default, drawn without superimposed grids. Calling `VSEDGV` and `VSEDGC` allows grids to be drawn and their colour index chosen. By default the grids are drawn at the edges of the cells making up the data arrays. For large arrays the edge lines between data elements can obscure the data being drawn, to prevent this `VSEDGI` can be called to specify the starting point for the grids and the offset between them. These offsets and increments can be specified in x and y (and z for IsoSurfaces) independently.

8. Initialization

Initialization involves three categories of attributes.

- *Transformation attributes* are the current Rotation, Translation and Magnification matrix (see Appendix T).
- *Scaling attributes* are those which affect the underlying 2-D scales. These are:
  - 3-D limits
  - Fit type
  - Filling method
  - Viewing position

  For more notes about each of these, consult the relevant documentation.
- *Drawing attributes* affect pen colour, edge visibility *etc.*

Subroutine `VSINIT` can be called with any number between 1 and 7 to reinitialize the above:

1. Transformation attributes
2. Scaling attributes
3. Transformation & Scaling attributes
4. Drawing attributes
5. Transformation & Drawing attributes
6. Transformation & Scaling attributes
7. Transformation, Scaling & Drawing attributes

9. Example Programs

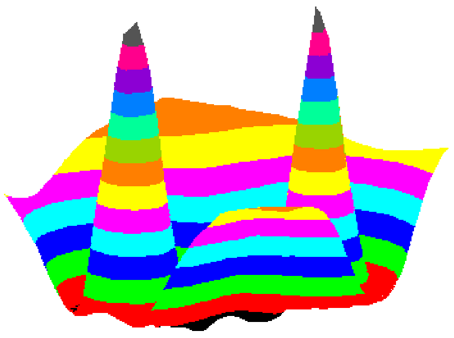
This chapter is made up from a sequence of examples which use *SIMPLEPLOT ViSualization* subroutines with some of the more common SIMPLEPLOT facilities

This page intentionally left blank.

9.1 Surface from a 2-D array

This example illustrates how to:

- start a *SIMPLEPLOT ViSualization* picture,
- draw a surface of data held in a 2-D array,
- output a *SIMPLEPLOT ViSualization* picture.



```
      PROGRAM VIS01
      INTEGER NX, NY
      PARAMETER(NX=36, NY=40)
      REAL Z(NX,NY)                        ! 2-d user matrix
      OPEN(10, FILE='vis01.dat', STATUS='OLD')
      READ(10,*) Z                         ! Read data
      CLOSE(10)
C
      CALL VSNEW                           ! ViSualization NEW picture
        CALL VSRG(Z, NX, NY)               ! Draw surface Z
      CALL VSOUT                           ! ViSualization OUTput
C
      CALL ENDPLT                          ! Close SIMPLEPLOT
      END
```

Example 1. A surface from a 2-D array

Explanation of subroutines

`VSNEW` starts a new SIMPLEPLOT picture, and applies all scaling requests.

When the first SIMPLEPLOT subroutine is called, the first diagnostic message is produced; for example:

```
(SIMPLEPLOT Mark 2-15(001)F)
```

`VSRG(Z,NX,NY)` draws a surface picture of the data in the `NX` × `NY` array `Z` into the raster image. By default the full range of `Z` values is mapped on to colour indices 1 to 15. The image is not displayed yet.

`VSOUT` must be called to output the image.

`ENDPLT` must be called when plotting is complete. It empties any plotting buffers which are in use, closes the plotting device, triggers the output of diagnostic messages still outstanding for the last picture,

```
(END OF PICTURE)
```

and then outputs the following messages:

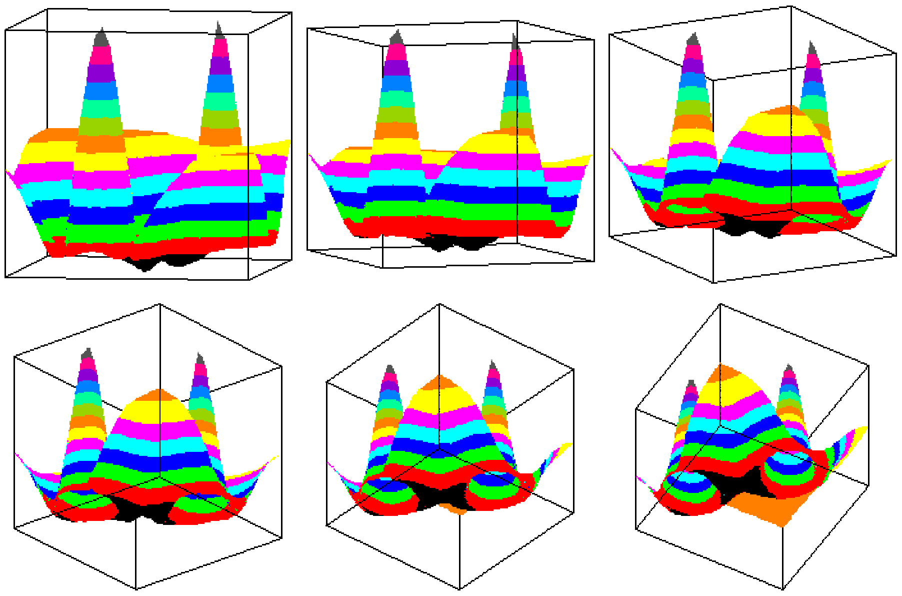
```
    (DEVICE CLOSED)
    (SIMPLEPLOT CLOSED)
```

The subroutines for producing *SIMPLEPLOT ViSualization* pictures can be used in conjunction with any of the standard SIMPLEPLOT facilities described in the *SIMPLEPLOT Primer* or the *SIMPLEPLOT Reference manual*.

9.2 Viewing a surface from different positions

This example illustrates how to:

- vary the Viewing Position of a surface,
- draw the Limiting Box round each image.



```
      PROGRAM VIS02
      INTEGER I, NX, NY
      PARAMETER(NX=36, NY=40)
      REAL Z(NX,NY)                  ! 2-D user matrix
      REAL THETA, PHI
      OPEN(10, FILE='vis01.dat', STATUS='OLD')
      READ(10,*) Z
      CLOSE(10)
C
      CALL GROUP(3,2)                ! 6 pictures on PAGE (3x2)
      CALL MARGIN(0.2)               ! smaller margin
      CALL VSLBOX(1)                 ! request Limiting Box
C
      DO 10 I=1,6
        THETA = I*10.0               ! 10, 20, 30, ...
        PHI = 15.0-I*10.0            ! 5, -5, -15, ...
        CALL VSVRTP(0.0, THETA, PHI) ! Viewing Position no perspective
        CALL VSNEW                   ! ViSualization NEW picture
          CALL VSRG(Z, NX, NY)       ! ViSualization plot from ReGular grid
        CALL VSOUT                   ! ViSualization OUTput image
10    CONTINUE
C
      CALL ENDPLT
      END
```

Example 2. Varying the Viewing Position

Explanation of subroutines

`GROUP(NHORIZ,NVERT)` specifies how subsequent pictures are to be grouped.

`MARGIN(CMS)` specifies the size in centimetres of margins between pictures. The default 2 cm margin is reduced in this example to allow larger images on the page.

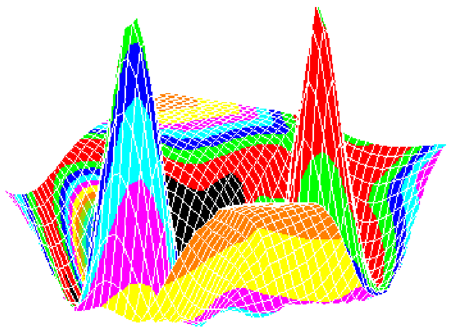
`VSLBOX(LEVEL)` specifies whether the Limiting Box is to be drawn round every subsequent image when `VSNEW` is called.

`VSVRTP(RADIUS,THETA,PHI)` specifies the viewing position from the Origin (0.0,0.0,0.0). When `RADIUS`=0.0, the application of perspective is inhibited. `THETA` is the Horizontal Viewing Angle in degrees, and `PHI` is the Vertical Viewing Angle in degrees. The change in Viewing Position takes effect with the next call of `VSNEW`.

9.3 Contours of one data set on the surface of another

This example illustrates how to:

- draw the contours of one data set on the surface of another,
- draw the element configuration.



```
      PROGRAM VIS03
      INTEGER NX, NY
      PARAMETER(NX=36, NY=40)
      REAL U(NX,NY), Z(NX,NY)               ! 2-d user matrices
      OPEN(10, FILE='vis01.dat', STATUS='OLD')
      READ(10,*) Z, U                       ! Read 2 sets of data
      CLOSE(10)
C
      CALL VSEDGV(1)                        ! Set EDGe Visibility on
      CALL VSEDGC(0)                        ! Set EDGe Colour to index 0
      CALL VSNEW                            ! ViSualization NEW picture
        CALL VSRGU(Z, U, NX, NY)            ! Draw U on surface of Z
      CALL VSOUT                            ! ViSualization OUTput
C
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 3. Two data sets

Explanation of subroutines

`VSEDGV(IVIS)` sets the *visibility* of the edge of polygons making up a surface. By setting the edge visibility on, the underlying data structure can be seen. This gives a useful visual clue to the shape of the underlying surface when the contour levels represent a different variable.

`VSEDGC(ICIX)` sets the colour index of edges. Colour index 0 draws the edge of elements in background.

`VSRGU(Z2ARR,U2ARR,NX,NY)` draws contours of a variable tabulated in array `U2ARR` on the surface representing array `Z2ARR`.

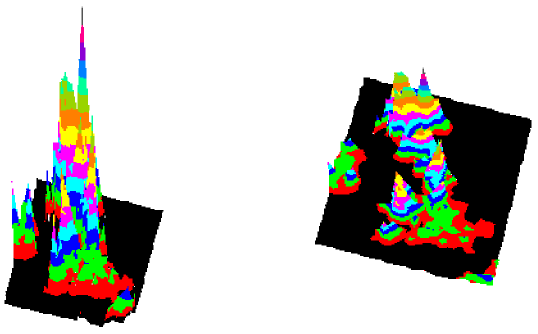
9.4 Setting an appropriate fit type

This example illustrates how to:

- change the relationship between x-y-z scales.
- set the 3-D scales of the Limiting Box,

This example shows heights above sea level in Britain. The units of height are different from the units expressing their locations. The first picture shows the default fitting – the comparatively large z range reduces the intelligibility of the diagram.

`VSFIT` is called before the second picture to specify that only the x and y axes are in proportion to the scales. The z-axis is set equal in length to the longer of the x and y axes.



```
      PROGRAM VIS04
      INTEGER NX, NY
      PARAMETER(NX=30, NY=30)
      REAL U(NX,NY), UMIN, UMAX
C
      OPEN(10, file='vis04.dat', status='OLD')
      READ(10,*) U
      CLOSE(10)
C
      CALL GROUP(2, 1)
      CALL LIMEXC(U, NX*NY, UMIN, UMAX)
      CALL VS3DLM(1.0, REAL(NX), 1.0, REAL(NY), UMIN, UMAX)
      CALL VSVRTP(10.0, 15.0, 75.0)
C
      CALL VSNEW
        CALL VSRG(U, NX, NY)       ! default map of Britain
      CALL VSOUT
C
      CALL VSFIT('XY')             ! only x and y scales proportional
      CALL VSNEW
        CALL VSRG(U, NX, NY)       ! map with new fit type
      CALL VSOUT
C
      CALL ENDPLT
      END
```

Example 4. Setting an appropriate fit type

Explanation of subroutines

`LIMEXC(RARR,NARR,VARMIN,VARMAX)` (LIMits EXClusive) scans the values in any array (ignoring 'no-data' values) to find the minimum and maximum values.

`VS3DLM(XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX)` defines the 3-D coordinate scales for subsequent drawing.

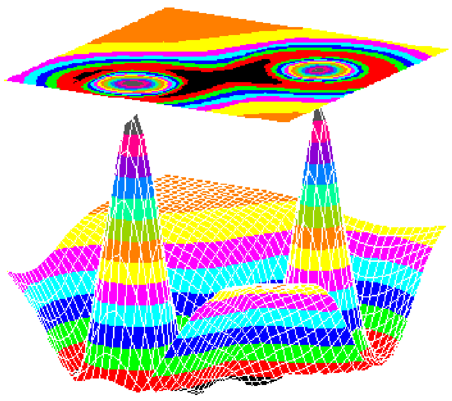
`VSFIT(CHAXES)` specifies which of the x-y-z scales relate to each other. By default the ranges specified by `VS3DLM` are allocated similar scales. One unit in x is the same as 1 unit in y and in z. `VSFIT` specifies which of the coordinates are in similar units, to set the relative axis length.

9.5 A contour map on a plane above a surface

This example illustrates how to:

- draw a surface and contour map of the same data in a single image,
- specify scales when the x, y and z scales are not related.

A contour map is drawn onto a plane by setting up a planar z matrix, and by drawing the contours of the data onto the surface of the plane.



Explanation of subroutines

`VSFIT(' ')` specifies that none of the scales match, so each axis is the same length.

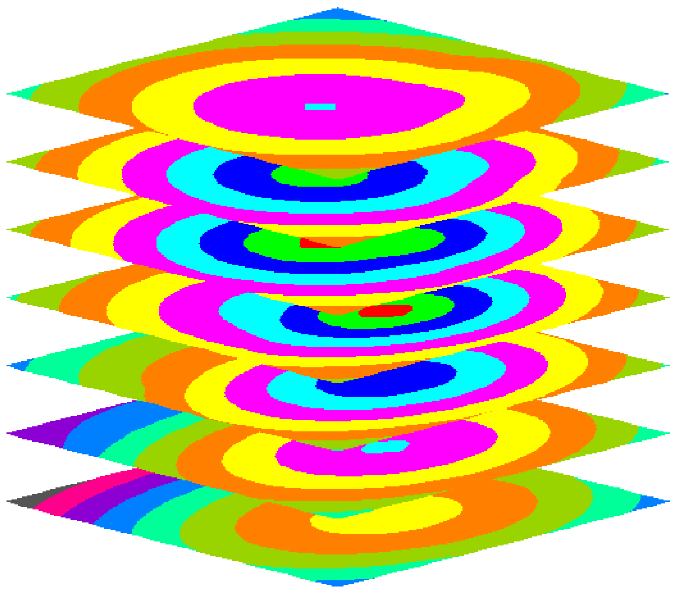
```
      PROGRAM VIS05
      INTEGER NX, NY
      PARAMETER(NX=36, NY=40)
      REAL Z1(NX,NY), Z2(NX,NY)           ! 2-d user matrices
      REAL ZMIN, ZMAX, ZTOP
      OPEN(10, FILE='vis01.dat', STATUS='OLD')
      READ(10,*) Z1
      CLOSE(10)
C
      CALL LIMEXC(Z1, NX*NY, ZMIN, ZMAX)  ! Find range of Z1
      ZTOP = ZMAX+(ZMAX-ZMIN)*0.1         ! + 10% range at top
      CALL VS3DLM(1.0, REAL(NX), 1.0, REAL(NY), ZMIN, ZTOP) ! 3D LiMits
      CALL VSFIT(' ')                     ! Don't match axes to scales
      CALL DEFSLC(ZTOP, NX, NY, Z2)       ! Set all Z2 to ZTOP
      CALL VSEDGC(0)                      ! Set EDGe Colour to index 0
      CALL VSNEW                          ! ViSualization NEW picture
        CALL VSEDGV(1)                    ! Set EDGe Visibility on
        CALL VSRG(Z1, NX, NY)             ! Draw Z1
        CALL VSEDGV(0)                    ! Set EDGe Visibility off
        CALL VSRGU(Z2, Z1, NX, NY)        ! Draw Z1 on Z2 plane
      CALL VSOUT                          ! ViSualization OUTput
      CALL ENDPLT                         ! Close SIMPLEPLOT
      END
C
      SUBROUTINE DEFSLC(ZVAL, NX, NY, ZOUT)
C Define plane onto which to project contours
      REAL ZVAL                           !  IN: Constant z value
      INTEGER NX, NY                      !      Dimensions
      REAL ZOUT(NX, NY)                   ! OUT: Plane
      INTEGER I, J
      DO 10 J=1, NY
        DO 20 I=1, NX
          ZOUT(I,J) = ZVAL
20      CONTINUE
10    CONTINUE
      END
```

Example 5. Surface with contour

9.6 Stacked contour plots from a 3-D matrix

This example illustrates how to:

- define the mapping from data level to colour,
- draw a family of contour plots stacked on top of each other.



Explanation of subroutines

`VSUTOC(UMIN,UMAX,MINCIX,MAXCIX)` specifies how data levels relate to colour indices. In this example, the range of user data values for the 3-D matrix is found by `LIMEXC`, and is mapped on to colour indices 2 to 15. Each separate contour picture then uses the same colour scale.

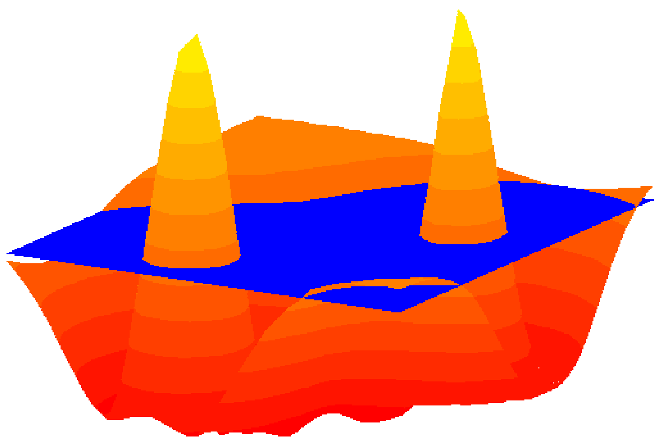
```
      PROGRAM VIS06
      INTEGER NX, NY, NZ
      PARAMETER(NX=21, NY=21, NZ=21)
      REAL Z(NX,NY), U(NX,NY,NZ), UMIN, UMAX
      INTEGER K
      OPEN(10, FILE='vis06.dat', STATUS='OLD')
      READ(10,*) U
      CLOSE(10)
C
      CALL VSVRTP(0.0, -45.0, 15.0)        ! Viewing position
      CALL VS3DLM(1.0, REAL(NX), 1.0, REAL(NY), 1.0,REAL(NZ)) ! 3D LiMits
      CALL LIMEXC(U, NX*NY*NZ, UMIN, UMAX)
      CALL VSUTOC(UMIN, UMAX, 2, 15)       ! Relate User data TO Colours
      CALL VSNEW                           ! ViSualization NEW picture
        DO 10 K= 1, NZ, 3
          CALL DEFSLC(REAL(K), NX, NY, Z)  ! Set all Z to REAL(K)
          CALL VSRGU(Z, U(1,1,K), NX, NY)  ! Draw surface on plane
10      CONTINUE
      CALL VSOUT                           ! ViSualization OUTput
      CALL ENDPLT                          ! Close SIMPLEPLOT
      END
C
      SUBROUTINE DEFSLC( ZVAL, NX, NY, ZOUT )
C Define plane onto which to project contours
      REAL ZVAL                            !  IN: Constant z value
      INTEGER NX, NY                       !      Dimensions
      REAL ZOUT(NX, NY)                    ! OUT: Plane
      INTEGER I, J
      DO 10 J=1, NY
        DO 20 I=1, NX
          ZOUT(I,J) = ZVAL
20      CONTINUE
10    CONTINUE
      END
```

Example 6. Stacked contours

9.7 Composite images

This example illustrates how to:

- define a colour map,
- set the fill colour index,
- fill a polygon,
- draw a composite image made from a surface and a plane.



```
      PROGRAM VIS07
      INTEGER NX, NY, NCORN, IRED, IYELLO, IBLUE
      PARAMETER(NX=36, NY=40, NCORN=4, IRED=16, IYELLO=28, IBLUE=29)
      REAL Z(NX,NY), ZMIN, ZMAX
      REAL XSLICE(NCORN), YSLICE(NCORN), ZSLICE(NCORN)
      DATA XSLICE/1.0, NX, NX, 1.0/
      DATA YSLICE/1.0, 1.0, NY, NY/
      DATA ZSLICE/-2.0, -2.0, -2.0, -2.0/
      OPEN(10, FILE='vis01.dat', STATUS='OLD')
      READ(10,*) Z                          ! Read data
      CLOSE(10)
      CALL LIMEXC(Z, NX*NY, ZMIN, ZMAX)      ! Find data limits
      CALL VS3DLM(1.0, REAL(NX), 1.0, REAL(NY), ZMIN, ZMAX) ! 3D LiMits
C
      CALL VSCRGB(IRED, 1.0, 0.0, 0.0)       ! IRED = red
      CALL VSCRGB(IYELLO, 1.0, 1.0, 0.0)     ! IYELLO = yellow
      CALL VSCRNG(IRED, IYELLO)              ! palette = red to yellow
      CALL VSNEW                             ! ViSualization NEW picture
        CALL VSUTOC(ZMIN, ZMAX, IRED, IYELLO) ! Relate User data TO Colour
        CALL VSRG(Z, NX, NY)                 ! Draw surface Z
        CALL VSCRGB(IBLUE, 0.0, 0.0, 1.0)    ! Set colour IBLUE to blue
        CALL VSFILC(IBLUE)                   ! FILl with Colour IBLUE
        CALL VSPGFL(XSLICE, YSLICE, ZSLICE, NCORN) ! Plane at z = -2.0
      CALL VSOUT                             ! ViSualization OUTput
      CALL ENDPLT                            ! Close SIMPLEPLOT
      END
```

Example 7. A slice through a surface

Explanation of subroutines

`VSCRGB(ICIX,RED,GREEN,BLUE)` defines the target colour for the colour index given, in terms of red, green and blue components. When the device has limited colour capability, an attempt is made to render different colours using the available resources.

`VSCRNG(ICIX1,ICIX2)` specifies a graded range of target colours in the *SIMPLEPLOT ViSualization* palette. Colours are interpolated between index `ICIX1` and `ICIX2`, in this example from red to yellow.

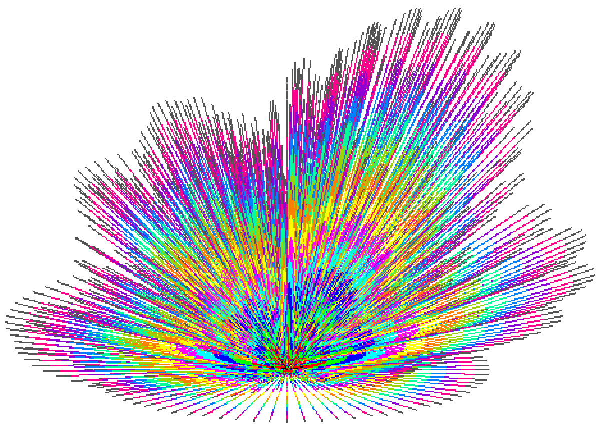
`VSFILC(ICIX)` fills subsequent polygons and blocks with colour index `ICIX`.

`VSPGFL(XARRAY,YARRAY,ZARRAY,NPTS)` draws a polygon held in arrays `XARRAY`, `YARRAY` and `ZARRAY`, dimensioned `NPTS`. In this example, the plane at $z = -2.0$ is filled in blue.

9.8 Data from (r, θ, ϕ) coordinates - a hedgehog

This example illustrates how to:

- convert from (r, θ, ϕ) coordinates to (x, y, z),
- tint a simple polyline.



Explanation of subroutines

`VSPLTN(XARRAY,YARRAY,ZARRAY,UARRAY,NPTS)` joins `NPTS` points by straight lines. The positions of the points are specified by (`XARRAY(i)`, `YARRAY(i)`, `ZARRAY(i)`), and the data values which are represented by colours, are specified by `UARRAY(i)`. Colours are changed along the lines to show the behaviour of `UARRAY(i)`.

In the example, `VSPLTN` is called repeatedly to join individual points to $(0, 0, 0)$. The u value is equivalent to the distance of the point from the centre. At $(0, 0, 0)$, $u = 0.0$. This example also shows how to convert data from (R, θ, ϕ) to equivalent (x, y, z) values $(R \cos\theta \cos\phi,\ R \sin\theta \cos\phi,\ R \sin\phi)$.

```
      PROGRAM VIS08
      INTEGER I, J, NLONG, NLAT, NPT
      REAL RADIAN, UMIN, UMAX, RLONG, RLAT, R
      PARAMETER(RADIAN=3.14/180.0, R=32.0, NLONG=73, NLAT=18, NPT=2)
      REAL U(NLONG,NLAT), X(NPT), Y(NPT), Z(NPT), RADIUS(NPT)
      OPEN(10, FILE='vis08.dat', STATUS='OLD')
      READ(10,*) U
      CLOSE(10)
      CALL LIMEXC(U, NLONG*NLAT, UMIN, UMAX)  ! Find data limits
      CALL VS3DLM(-R, R, -R, R, 0.0, R)       ! Set 3D LiMits
      CALL VSNEW                              ! ViSualization NEW picture
        X(1) = 0.0                            ! Define centre ...
        Y(1) = 0.0                            ! ... point ...
        Z(1) = 0.0                            ! ... of data
        RADIUS(1) = 0.0                       ! Central radius
        DO 20 I=1,NLONG
          RLONG = RADIAN * (I-37) * 5.0       ! Longitude in degrees
          DO 10 J=1,NLAT
            RLAT = RADIAN * (J-1) * 5.0       ! Latitude in degrees
            RADIUS(2) = U(I,J) - UMIN         ! Length-> power gain
            X(2) = RADIUS(2)*COS(RLONG)*COS(RLAT)
            Y(2) = RADIUS(2)*SIN(RLONG)*COS(RLAT)
            Z(2) = RADIUS(2)*SIN(RLAT)
            CALL VSPLTN(X, Y, Z, RADIUS, NPT) ! TiNt PolyLine
 10       CONTINUE
 20     CONTINUE
      CALL VSOUT                              ! ViSualization OUTput
      CALL ENDPLT                             ! Close SIMPLEPLOT
      END
```

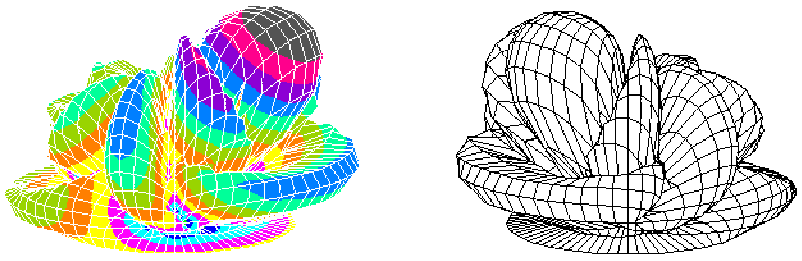
Example 8. A hedgehog

9.9 A surface and a mesh from (x, y, z, u) coordinates

This example illustrates how to:

- draw a surface from arrays of grids of data,
- rotate a *SIMPLEPLOT ViSualization* image,
- draw the mesh only of a surface, applying hidden line removal.

The data set used in this example is a modified version of that used in the previous example. In this case, data are in the form (x, y, z, u), rather than (r, θ, ϕ).



```
PROGRAM VIS09
REAL RMIN, RMAX, RVAL
INTEGER NLONG, NLAT
PARAMETER(RVAL=32.0, NLONG=73, NLAT=18)
REAL X(NLONG,NLAT), Y(NLONG,NLAT), Z(NLONG,NLAT), R(NLONG,NLAT)
OPEN(10, FILE='vis09.dat', STATUS='OLD')
READ(10,*) X, Y, Z, R
CLOSE(10)
CALL GROUP(2,1)
CALL VS3DLM(-RVAL, RVAL, -RVAL, RVAL, 0.0, RVAL)     ! 3D LiMits
CALL VSEDGV(1)                           ! Set EDGe Visibility on
CALL VSEDGC(0)                           ! Set EDGe Colour to index 0
CALL VSNEW                               ! ViSualization NEW picture
  CALL VSXYZU(X, Y, Z, R, NLONG, NLAT)   ! Draw R on surface X,Y,Z
CALL VSOUT                               ! ViSualization OUTput
CALL VSROT('Z3', -100.0)                 ! Rotate z by -100 degrees
CALL VSEDGC(1)                           ! Restore EDGe Colour
CALL LIMEXC(R, NLONG*NLAT, RMIN, RMAX)   ! Find data limits
CALL VSUTOC(0.0, RMAX, 0, 0)             ! Relate User data TO Colour 0
CALL VSNEW                               ! ViSualization NEW picture
  CALL VSXYZ(X, Y, Z, NLONG, NLAT)       ! Draw surface X,Y,Z
CALL VSOUT                               ! ViSualization OUTput
CALL ENDPLT                              ! Close SIMPLEPLOT
END
```

Example 9. A 3-D surface

Explanation of subroutines

`VSXYZU(X2ARR,Y2ARR,Z2ARR,U2ARR,NX,NY)` draws the surface joining the points (`X2ARR(i,j)`, `Y2ARR(i,j)`, `Z2ARR(i,j)`) over the grid defined by lines of constant i against lines of constant j, with the colour representing `U2ARR(i,j)`.

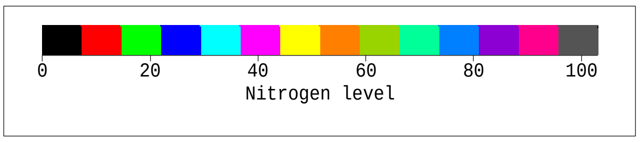
VSROT(CHAXIS,ROTATE) rotates the *SIMPLEPLOT ViSualization* image by ROTATE degrees around axis CHAXIS. In this example, the image is rotated by $-100°$ around the z-axis.

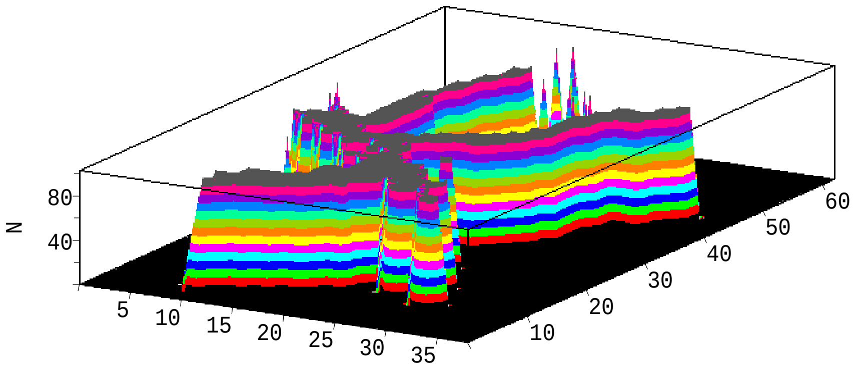
VSXYZ(X2ARR,Y2ARR,Z2ARR,NX,NY) draws the same surface. All data values are represented by a single colour, in this case, background, which is specified by VSUTOC(..., 0, 0). The mesh is drawn by calling VSEDGV(1) to set the edge visibility on.

9.10 Adding axes and keys

This example illustrates how to:

- add simple axes,
- add a key to user data levels,
- control the relative lengths of the x, y and z axes.





```
PROGRAM VIS10
INTEGER NX, NY, I, J
PARAMETER(NX=38, NY=62)
REAL Z(NX,NY), ZMIN, ZMAX
OPEN(UNIT=10, FILE='vis10.dat', STATUS='OLD')
READ(10, *) ((Z(I, J), I = 1, NX), J = NY, 1, -1)
CLOSE(10)
CALL VSLBOX(1)                          ! Request Limiting BOX
CALL LIMEXC(Z, NX*NY, ZMIN, ZMAX)       ! Find data LIMits
CALL VSFITP(REAL(NX), REAL(NY), 10.0)   ! FIT Proportional scales
CALL VS3DLM(0.0, REAL(NX), 0.0, REAL(NY), ZMIN, ZMAX)
CALL VSNEW                              ! ViSualization NEW picture
  CALL VSRG(Z, NX, NY)                  ! Surface from ReGular grid
CALL VSOUT                              ! ViSualization OUTput
CALL VSAXDR(' ', ' ', 'N')              ! AXis DRaw
CALL VSK7H('T', 'C', ZMIN, ZMAX, 'Nitrogen level') ! Key Horizontal
CALL ENDPLT                             ! Close SIMPLEPLOT
END
```

Example 10. A surface from a 2-D array

Explanation of subroutines

`VSFITP(XPROPN,YPROPN,ZPROPN)` specifies the relative physical lengths of the x, y and z axes. This example keeps the lengths of the x and y axes in proportion to the data, and sets a smaller z-axis length.

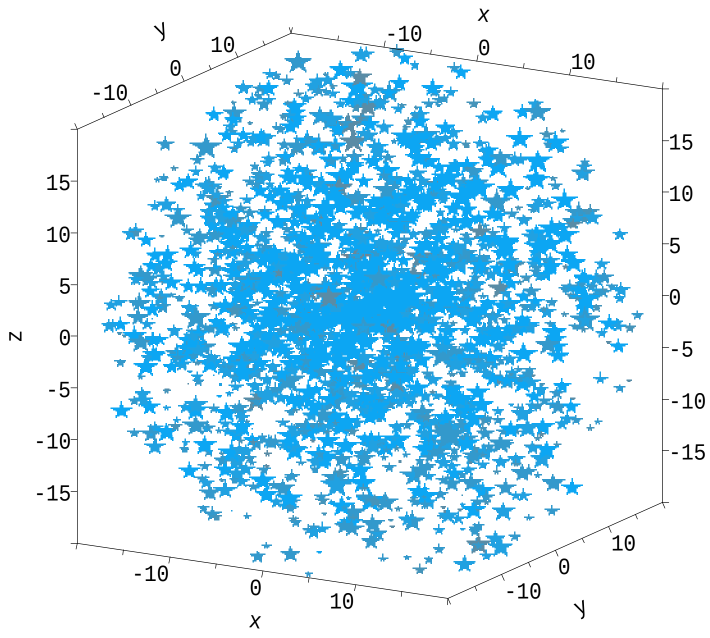
`VSAXDR(CAPX,CAPY,CAPZ)` draws 3-dimensional axes on the current *SIMPLEPLOT ViSualization* image. `CAPX`, `CAPY` and `CAPZ` are added as axis labels. Individual axis properties may be controlled using the appropriate `AX*` subroutines.

`VSK7H(VCHAR,HCHAR,ULEFT,URIGHT,CAP)` draws a complete horizontal key to user data, labelled `CAP`. Samples of the contour colours corresponding to the range `ULEFT` to `URIGHT` are annotated with the data levels. In this example, the key is positioned `'T'`op `'C'`entre.

9.11 Polymarkers

This example illustrates how to:

- draw all six 3-D axes,
- draw polymarkers into a *SIMPLEPLOT ViSualization* image,
- set marker colour and size.



Explanation of subroutines

`AXLOCN(CHAXIS,'B')` specifies that `'B'`oth axes are drawn for the specified scales. `CHAXIS` `'*3'` means that all 3-D axes are specified.

`VSCHLS(ICIX,HUE,BRIGHT,SAT)` defines the target colour for the colour index given. When the device has limited colour capability, an attempt is made to render different colours using the available resources.

`VSPMC(ICIX)` specifies the marker colour index.

`VSPMMG(FACTOR)` multiplies the marker size by `FACTOR`. The size of each marker in this example is scaled to relate to the magnitude of the corresponding star.

`VSPMDR(XARRAY,YARRAY,ZARRAY,NPTS,MKTYPE)` draws `NPTS` polymarkers centred at the (x, y, z) value in arrays `XARRAY`, `YARRAY` and `ZARRAY`. Marker type `MKTYPE` is used. This example uses marker 104, a solid star.

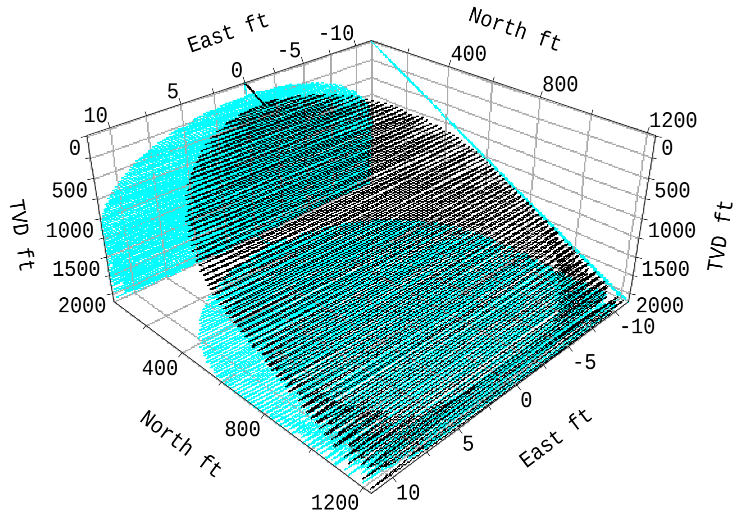
```
      PROGRAM VIS11
      INTEGER NSTARS(6), NPTS, I, J, K
      PARAMETER(NPTS=2151)
      REAL R, THETA, PHI
      REAL X(NPTS), Y(NPTS), Z(NPTS), RMAG(NPTS)
      DATA NSTARS/5, 87, 218, 602, 537, 702/
      OPEN(UNIT=10, FILE='vis11.dat', STATUS='OLD')
      DO 10, I = 1,NPTS                     ! Read position & magnitude
        READ(10,*) R, THETA, PHI, RMAG(I)
        X(I) = R * COS(PHI) * SIN(THETA)    ! Convert from ...
        Y(I) = R * COS(PHI) * COS(THETA)    ! ... (r,theta,phi) ...
        Z(I) = R * SIN(PHI)                 ! ... to (x,y,z)
10    CONTINUE
      CLOSE(10)
      CALL VS3DLM(-20.0, 20.0, -20.0, 20.0, -20.0, 20.0)  ! Set scales
      CALL AXLOCN('*3', 'B')                ! Both sets of axes for x-y-z
      CALL VSNEW                            ! Start picture
        K = 0                               ! Index to marker
        DO 30, J = 1, 6                     ! Draw each group in turn
          CALL VSCHLS(J+9, 320.0, 0.5, 0.15*J) ! Set colour indices 10..15
          CALL VSPMC(J+9)                   ! Set marker colour
          DO 20, I = 1, NSTARS(J)           ! Draw each marker in group
            K = K+1                         ! Next marker in array
            CALL VSPMMG(0.1 * RMAG(K))      ! Set marker size
            CALL VSPMDR(X(K), Y(K), Z(K), 1, 104)  ! Draw single marker
20        CONTINUE
30      CONTINUE
      CALL VSOUT                            ! Output image
      CALL VSAXDR('x', 'y', 'z')            ! Add axes
      CALL ENDPLT
      END
```

Example 11. Polymarkers showing star data

9.12 Axis grids and polylines

This example illustrates how to:

- add grids to axis backdrops,
- project a polyline onto a plane,
- select a polyline colour.



Explanation of subroutines

`VSAXGC(IXCIX,IYCIX,IZCIX)` specifies the colour index for grid lines on each back plane.

`VSPLDR(XARRAY,YARRAY,ZARRAY,NPTS)` draws a polyline held in arrays `XARRAY`, `YARRAY` and `ZARRAY`, dimensioned `NPTS`.

`VSLINC(ICIX)` specifies the colour index for polylines.

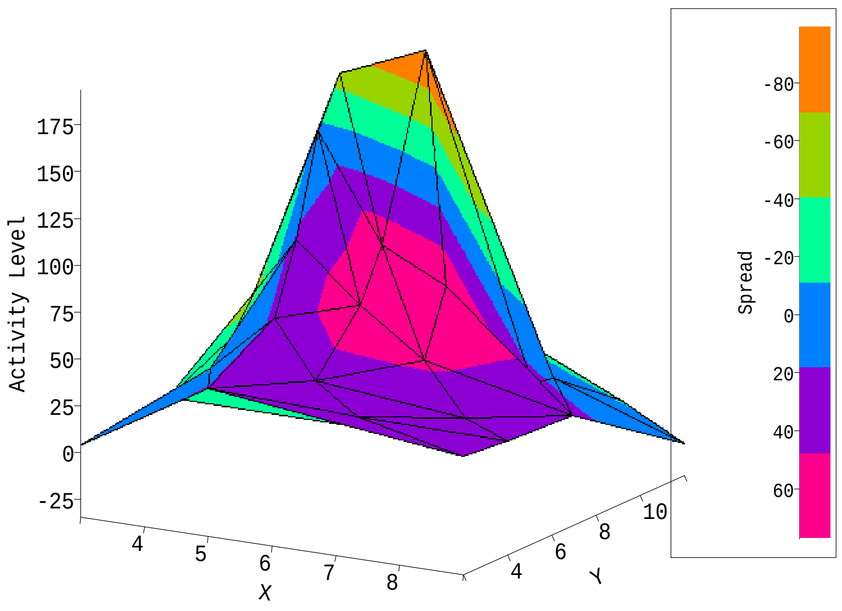
```
      PROGRAM VIS12
      INTEGER MAXPTS, NPTS, I
      PARAMETER(MAXPTS = 1024)
      REAL XDRILL(MAXPTS), YDRILL(MAXPTS), ZDRILL(MAXPTS)
      REAL XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX
      REAL XCONST(MAXPTS), YCONST(MAXPTS), ZCONST(MAXPTS)
C
      OPEN(UNIT=10, FILE='vis12.dat', STATUS='OLD')
        READ(10, *) XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX
        READ(10, *) NPTS
        READ(10, *) (ZDRILL(I), XDRILL(I), YDRILL(I), I = 1, NPTS)
      CLOSE(10)
      DO 10, I=1, NPTS                    ! Set constant data for ...
        XCONST(I) = XMIN                  ! ... x ...
        YCONST(I) = YMIN                  ! ... y ...
        ZCONST(I) = ZMAX                  ! ... & z planes
10    CONTINUE
C
      CALL VS3DLM(XMIN, XMAX, YMIN, YMAX, ZMAX, ZMIN) ! Set 3D scales
      CALL VSFITP(1.0, 1.0, 0.5)          ! FIT Proportions of axis lengths
      CALL AXLOCN('*3','B')               ! Both axes for each scale ...
      CALL VSAXGC(15, 15, 15)             ! ...with grids in Colour 15
      CALL VSVRTP(5.0, 135.0, 30.0)       ! Specify viewpoint
      CALL VSNEW                                       ! Start new picture
        CALL VSPLDR(XDRILL, YDRILL, ZDRILL, NPTS)      ! DRaw PolyLine
        CALL VSLINC(5)                                 ! Set LINe Colour
        CALL VSPLDR(XCONST, YDRILL, ZDRILL, NPTS)      ! Project onto x-
        CALL VSPLDR(XDRILL, YCONST, ZDRILL, NPTS)      ! ...   y- ...
        CALL VSPLDR(XDRILL, YDRILL, ZCONST, NPTS)      ! ... & z-plane
      CALL VSOUT                                       ! Flush buffers
      CALL VSAXDR('East ft', 'North ft', 'TVD ft')     ! DRaw AXes
      CALL ENDPLT
      END
```

Example 12. Axis grids and polylines

9.13 Ungridded data

This example illustrates how to:

- draw a *SIMPLEPLOT ViSualization* picture from ungridded data,
- add a vertical key.



```
PROGRAM VIS13
INTEGER NPTS, MELEM
PARAMETER(NPTS=29, MELEM = 3*NPTS)
INTEGER I, NELEM
INTEGER IELEM(3,MELEM), NEIGH(3,MELEM)
REAL X(NPTS), Y(NPTS), Z1(NPTS), Z2(NPTS)
REAL XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX, UMIN, UMAX
OPEN(10, FILE='vis13.dat', STATUS='OLD')
  READ(10,*) (X(i), Y(i), Z1(i), Z2(i), i=1,NPTS)
CLOSE(10)
CALL ZZORDR(X, Y, NPTS, IELEM, NEIGH, NELEM, MELEM)  ! Structure data
CALL VSFITP(1.0, 1.0, 1.0)          ! Equal length axes
CALL VSEDGV(1)                      ! Draw element boundaries
CALL LIMEXC(X, NPTS, XMIN, XMAX)    ! Find range of x...
CALL LIMEXC(Y, NPTS, YMIN, YMAX)    ! ... and Y ...
CALL LIMEXC(Z2, NPTS, ZMIN, ZMAX)   ! ...
CALL LIMEXC(Z1, NPTS, UMIN, UMAX)   ! Draw contours over range ...
CALL VS3DLM(XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX)
CALL PAGMRG(2.0, 2.0, 1.0, 0.0)     ! PiCture MaRGin
CALL VSNEW
  CALL VSUTOC(UMIN, UMAX, 8, 14)    ! ... of Z1
  CALL VSZU(X, Y, Z2, Z1, NPTS, IELEM, 3, NELEM)  ! Draw Z1 on Z2
CALL VSOUT
CALL VSAXDR('X', 'Y', 'Activity Level')      ! Draw axes
CALL VSK7V('C', 'F', UMIN, UMAX, 'Spread')  ! Vertical Key Centre Right
CALL ENDPLT
END
```

Example 13. Ungridded data

Explanation of subroutines

`ZZORDR(XARR,YARR,NPTS,I2ARR,N2ARR,NELEMS,ISIZE)` organizes ungridded data into triangular elements and neighbours.

`PAGMRG(CMS,RCMS,BCMS,TCMS)` sets the margin for the left, right, bottom and top of the page. Each argument is measured in centimetres: in this example, `CMS` `RCMS` are each set to 2cm, and `BCMS` is set to 1cm to allow space for the axis labels.

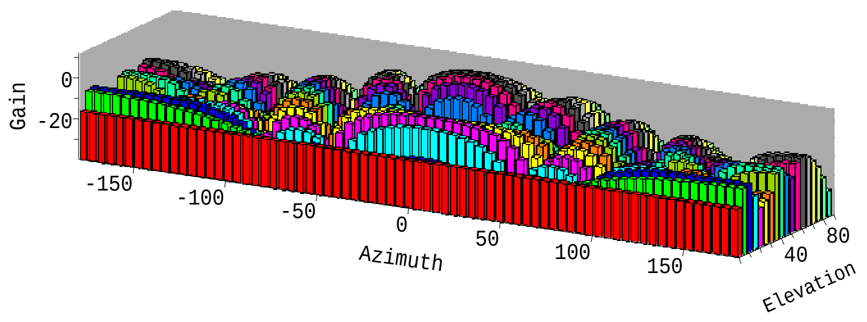
`VSZU(XARR,YARR,ZARR,UARR,NPTS,I2ARR,NNODES,NELS)` draws contours from 3-D data structured into elements.

`VSK7V(VCHAR,HCHAR,UTOP,UBOTTM,CAP)` draws a complete vertical key, labelled `CAP`. Samples of the contour colours corresponding to the range `UTOP` to `UBOTTM` are given and annotated with the data levels. In this example, the key is positioned vertically `'C'`entre, `'F'`ollowing the picture.

9.14 3-D bar chart

This example illustrates how to:

- draw a 3-D bar chart from an array of numeric data,
- set a base level for bars,
- fill an axis back plane.



```
      PROGRAM VIS14
      INTEGER NLONG, NLAT
      PARAMETER(NLONG=73, NLAT=18)
      REAL U(NLONG, NLAT), UMIN, UMAX
      INTEGER ISHAD(NLAT), ICOL
      OPEN(10, FILE='vis08.dat', STATUS='OLD')
        READ(10,*) U
      CLOSE(10)
      DO 10 ICOL=1,NLAT                          ! For NLAT colours ...
        ISHAD(ICOL) = ICOL+1                     ! .. start at colour 2
10    CONTINUE
      CALL SQSHAD(ISHAD, NLAT)                   ! .. for SHADing SeQuence
      CALL LIMEXC(U, NLONG*NLAT, UMIN, UMAX)     ! Find LIMits EXClusive
      CALL VS3DLM(-180.0, 180.0, 0.0, 90.0, UMIN, UMAX)  ! Set 3D LiMits
      CALL VSEDGV(1)                             ! EDGe Visibility on
      CALL VSAXFC(15, 15, 15)                    ! Colour for rear planes
      CALL VSNEW                                 ! ViSualization NEW picture
        CALL VSBRFC(2)                           ! Fill Colours - columns
        CALL VSBRBZ(UMIN)                        ! Set BaR Base Z value
        CALL VSBRFL(U, NLONG, NLAT, 1)           ! BaR FiLl
      CALL VSOUT                                 ! ViSualization OUTput
      CALL VSAXDR('Azimuth', 'Elevation', 'Gain')  ! AXis DRaw
      CALL ENDPLT                                ! Close SIMPLEPLOT
      END
```

Example 14. 3-D bar chart

Explanation of subroutines

SQSHAD(IARR,NARR) specifies a sequence of NARR shading patterns. In this example, the patterns in IARR are set to start from pattern 2, thus avoiding pattern 1 (black on this device).

VSEDGV(1) specifies that outlines are drawn around filled areas including 3-D bars.

VSAXFC(IXFCIX,IYFCIX,IZFCIX) specifies the colour index for each of the x, y and z planes.

VSBRFC(ICODE) specifies how bars are to be coloured. VSBRFC(2) shades each column in a different colour. The colours used are those specified by SQSHAD.

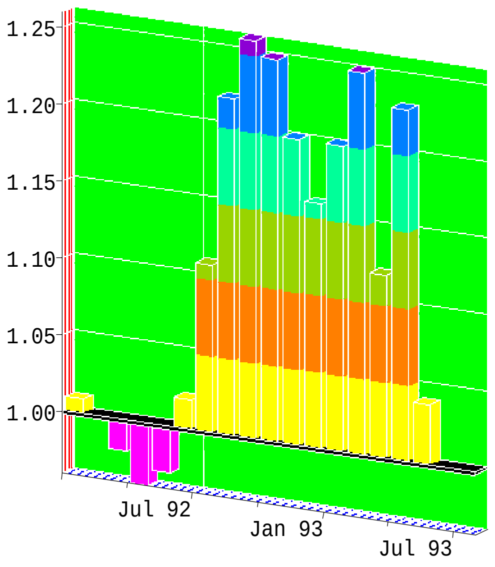
`VSBRBZ(ZVAL)` sets the base level for a bar chart. By default, bars are drawn to $z = 0.0$.

`VSBRFL(Z,NX,NY,NSETS)` draws a 3-D bar chart. The bar chart is drawn from data held in `Z`, dimensioned `NX` $\times$ `NY` $\times$ `NSETS`. If `NSETS` > 1, more than one data set is stacked.

9.15 Tinted bar chart

This example illustrates how to:

- convert (year,month,day) to internal time format,
- draw a time-based axis,
- draw a tinted 3-D bar chart.



```
      PROGRAM VIS15
      INTEGER NBARS
      PARAMETER(NBARS=19)
      REAL RATIO(NBARS), DAY1, DAYN, RMIN, RMAX, BARWD
      DATA RATIO /1.01, 1.0,  0.98, 0.96, 0.97, 1.02, 1.11, 1.22, 1.26,
     +      1.25, 1.2,  1.16, 1.2,  1.25, 1.12, 1.23, 1.04, 1.0,  1.0/
      CALL KTREAL('M', 1992, 1, 1.0, DAY1)  ! Convert from (y,m,d) to ...
      CALL KTREAL('M', 1993, 8, 1.0, DAYN)  ! ... date in internal format
      CALL LIMEXC(RATIO, NBARS, RMIN, RMAX) ! Find data LIMits
      BARWD = (DAYN-DAY1)/NBARS             ! Width of bars
      CALL VS3DLM(DAY1, DAYN, 0.0, BARWD, RMIN, RMAX) ! 3D LiMits
      CALL VSFIT('XY')                      ! x:y scales in proportion
      CALL VSAXFC(2, 3, 4)                  ! AXis plane Fill Colour
      CALL AXLBTP('X3', 'D')                ! TiMe scale on X 3d axis
      CALL AXSBDV('X3', DAY1, DAYN-DAY1)    ! No AXis SuBDiVisions
      CALL VSAXGC(0, 0, -1)                 ! AXis Grid Colour
      CALL VSNEW                            ! ViSualization NEW picture
        CALL VSEDGV(1)                      ! EDGe Visibility ON
        CALL VSEDGC(0)                      ! EDGe Colour index 0
        CALL VSBLFL(DAY1, DAYN, 0.0, BARWD, 0.999, 1.001)  ! Draw shelf
        CALL VSBRBZ(1.0)                    ! Set BaR Base at Z=1.0
        CALL VSUTOC(0.9, 1.3, 5, 13)        ! Relate User data TO Colours
        CALL VSBRTN(RATIO, NBARS, 1)        ! Draw BaR chart (TiNted)
      CALL VSOUT                            ! ViSualization OUTput
      CALL AXSBDV('X3', 0.0, 0.0)           ! Restore default subdivisions
      CALL AXCLR('Y3', 1)                   ! AXis CLear
      CALL VSAXDR(' ', ' ', ' ')            ! AXis DRaw
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 15. Tinted 3-D bar chart

Explanation of subroutines

`KTREAL(UNITS,IVAL1,IVAL2,RVAL,VALUE)` converts from a (year,month,day) triple to data in internal SIMPLEPLOT time format.

`AXSBDV(CHAXIS,OFFSET,DELTA)` specifies that tickmarks on axis `CHAXIS` are drawn at interval `DELTA` from `OFFSET`.

`AXSBDV` is called twice in this example – before `VSNEW`, to specify that no grids are drawn from the x-axis, projected onto the z plane, and before `VSAXDR` to restore the default when the axis is drawn.

`VSBLFL(X1,X2,Y1,Y2,Z1,Z2)` draws a single bar from `X1` to `X2` in x, `Y1` to `Y2` in y, and `Z1` to `Z2` in z. `VSBLFL` is used in this example to draw a shelf at $z = 1.0$.

`AXLBTP('X3','D')` specifies that the 3-D x-axis is annotated with `'D'`ates.

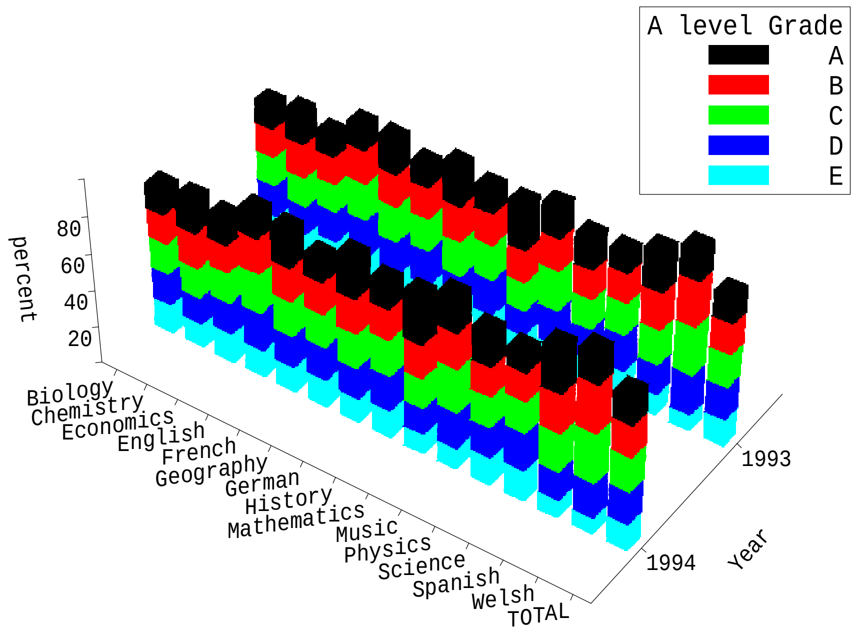
`VSBRTN(Z,NX,NY)` draws a tinted bar chart, relating the data level to colour.

`AXCLR(CHAXIS,1)` specifies that axis labels and annotation are omitted from axis `CHAXIS`.

9.16 Stacked bar chart

This example illustrates how to:

- draw a stacked 3-d bar chart,
- control the bar size.



Explanation of subroutines

`VSBRFC(3)` shades different data sets with different colours.

`VSBRSZ(X1,X2,Y1,Y2)` sets the bar size. The size in this example is calculated to give bars which are roughly square.

`VSBRFL(Z,NX,NY,NSETS)` draws a stacked 3-D bar chart if `NSETS` > 1.

`AXLBSL(CHAXIS,'H')` specifies `'H'`orizontal labels on axis `CHAXIS`.

`AXTXT7(CHAXIS,W1,STEP,LABARR,NARR)` adds the labels in `LABARR` to axis `CHAXIS`.

`VSKEYS('T','R',LABARR,NARR,CAP)` writes a key for `NARR` distinct data levels, and positions it `'T'`op `'R'`ight. Unlike `VSK7H` and `VSK7V`, which draw a tinted grid on a u-axis, `VSKEYS` uses separate shading patterns, and is unaffected by the `AX*` subroutines.

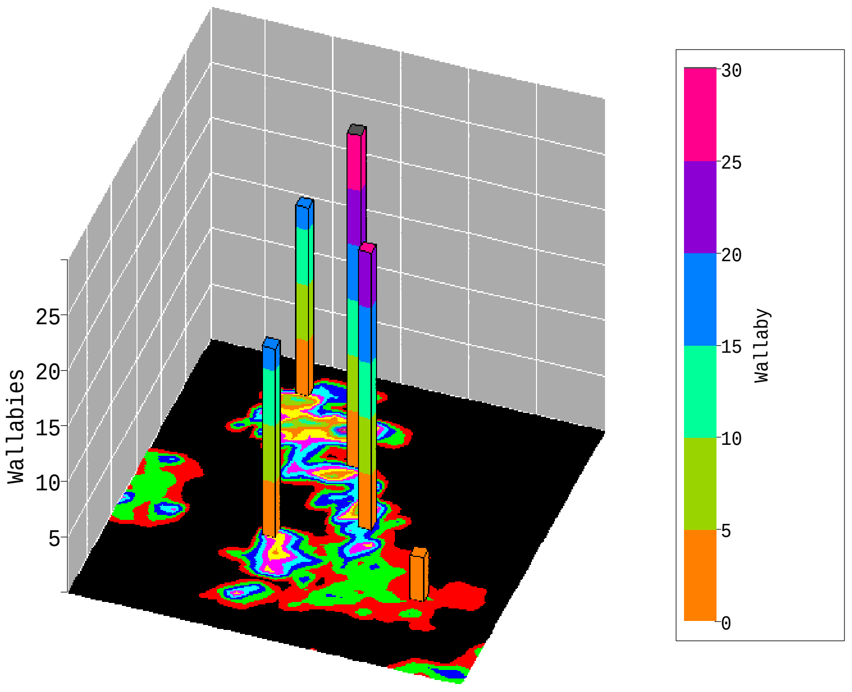
```
      PROGRAM VIS16
      REAL XPROP, YPROP, ZPROP, XWIDTH, YWIDTH
      PARAMETER(XPROP=50.0, YPROP=30.0, ZPROP=20.0)
      INTEGER NGRADE, NSUBJ, NYEAR, I, J, K
      PARAMETER(NSUBJ=15, NGRADE=5, NYEAR=2)
      REAL GRADES(NSUBJ,NYEAR,NGRADE)
      CHARACTER*18 SUBJCT(NSUBJ)
      CHARACTER*1 GRADE(NGRADE)
      DATA GRADE/'A', 'B', 'C', 'D', 'E'/
      OPEN(UNIT=10,FILE='vis16.dat',STATUS='OLD')
      DO 10 I=1,NSUBJ                      ! Read data by subject
        READ(10,'(A)') SUBJCT(I)
        READ(10, *) ((GRADES(I,J,K), K=1,NGRADE), J=1,NYEAR)
10    CONTINUE
      CLOSE(10)
      CALL VSVRTP(10.0, 35.0, 40.0)        ! Set View by R Theta Phi
      CALL VS3DLM(0.0, REAL(NSUBJ), 1994.5, 1992.5, 0.0, 100.0)
      CALL VSFITP(XPROP, YPROP, ZPROP)           ! FiT Proportional axes
      CALL VSNEW                           ! ViSualization NEW picture
        CALL VSBRFC(3)                     ! Set BaR Fill Colour
        XWIDTH = REAL(NSUBJ)/XPROP
        YWIDTH = REAL(NYEAR)/YPROP
        CALL VSBRSZ(0.5-XWIDTH,0.5+XWIDTH,0.5-YWIDTH,0.5+YWIDTH)
        CALL VSBRFL(GRADES, NSUBJ, NYEAR, NGRADE)   ! FiLl BaRs
      CALL VSOUT                           ! ViSualization OUTput
      CALL AXSBDV('Y3', 1.0, 1.0)          ! Set y AXis SuBDiVisions
      CALL AXCLR('X3', 1)                  ! CLeaR 3d X AXis
      CALL VSAXDR(' ', 'Year', 'percent')  ! AXis DRaw
      CALL AXTXT7('X3', 0.5, 1.0, SUBJCT, NSUBJ)  ! Draw AXis TeXT
      CALL VSKEYS('T', 'R', GRADE, NGRADE, 'A level Grade')  ! SHaded KEYS
      CALL ENDPLT                          ! Close SIMPLEPLOT
      END
```

Example 16. Stacked 3-D bar chart

9.17 Drawing an individual bar

This example illustrates how to:

- draw a single bar on a plane,
- add a single axis,
- alter the u-axis, used on *SIMPLEPLOT ViSualization* keys.



Explanation of subroutines

`VSBLTN(X1,X2,Y1,Y2,Z1,Z2,U1,U2)` draws a single bar from `X1` to `X2` in x, `Y1` to `Y2` in y, and `Z1` to `Z2` in z. The bar is tinted from the colour derived from `U1` at $z =$ `Z1` to the colour derived from `U2` at $z =$ `Z2`.

`AXIS7(CHAXIS, CAP)` draws a single axis, labelled `CAP`. If `CHAXIS` is `X3`, `Y3` or `Z3`, the 3-D x, y or z-axis is drawn.

`AXLOCN('U3','F')` specifies that the u-axis follows the key. Axis labels are drawn to the right of the key drawn by `VSK7V`.

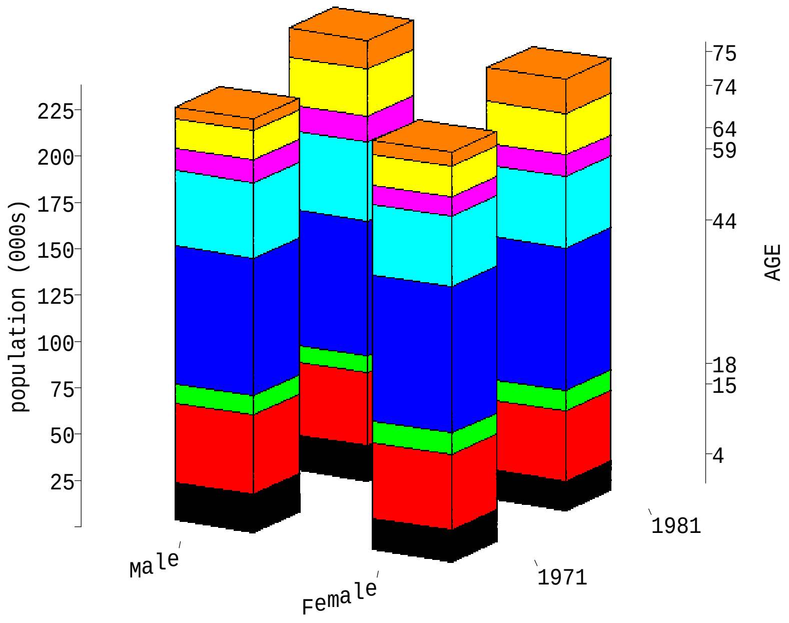
```
      PROGRAM VIS17
      INTEGER NX, NY, NALL, I
      PARAMETER(NX=30, NY=30, NALL=NX*NY)
      REAL GB(NX,NY), BASE(NX,NY)          ! Heights for GB; base for map
      REAL X(5), Y(5), Z(5)
      DATA X/24.0, 12.0, 18.0, 15.0, 9.0/   ! Positions for bars in ...
      DATA Y/9.0, 12.0, 15.0, 21.0, 27.0/   ! ... matrix coordinates
      DATA Z/4.0, 17.0, 25.0, 30.0, 17.0/   ! Bar heights
      DATA BASE/NALL*0.0/                   ! Base z value for map
      OPEN(10, FILE='vis04.dat', STATUS='OLD') ! Read ...
        READ(10, *) GB                      ! ... heights above sea level
      CLOSE(10)
      CALL VS3DLM(1.0, REAL(NX), 1.0, REAL(NY), 0.0, 30.0)  ! 3d scales
      CALL VSVRTP(0.0, 20.0, 40.0)          ! Set viewing position
      CALL VSAXFC(15, 15, -1)               ! Specify colour of backdrop
      CALL VSAXGC(0, 0, -1)                 ! Specify grid colour
      CALL VSNEW                            ! Start raster image
        CALL VSRGU(BASE, GB, NX, NY)        ! Draw background map
        CALL VSUTOC(0.0, 30.0, 8, 14)       ! Map bar range to 8..14
        CALL VSEDGV(1)                      ! EDGe Visibility ON
        DO 10, I = 1, 5                     ! Fill each bar in turn
          CALL VSBLTN(X(I)-0.5, X(I)+0.5, Y(I)-0.5, Y(I)+0.5, ! BLock TiNt
     1                0.0, Z(I), 0.0, Z(I)) ! ... (1x1) base from Z = 0.0
10      CONTINUE
      CALL VSOUT                            ! ViSualization OUTput
      CALL AXIS7('Z3', 'Wallabies')         ! Draw z axis
      CALL AXLOCN('U3', 'F')                ! u-axis Following the key
      CALL VSK7V('C', 'F', 30.0, 0.0, 'Wallaby') ! Key Vertical
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 17. Single bar

9.18 Inquiring the size of stacked bars

This example illustrates how to:

- inquire the range of a stacked bar chart,
- invert the colour order for bars.



Explanation of subroutines

`QVBRFL(Z,NY,NY,NSETS,ZMIN,ZMAX)` inquires the z scale required by a *SIMPLEPLOT ViSualization* stacked barchart. `ZMIN` gives the base, and `ZMAX` gives the height of the highest stack.

`VSBRUP(IUP)` specifies the stacking direction of bars. `VSBRUP(1)` specifies that bars are stacked from the bottom upwards.

`AXLAB7(CHAXIS,W,CAP)` writes label `CAP` at data level `W` on the `CHAXIS` axis.

```
      PROGRAM VIS18
      INTEGER NSEX, NYEAR, NAGE, i
      PARAMETER(NSEX=2, NYEAR=2, NAGE=8)
      REAL POP(NSEX, NYEAR, NAGE), ZMIN, ZMAX, ZLEV
      CHARACTER AGE(NAGE)*2, SEX(NSEX)*6, YEAR(NYEAR)*4
      OPEN(UNIT=10, FILE='vis18.dat', STATUS='OLD')
      READ(10, '(A,4F8.3)')(AGE(I),
     +   ((POP(K,J,I), K=1,NSEX), J=1,NYEAR), I=1,NAGE)
      READ(10, '(2(A,1X))') (SEX(I), I=1,NSEX)
      READ(10, '(2(A,3X))') (YEAR(I), I=1,NYEAR)
      CLOSE(10)
      CALL QVBRFL(POP, NSEX, NYEAR, NAGE, ZMIN, ZMAX) ! Inquire bar size
      CALL VS3DLM(0.0, 2.0, 0.0, 2.0, ZMIN, ZMAX)     ! Set 3-d limits
      CALL VSFITP(1.0, 1.0, 1.0)                ! x-y-z axes same length
      CALL VSNEW
        CALL VSBRFC(3)                          ! Colour according to set
        CALL VSEDGV(1)                          ! EDGe Visibility on
        CALL VSBRSZ(0.3, 0.7, 0.3, 0.7)         ! BaR SiZe smaller
        CALL VSBRUP(1)                          ! Reverse driection of bar
        CALL VSBRFL(POP, NSEX, NYEAR, NAGE)     ! BaR FiLl
      CALL VSOUT
      CALL AXTXT7('X3', 0.5, 1.0, SEX, NSEX)    ! Annotate x-axis
      CALL AXTXT7('Y3', 0.5, 1.0, YEAR, NYEAR)  ! Annotate y-axis
      CALL AXIS7('Z3', 'population (000s)')     ! Draw z-axis
      CALL AXCLR('*3', 1)                       ! Clear z and u axes
      CALL AXLOCN('Z3', 'F')                    ! Following the picture ...
      CALL AXIS7('Z3', 'AGE')                   ! ... Draw second z-axis
      ZLEV = 0.0
      DO 100 I=1,NAGE                           ! ... For each stack ...
        ZLEV = ZLEV + POP(NSEX, NYEAR, I)       ! ... find top of bar ..
        CALL AXLAB7('Z3', ZLEV, AGE(I))         ! ... and label
 100  CONTINUE
      CALL ENDPLT
      END
```

Example 18. Annotated key

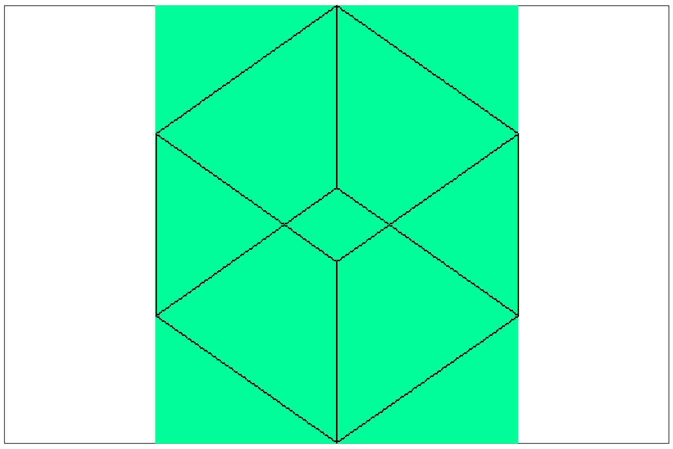
9.19 SIMPLEPLOT 2-D coordinates

VSNEW starts a new picture, and defines 2-D Cartesian scales to contain the whole projected image. QVSCAL inquires the underlying 2-D user scales used. The scale units have a 1:1 aspect ratio, and fill at least one of the picture dimensions.

This example illustrates how to identify the picture limits and the Limiting Box, and how to mix general SIMPLEPLOT software with the ViSualization module.

This example illustrates how to:

- display the picture limits,
- find the underlying SIMPLEPLOT 2-D user scales,
- use these scales to fill a rectangle containing the Limiting Box.



```
      PROGRAM VIS19
      REAL XMIN, XMAX, YMIN, YMAX           ! 2-d user scales
      CALL BOXPIC(.TRUE.)                   ! Request BOX round PICture
      CALL VSLBOX(1)                        ! Request Limiting BOX
C
      CALL VSVRTP(0.0, 45.0, 45.0)          ! Set Viewing position
      CALL VSNEW                            ! ViSualization NEW picture
        CALL QVSCAL(XMIN, XMAX, YMIN, YMAX)         ! Inquire scales
        CALL SHDEBX(XMIN, YMIN, XMAX, YMAX, 10)     ! SHaDE BoX
      CALL VSOUT                            ! ViSualization OUTput
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 19. Using 2-D coordinates

Explanation of subroutines

BOXPIC(.TRUE.) requests that a boundary is drawn when a new picture is started.

QVSCAL(XMIN,XMAX,YMIN,YMAX) inquires underlying SIMPLEPLOT 2-D coordinates.

SHDEBX(X1,Y1,X2,Y2,IPATTN) fills a rectangular area defined by two opposite corners with the fill pattern specified.

9.20 Annotation

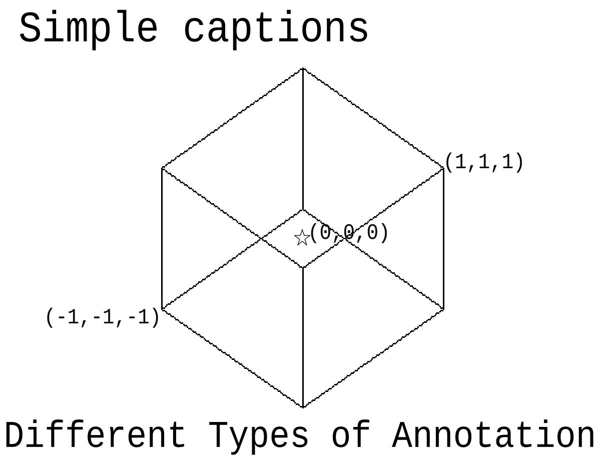
Annotation]

All the text facilities of the SIMPLEPLOT library are available with the ViSualization module.

Text can be positioned descriptively by reference to positions on the picture, group or page, or explicitly using the underlying 2-D coordinate system.

This example illustrates how to:

- annotate using descriptive text positions,
- find the 2-D coordinates of a 3-D point,
- annotate using 2-D coordinates.



```
      PROGRAM VIS20
      REAL X, Y, D                          ! 2-d (x,y) and depth
      CALL VSVRTP(0.0, 45.0, 45.0)          ! Set Viewing position
      CALL VSLBOX(1)                        ! Request Limiting BOX
      CALL VSNEW                            ! ViSualization NEW picture
        CALL KVXYD(1.0, 1.0, 1.0, X, Y, D)    ! Convert to (X,Y,D)
        CALL CP7LB(X, Y, '(1,1,1)')           ! Draw caption at (X,Y)
        CALL KVXYD(0.0, 0.0, 0.0, X, Y, D)    ! Convert to (X,Y,D)
        CALL CP7PT(X, Y, 31, '(0,0,0)')       ! Add caption/marker at (X,Y)
        CALL KVXYD(-1.0, -1.0, -1.0, X, Y, D) ! Convert to (X,Y,D)
        CALL LABJST('T', 'R')                 ! Justify Top/Right
        CALL CP7LB(X, Y, '(-1,-1,-1)')        ! Add caption at (X,Y)
      CALL VSOUT                            ! ViSualization OUTput
      CALL TITLE7('O', 'L', 'Simple captions')
      CALL TITLE7('L', 'C', 'Different Types of Annotation')
C
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 20. Annotation

Explanation of subroutines

`KVXYD(X3D,Y3D,Z3D,X2D,Y2D,DEPTH)` converts a 3-dimensional point to its 2-dimensional equivalent, and gives the depth as the distance of the transformed point from the Viewing Plane.

`CP7LB(X,Y,CAP)` draws a caption, by default with the Bottom Left of the caption at the (x, y) coordinates specified.

`CP7PT(X,Y,MKTYPE,CAP)` draws the symbol `MKTYPE` centred at the point (x, y) and draws the caption to the right of the symbol. Symbols can be also drawn straight into the Raster image using subroutine `VSPMDR`.

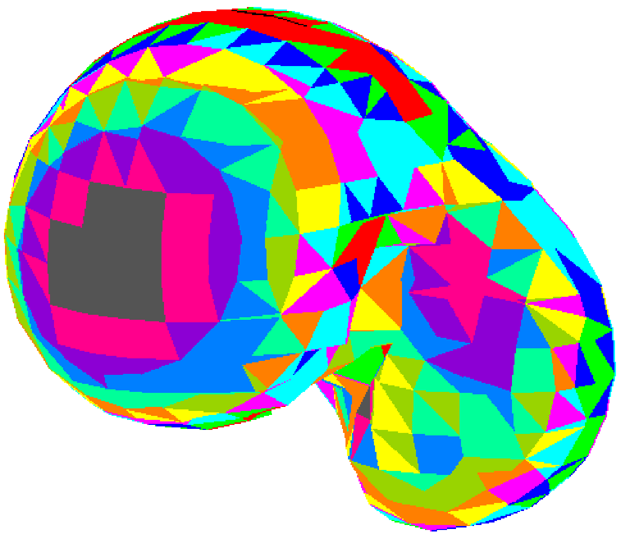
`LABJST(VCHAR,HCHAR)` specifies the justification for subsequent caption labels. In this example, the (x, y) coordinates are specified to be at the Top Right of the caption.

`TITLE7(VCHAR, HCHAR, CAP)` writes title `CAP` at the position defined by characters `VCHAR` and `HCHAR`.

9.21 IsoSurface from a 3-D array

This example illustrates how to:

- draw an IsoSurface of data held in a 3-D array,



```
      PROGRAM VIS21
      INTEGER NX, NY, NZ
      PARAMETER(NX=21, NY=21, NZ=21)
      REAL V(NX,NY,NZ)                      ! 3-d user matrix
      OPEN(10, FILE='vis21.dat', STATUS='OLD')
      READ(10,*) V                          ! Read data
      CLOSE(10)
C
      CALL VSNEW                            ! ViSualization NEW picture
        CALL VSIS(V, NX, NY, NZ, -0.3)      ! Draw IsoSurface of V = -0.3
      CALL VSOUT                            ! ViSualization OUTput
C
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 21. An IsoSurface from a 3-D array

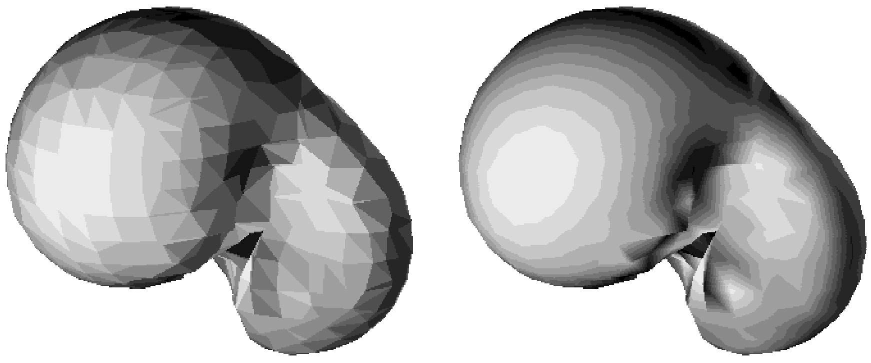
Explanation of subroutines

`VSIS(V,NX,NY,NZ,RVAL)` draws an IsoSurface picture of data level `RVAL` in the `NX` × `NY` × `NZ` array `V` into the raster image. By default the light-sourced intensity values on the surface are mapped on to colour indices 1 to 15.

9.22 Light-sourcing IsoSurfaces

This example illustrates how to:

- set light-sourcing attributes,
- set up a grey scale palette.



```
      PROGRAM VIS22
      INTEGER NX, NY, NZ
      PARAMETER(NX=21, NY=21, NZ=21)
      REAL V(NX,NY,NZ)                      ! 3-d user matrix
      OPEN(10, FILE='vis21.dat', STATUS='OLD')
      READ(10,*) V                          ! Read data
      CLOSE(10)
      CALL GROUP(2, 1)
C
      CALL VSCGS(0.0, 1.0, 2, 15)           ! Set grey scale palette
      CALL VSUTOC(0.0, 1.0, 2, 15)          ! Map intensities to palette
C
      CALL VSLSSM(1)                        ! Default shading method
      CALL VSNEW                            ! ViSualization NEW picture
        CALL VSIS(V, NX, NY, NZ, -0.3)      ! Draw IsoSurface of V = -0.3
      CALL VSOUT                            ! ViSualization OUTput
C
      CALL VSLSSM(2)                        ! Smooth shading method
      CALL VSNEW                            ! ViSualization NEW picture
        CALL VSIS(V, NX, NY, NZ, -0.3)      ! Draw IsoSurface of V = -0.3
      CALL VSOUT                            ! ViSualization OUTput
C
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 22. Light-sourcing IsoSurfaces

Explanation of subroutines

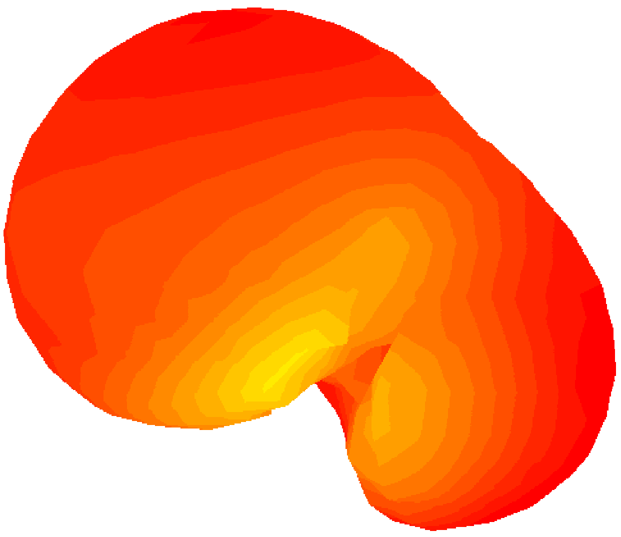
`VSLSSM(METHOD)` specify the shading method for light-sourcing. Shading method 0 (the default) shades each facet of the IsoSurface individually with a single colour. Shading method 1 interpolates colour across each facet.

`VSCGS(RFROM,RTO,ICIX1, ICIX2)` specify a graded grey scale palette suitable for light-sourcing. The intensities for colour indices `ICIX1` and `ICIX2` are given by `RFROM` and `RTO` respectively. Intensities are interpolated for intermediate colour indices.

9.23 Contours of one data set on the IsoSurface of another

This example illustrates how to:

- draw an IsoSurface shaded using the contours of another variable,
- set up a graded colour palette.



```
      PROGRAM VIS23
      INTEGER NX, NY, NZ, CIXMIN, CIXMAX
      PARAMETER(NX=21, NY=21, NZ=21, CIXMIN=2, CIXMAX=15)
      REAL V(NX,NY,NZ)                      ! 3-d user matrix
      REAL U(NX,NY,NZ)                      ! 3-d contour matrix
      OPEN(10, FILE='vis21.dat', STATUS='OLD')
      READ(10,*) V, U                       ! Read data
      CLOSE(10)
C
      CALL VSCRGB(CIXMIN, 1.0, 1.0, 0.0)    ! Palette from yellow
      CALL VSCRGB(CIXMAX, 1.0, 0.0, 0.0)    ! ...to Red
      CALL VSCRNG(CIXMIN, CIXMAX)           ! Fill in rest of palette
      CALL VSUTOC(0.0, 7.0, CIXMIN, CIXMAX) ! Map intensities to palette
C
      CALL VSLSSM(2)                        ! Smooth shading method
      CALL VSNEW                            ! ViSualization NEW picture
        CALL VSISU(V, U, NX, NY, NZ, -0.3)  ! Draw IsoSurface of V = -0.3
      CALL VSOUT                            ! ViSualization OUTput
C
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

Example 23. 2 data sets

Explanation of subroutines

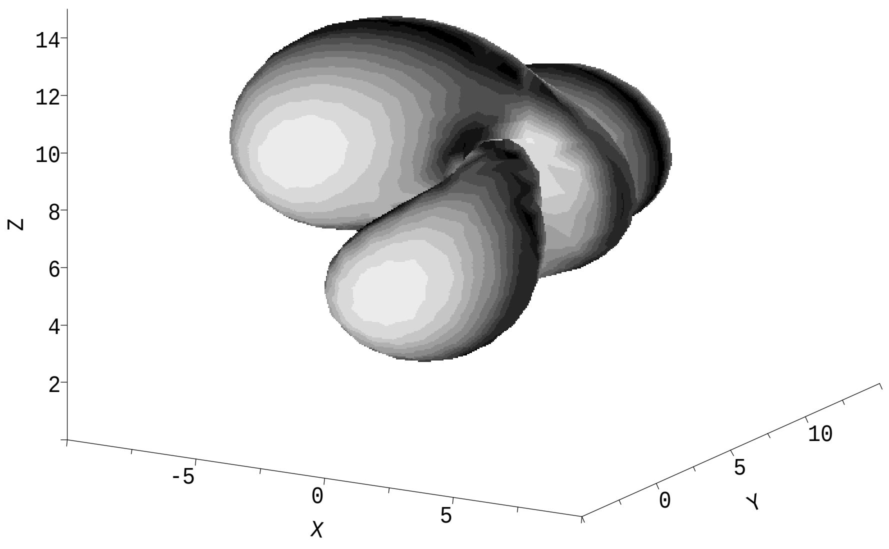
`VSISU(V,U,NX,NY,NZ,RVAL)` draws an IsoSurface picture of data level `RVAL` in the `NX` × `NY` × `NZ` array `V` into the raster image. The IsoSurface is shaded with the contours of the parallel array `U`, which are mapped to colour indices 1 to 15 by default.

`VSCRNG(ICIX1,ICIX2)` specify a graded sequence of colours. A sequence of colour values are interpolated from colour index `ICIX1` to `ICIX2`. The start and end colours of the interpolation are given by the by the colours assigned to `ICIX1` and `ICIX2`.

9.24 Specifying the x-y-z values associated with a regular grid

This example illustrates how to:

- specify x-y-z values associated with the regular grid,
- superimpose multiple data levels.



Explanation of subroutines

`VSEQX(XSTART,XSTEP)` Specifies the base (`XSTART`) and interval (`XSTEP`) for gridded data in the `X` dimension.

`VSEQY(YSTART,YSTEP)` Specifies the base (`YSTART`) and interval (`YSTEP`) for gridded data in the `Y` dimension.

`VSEQZ(ZSTART,ZSTEP)` Specifies the base (`ZSTART`) and interval (`ZSTEP`) for gridded data in the `Z` dimension. This routine only applies to IsoSurface pictures.

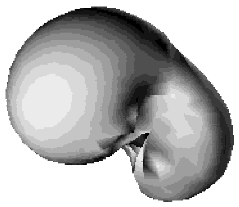
```
      PROGRAM VIS24
      INTEGER NX, NY, NZ
      PARAMETER(NX=21, NY=21, NZ=21)
      REAL XMIN, YMIN, ZMIN, XMAX, YMAX, ZMAX
      PARAMETER(XMIN=-10.0, YMIN=-5.0, ZMIN=0.0)
      PARAMETER(XMAX=10.0, YMAX=15.0, ZMAX=15.0)
      REAL XDIV, YDIV, ZDIV
      PARAMETER(XDIV=1.0, YDIV=1.0, ZDIV=0.75)
      REAL V(NX,NY,NZ)                      ! 3-d user matrix
      OPEN(10, FILE='vis21.dat', STATUS='OLD')
      READ(10,*) V                          ! Read data
      CLOSE(10)
C
      CALL VSCGS(0.0, 1.0, 2, 15)           ! Set grey scale palette
      CALL VSUTOC(0.0, 1.0, 2, 15)          ! Map intensities to palette
C
      CALL VS3DLM(XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX) ! Set true data scales
      CALL VSEQX(XMIN, XDIV)                ! Map data into scales in X
      CALL VSEQY(YMIN, YDIV)                ! and Y
      CALL VSEQZ(ZMIN, ZDIV)                ! and Z
      CALL VSLSSM(2)                        ! Smooth shading method
      CALL VSNEW                            ! ViSualization NEW picture
        CALL VSIS(V, NX, NY, NZ, -0.3)      ! Draw IsoSurface of V = -0.3
        CALL VSIS(V, NX, NY, NZ, 0.3)       ! Also draw IsoSurface of V = 0.3
      CALL VSOUT                            ! ViSualization OUTput
      CALL VSAXDR('X', 'Y', 'Z')
C
      CALL ENDPLT                           ! Close SIMPLEPLOT
      END
```

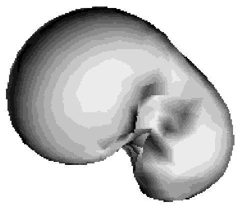
Example 24. Specifying x-y-z values associated with a regular grid

9.25 Drawing an IsoSurface in slices

This example illustrates how to:

- move the IsoSurface light source,
- draw an IsoSurface a slice at a time,
- specify the omission of the 6 end planes of the data set.





```
      PROGRAM VIS25
      INTEGER NX, NY, NZ, I
      PARAMETER(NX=21, NY=21, NZ=21)
      REAL V(NX,NY,NZ)                    ! 3-d user matrix
      OPEN(10, FILE='vis21.dat', STATUS='OLD')
      READ(10,*) V                        ! Read data
      CLOSE(10)
      CALL GROUP(2, 1)
C
      CALL VSCGS(0.0, 1.0, 2, 15)         ! Set grey scale palette
      CALL VSUTOC(0.0, 1.0, 2, 15)        ! Map intensities to palette
C
      CALL VSLSSM(2)                      ! Smooth shading method
      CALL VSNEW                          ! ViSualization NEW picture
        CALL VSIS(V, NX, NY, NZ, -0.3)    ! Draw IsoSurface of V = -0.3
      CALL VSOUT                          ! ViSualization OUTput
C
      CALL VSLSLD(45.0, 5.0)              ! Move light source
      CALL VS3DLM( 0.0, 20.0, 0.0, 20.0, 0.0, 20.0) ! Define scales
      CALL VSISND(1)                      ! Don't draw end planes
      CALL VSNEW                          ! ViSualization NEW picture
        DO 100 I=1,NZ-3                   ! Draw NZ-3 single slices
          CALL VSEQZ(I - 1.0, 1.0)        ! Set position of slice
          CALL VSIS(V(1, 1, I), NX, NY, 4, -0.3) ! Draw IsoSurface slice
 100    CONTINUE
      CALL VSOUT                          ! ViSualization OUTput
C
      CALL ENDPLT                         ! Close SIMPLEPLOT
      END
```

Example 25. Drawing an IsoSurface in slices

Explanation of subroutines

`VSISND(ICODE)` Controls the drawing of the 6 end planes of the data passed to `VSIS` or `VSISU`. By default (`ICODE` = 0) the data in the end planes are drawn. Calling `VSISND` with `ICODE` = 1 causes these end planes to be omitted when drawing. The surface direction is computed by an approximation near the edge of the data array, which is not continuous from one slice to the next. To avoid discontinuities in the picture drawn in slices, edge planes should be omitted. To achieve this at least 4 slices of data must be passed. If the end planes are omitted then only two slices are left to form a single plane of the image.

`VSLSLD(THETA,PHI)` Specifies the direction of the IsoSurface light source as a pair of angles `THETA` and `PHI`.

A. Subroutine specifications

This appendix gives formal specifications for all the *SIMPLEPLOT ViSualization* subroutines. Full specifications of other subroutines are given in the *SIMPLEPLOT Reference manual* (8^{th} edition). All specifications are given in a similar format whether they are classified as graphics, specification or auxiliary subroutines. For example:

SUBROUTINE NAME (ARG1, ARG2)

Name

`NAME` – brief summary line

Availability Section 1, 2, 4, 5, 6, 7 or +, released version 2-*n*.

Arguments Throughout the specifications, arguments of type `INTEGER` have been given names starting with `I–N`, and arguments of type `REAL` have been given names starting with the letters `A–H` and `O–Z`.

IN only arguments are identified as *expression*; a variable name, a constant value or an expression may be used for such arguments.

INOUT or OUT only arguments are identified as *variable*.

`RVAL`	`REAL` expression	Expression with floating point value
`IVAL`	`INTEGER` expression	Expression with integer value
`TORF`	`LOGICAL` expression	Expression with logical value (`.TRUE.` or `.FALSE.`)
`CHAR`	`CHARACTER*1`	Expression with single character value
`CHTYPE`	`CHARACTER*2`	Expression with two character value
`CAP`	`STRING` expression	Expression with any length character value
`VARX`	`REAL` variable	Variable of type `REAL`
`STR`	`STRING` variable	Variable of type `CHARACTER*`*n*
`LABARR`	`STRING` array	Array of string values
`IARR`	`INTEGER` array	1-dimensional array of `INTEGER` values
`I2ARR`	`INTEGER` 2-D array	2-dimensional array of `INTEGER` values
`DARR`	`REAL` array	1-dimensional array of `REAL` values
`D2ARR`	`REAL` 2-D array	2-dimensional array of `REAL` values
`D3ARR`	`REAL` 3-D array	3-dimensional array of `REAL` values

`KVXYD`[7]	convert a 3-D point (x, y, z) to (x, y) and depth
`QV3DLM`[7]	inquire the 3-D limits for plottable data
`QVBRFL`[7]	inquire the z scale required by a 3-D barchart
`QVDPLM`[7]	inquire the range of depths for the current picture
`QVSCAL`[7]	inquire the scale limits in SIMPLEPLOT user coordinates
`VS3DLM`[7]	set the 3-D Limiting Box for plottable data
`VSAXDR`[7]	draw 3-D axes on a *SIMPLEPLOT ViSualization* picture
`VSAXFC`[7]	specify colours for filled rear planes
`VSAXGC`[7]	specify the colour indices for grid lines on the rear planes
`VSBLFL`[7]	fill a 3-D block on a *SIMPLEPLOT ViSualization* picture
`VSBLTN`[7]	tint a 3-D block on a *SIMPLEPLOT ViSualization* picture
`VSBRBZ`[7]	specify the base z value for 3-D barcharts
`VSBRFC`[7]	specify colour sequences on subsequent 3-D barcharts
`VSBRFL`[7]	fill a *SIMPLEPLOT ViSualization* 3-D barchart
`VSBRSZ`[7]	specify the widths of bars
`VSBRTN`[7]	tint a *SIMPLEPLOT ViSualization* 3-D barchart
`VSBRUP`[7]	specify the stacking direction for 3-D barcharts
`VSCGS`[7]	set the palette for a range of colour indices to a grey scale
`VSCHLS`[7]	set hue, lightness and saturation for the palette
`VSCIM`[7]	switch the method of interpolating colour sequences
`VSCRGB`[7]	specify red, green and blue levels for the palette
`VSCRNG`[7]	specify a graded range of target colours in the palette
`VSEDGC`[7]	specify the colour index for the edge of filled areas
`VSEDGI`[7]	specify the offset and increment for edging of gridded data
`VSEDGV`[7]	specify whether the edge of polygons are to be drawn
`VSEQX`[7]	specify the equally-spaced x values for gridded 3-D data
`VSEQY`[7]	specify the equally-spaced y values for gridded 3-D data
`VSEQZ`[7]	specify the equally-spaced z values for gridded 3-D data
`VSFILC`[7]	specify the colour index for area fill subroutines
`VSFIT`[7]	specify which axis scales are measured in similar units
`VSFITP`[7]	specify the ratio between axis scale lengths
`VSFULL`[7]	specify how a projected 3-D object is to fill the picture
`VSINIT`[7]	reset *SIMPLEPLOT ViSualization* defaults
`VSIS`[7]	draw the light-sourced IsoSurface of a specified data level
`VSISND`[7]	specify whether IsoSurfaces should exclude six end planes
`VSISU`[7]	draw contours of one 3-D array on the IsoSurface of another
`VSK7H`[7]	draw a complete horizontal key to a tinted plot
`VSK7V`[7]	draw a complete vertical key to a tinted plot
`VSKEYS`[7]	draw a complete key with distinct shading patterns
`VSLBOX`[7]	specify whether Limiting Boxes are to be drawn
`VSLINC`[7]	specify the colour index to be used by line drawing subroutines
`VSLSLD`[7]	specify the direction of a light source
`VSLSSM`[7]	specify the shading method for light-sourcing
`VSMAG`[7]	magnify coordinates along any or all of the x-y-z axes
`VSNEW`[7]	start a new *SIMPLEPLOT ViSualization* picture
`VSOUT`[7]	flush the raster image
`VSPGFE`[7]	fill an edge-flagged planar polygon
`VSPGFL`[7]	fill a planar polygon with the current fill index
`VSPGTE`[7]	tint an edge-flagged planar polygon
`VSPGTN`[7]	interpolate colour from user data within a planar polygon
`VSPLDR`[7]	draw a polyline through a succession of (x, y, z) data points
`VSPLTN`[7]	interpolate colour from user data on a polyline
`VSPMC`[7]	specify the colour index for subsequent markers
`VSPMDR`[7]	draw a set of markers in a *SIMPLEPLOT ViSualization* image
`VSPMMG`[7]	specify the scaling factor for subsequent markers
`VSPOP`[7]	recall a transformation matrix saved by `VSPUSH`

`VSPUSH`[7]	save current *SIMPLEPLOT ViSualization* transformation matrix
`VSRG`[7]	draw a surface picture from a 2-D array
`VSRGU`[7]	draw contours on a surface picture from 2 functions of 2 variables
`VSROT`[7]	rotate coordinates around one of the x-y-z axes
`VSTRAN`[7]	translate coordinates along any or all of the x-y-z axes
`VSUTOC`[7]	define mapping from user data values to colour indices
`VSVRTP`[7]	set the Viewing Position in terms of a radius and 2 angles
`VSVXYZ`[7]	set the Viewing Position in terms of (x, y, z)
`VSXYZ`[7]	draw a surface picture from three 2-D arrays
`VSXYZU`[7]	draw contours on a surface picture from three 2-D arrays
`VSZ`[7]	draw a surface from data structured into elements
`VSZU`[7]	draw contours on a surface from data structured into elements

SUBROUTINE KVXYD (X3D, Y3D, Z3D, VARX2D, VARY2D, VARDEP)

Name

KVXYD – to convert a three-dimensional point (x, y, z) to its two-dimensional equivalent plus depth.

Availability Section 7, released version 2-13.

Arguments

X3D, Y3D, Z3D	REAL expressions	Coordinates of a point, specified in 3-D units before transformation
VARX2D, VARY2D	REAL variables	To receive the coordinates of the projected point
VARDEP	REAL variable	To receive the distance of the projected point from the Viewing Plane

Description

KVXYD converts a three-dimensional point (X3D,Y3D,Z3D) to (VARX2D,VARY2D), the equivalent 2-D coordinates once transformation and projection have been applied. VARX2D and VARY2D are measured in terms of the underlying SIMPLEPLOT scales.

KVXYD also calculates the value of the depth, VARDEP. By default, if a point is within the limits specified by VS3DLM, its depth lies in the range $-\sqrt{3} \leq$ VARDEP $\leq \sqrt{3}$. Points between the Origin and the Viewing Position have negative depths – the greater the depth, the further away a point is from the Viewing Position.

The permissible range of depths is altered following a call to VSFULL.

See also

QVDPLM, QVSCAL, VS3DLM, VSFULL, VSVRTP and VSVXYZ.

SUBROUTINE QV3DLM (VARX1, VARX2, VARY1, VARY2, VARZ1, VARZ2)

Name

`QV3DLM` – to inquire the three-dimensional limits for plottable data.

Availability Section 7, released version 2-13.

Arguments

`VARX1`	`REAL` variable	To receive value at start of x scale
`VARX2`	`REAL` variable	To receive value at end of x scale
`VARY1`	`REAL` variable	To receive value at start of y scale
`VARY2`	`REAL` variable	To receive value at end of y scale
`VARZ1`	`REAL` variable	To receive value at start of z scale
`VARZ2`	`REAL` variable	To receive value at end of z scale

Description

SIMPLEPLOT ViSualization scales are set to accommodate all data within a three-dimensional box. By default, the x, y and z limits of this box are all -1.0 to $+1.0$, although `VS3DLM` can change these limits.

`QV3DLM` inquires the data limits used for the current picture.

All points which lie inside the limits after transformations have been applied can be plotted. Other points may lie outside the 2-D SIMPLEPLOT scales.

`QV3DLM` may also be used to calculate the Origin, which is at the centre of the Limiting Box. This is at

$$\left(\frac{\texttt{VARX1+VARX2}}{2}, \frac{\texttt{VARY1+VARY2}}{2}, \frac{\texttt{VARZ1+VARZ2}}{2}\right)$$

`QV3DLM` gives the limitis for the current picture. It must be called after the scales have been set by `VSNEW`. If `VSNEW` has not been called, all the variables are set to the current 'no data' value.

Diagnostics

`[QV3DLM: No current ViSualization picture]`[1]

See also

`VS3DLM`.

SUBROUTINE QVBRFL (Z, NX, NY, NSETS, ZMIN, ZMAX)

Name

`QVBRFL` – to inquire the z scale required by a *SIMPLEPLOT ViSualization* 3-D barchart.

Availability Section 7, released version 2-14.

Arguments

`Z`	`REAL` 3-D array	`NX` × `NY` × `NSETS` array of user (z) values
`NX`, `NY`, `NSETS`	`INTEGER` expressions	Dimensions of 3-D data array
`ZMIN`	`REAL` variable	To receive minimum z
`ZMAX`	`REAL` variable	To receive maximum z

Description

`QVBRFL` returns the minimum and maximum values required on the z scale for `VSBRFL` to plot the data in array `Z` with the current base z value. This may be used to establish the z scale to set 3-D scales before drawing a stacked barchart.

Diagnostics

`[QVBRFL: Invalid array dimension]`[1]

See also

`VS3DLM`, `VSBRBZ` and `VSBRFL`.

SUBROUTINE QVDPLM (VARMIN, VARMAX)

Name

`QVDPLM` – to inquire the range of depths for the current *SIMPLEPLOT ViSualization* picture.

Availability Section 7, released version 2-13.

Arguments

`VARMIN`	`REAL` variable	To receive minimum depth
`VARMAX`	`REAL` variable	To receive maximum depth

Description

`QVDPLM` gives the range of depths of points inside the *SIMPLEPLOT ViSualization* Limiting Box. The limits for the box are set by `VS3DLM`, given the current scaling mode set by `VSFULL`.

Depth increases away from the Viewing Position – only points which are the other side of the Viewing Plane have positive depths.

The range of depths is affected by `VSFULL` as follows:

- `VSFULL(0)` (default), `VARMIN` $\geq -\sqrt{3}$ and `VARMAX` $\leq +\sqrt{3}$.
- `VSFULL(1)`, `VARMIN` $= -\sqrt{3}$ and `VARMAX` $= +\sqrt{3}$.
- `VSFULL(2)`, the range of depths depends on the transformations specified by `VSMAG`, `VSROT`, and `VSTRAN`.

`QVDPLM` gives the limits for the current picture. It must be called after the scales have been set by `VSNEW`. If `VSNEW` has not been called, all the variables are set to the current 'no data' value.

Diagnostics

`[QVDPLM: No current ViSualization picture]`[1]

See also

`VS3DLM`, `VSFULL`, `VSVRTP` and `VSVXYZ`.

SUBROUTINE QVSCAL (VARXMN, VARXMX, VARYMN, VARYMX)

Name

`QVSCAL` – to inquire the scale limits in SIMPLEPLOT user coordinates corresponding to the current projected plotting area.

Availability Section 7, released version 2-13.

Arguments

`VARXMN`	`REAL` variable	To receive minimum value of x scale
`VARXMX`	`REAL` variable	To receive maximum value of x scale
`VARYMN`	`REAL` variable	To receive minimum value of y scale
`VARYMX`	`REAL` variable	To receive maximum value of y scale

Description

`QVSCAL` returns the maximum possible plotting area for the current *SIMPLEPLOT ViSualization* picture. The plotting limits may be altered by calls to `VS3DLM`, `VSFIT`, `VSFITP`, `VSFULL`, `VSVRTP` or `VSVXYZ`.

If a point is within the limits specified by `VS3DLM`, the range of possible x-y values is determined by the Scaling Mode selected by `VSFULL`:

- `VSFULL(0)` (default),
  `VARXMN/VARYMN` $\geq -\sqrt{3}$ and `VARXMX/VARYMX` $\leq +\sqrt{3}$.
- `VSFULL(1)`, `VARXMN/VARYMN` $= -\sqrt{3}$, and `VARXMX/VARYMX` $= +\sqrt{3}$.
- `VSFULL(2)`, the 2-D limits depends on the transformations specified by `VSMAG`, `VSROT` and `VSTRAN`.

`QVSCAL` gives the limits for the current picture. It must be called after the scales have been set by `VSNEW`. If `VSNEW` has not been called, all the variables are set to the current 'no data' value.

Diagnostics

`[QVSCAL: No current ViSualization picture]`^{1}

See also

`VS3DLM`, `VSFIT`, `VSFITP`, `VSFULL`, `VSVRTP` and `VSVXYZ`.

SUBROUTINE VS3DLM (XSTART, XSTOP, YSTART, YSTOP, ZSTART, ZSTOP)

Name

VS3DLM – to set the three-dimensional Limiting Box for plottable data.

Availability Section 7, released version 2-13.

Arguments

XSTART	REAL expression	Value at start of x scale
XSTOP	REAL expression	Value at end of x scale
YSTART	REAL expression	Value at start of y scale
YSTOP	REAL expression	Value at end of y scale
ZSTART	REAL expression	Value at start of z scale
ZSTOP	REAL expression	Value at end of z scale

Description

SIMPLEPLOT ViSualization scales are set to accommodate all data within a three-dimensional box. By default, the x, y and z limits of this box are all -1.0 to $+1.0$. VS3DLM changes these limits.

All points which lie inside the Limiting Box after transformation can be plotted. Note that, although a point may have (x, y, z) coordinates within the limits set by VS3DLM, it may still lie outside the Limiting Box if any transformations have been applied.

For more information about the effect of transformations on scales, see documentation on the transformation subroutines VSMAG, VSROT and VSTRAN, and the scaling subroutines VSFIT, VSFITP and VSFULL.

VS3DLM is also used to set the Origin, which is at the centre of the Limiting Box. By default, this is at $(0.0, 0.0, 0.0)$, although after

```
CALL VS3DLM(XSTART, XSTOP, YSTART, YSTOP, ZSTART, ZSTOP)
```

it is at

$$\left(\frac{\mathtt{XSTART+XSTOP}}{2}, \frac{\mathtt{YSTART+YSTOP}}{2}, \frac{\mathtt{ZSTART+ZSTOP}}{2}\right)$$

VS3DLM does not take effect until the next call of VSNEW.

Default

Calling VS3DLM with XSTART=XSTOP, YSTART=YSTOP or ZSTART=ZSTOP restores the default for all scales.

See also

QV3DLM, VSFIT, VSFITP, VSFULL and VSINIT.

SUBROUTINE VSAXDR (CAPX, CAPY, CAPZ)

Name

VSAXDR – to draw 3-D axes on a *SIMPLEPLOT ViSualization* picture.

Availability Section 7, released version 2-14.

Arguments

CAPX	STRING expression	x-axis caption
CAPY	STRING expression	y-axis caption
CAPZ	STRING expression	z-axis caption

Description

VSAXDR can be called to draw x-y-z axes for the current *SIMPLEPLOT ViSualization* picture. VSAXDR must be called after 3-D scales have been applied by VSNEW. The axes are drawn directly on the picture, not into the raster image. It is preferable to draw axes after the image has been transmitted by VSOUT, to avoid the axis being overdrawn.

Diagnostics

`[VSAXDR: No current ViSualization picture]`[1]

See also

AX*, VSAXFC, VSNEW and VSOUT.

SUBROUTINE VSAXFC (IXFIX, IYFIX, IZFIX)

Name

`VSAXFC` – to specify the colours for filled rear planes of *SIMPLEPLOT ViSualization* pictures.

Availability Section 7, released version 2-14.

Arguments

`IXFIX`	`INTEGER` expression	Fill index for rear x plane
`IYFIX`	`INTEGER` expression	Fill index for rear y plane
`IZFIX`	`INTEGER` expression	Fill index for rear z plane

Description

`VSAXFC` specifies colour indices for filling the rear planes of the *SIMPLEPLOT ViSualization* Limiting Box. If a non-negative value is specified for any plane, it is filled whenever a new image is started. `VSAXFC` must therefore be called before `VSNEW`.

`IXFIX` specifies the colour index for the rear plane(s) with x constant, `IYFIX` specifies the colour index for the rear plane(s) with y constant, and `IZFIX` specifies the colour index for the rear plane(s) with z constant.

If colour index -1 is specified (default), the associated plane is not filled. Zero, one or two planes may be filled for each variable, depending on the Viewing Position set by `VSVRTP`/`VSVXYZ`.

In this release of the software, data may be mapped onto colour indices 0 to 95 inclusive.

Default

`CALL VSAXFC(-1,-1,-1)` restores the default.

Diagnostics

`[VSAXFC: Colour index` *`I`* `out of range]`^{1}

See also

`VSAXGC`, `VSOUT`, `VSVRTP` and `VSVXYZ`.

SUBROUTINE VSAXGC (IXCIX, IYCIX, IZCIX)

Name

`VSAXGC` – to specify the colour indices for grid lines on the rear planes of *SIMPLEPLOT ViSualization* pictures.

Availability Section 7, released version 2-14.

Arguments

`IXCIX`	`INTEGER` expression	Colour index for x
`IYCIX`	`INTEGER` expression	Colour index for y
`IZCIX`	`INTEGER` expression	Colour index for z

Description

`VSAXGC` specifies colour indices for grid lines on the rear planes of the *SIMPLEPLOT ViSualization* Limiting Box. When grids have been requested, they are drawn in the raster image whenever a new image is started. `VSAXGC` must therefore be called before `VSNEW`.

`IXCIX` specifies the colour index for grid lines on the rear plane(s) with x constant, `IYCIX` specifies the colour index for grid lines on the rear plane(s) with y constant, and `IZCIX` specifies the colour index for grid lines on the rear plane(s) with z constant.

If index -1 is specified, grids on the associated planes are not drawn, and if rear planes are filled, grid lines are formed by gaps in the filling. If index -2 is specified, the associated grids are omitted. This is the default.

Default

`CALL VSAXGC(-2,-2,-2)` restores the default.

Diagnostics

`[VSAXGC: Colour index` *`I`* `out of range]`[1]

See also

`VSAXFC`.

SUBROUTINE VSBLFL (X1, X2, Y1, Y2, Z1, Z2)

Name

VSBLFL – to fill a 3-D block on a *SIMPLEPLOT ViSualization* picture.

Availability Section 7, released version 2-14.

Arguments

X1, X2	REAL expressions	Range of x values
Y1, Y2	REAL expressions	Range of y values
Z1, Z2	REAL expressions	Range of z values

Description

VSBLFL draws a single block on the current *SIMPLEPLOT ViSualization* picture. This block is filled with the current fill index set by VSFILC, and subject to edge visibility and colour.

If the range of any dimension is zero (X1=X2, Y1=Y2 or Z1=Z2) nothing is drawn.

Diagnostics

`[VSBLFL: No current ViSualization picture]`^{1}

See also

VSBLTN, VSBRFL, VSEDGC, VSEDGV and VSFILC.

SUBROUTINE VSBLTN (X1, X2, Y1, Y2, Z1, Z2, U1, U2)

Name

`VSBLTN` – to tint a 3-D block on a *SIMPLEPLOT ViSualization* picture.

Availability Section 7, released version 2-14.

Arguments

`X1`, `X2`	`REAL` expressions	Range of x values
`Y1`, `Y2`	`REAL` expressions	Range of y values
`Z1`, `Z2`	`REAL` expressions	Range of z values
`U1`, `U2`	`REAL` expressions	Range of u values

Description

`VSBLTN` draws a single block on the current *SIMPLEPLOT ViSualization* picture. The block is tinted with colours ranging from the colour derived from `U1` at $z =$ `Z1` to the colour derived from `U2` at $z =$ `Z2`. The block is subject to edge visibility and colour.

Diagnostics

`[VSBLTN: No current ViSualization picture]`[1]

See also

`VSBLFL`, `VSBRTN`, `VSEDGC` and `VSEDGV`.

SUBROUTINE VSBRBZ (ZVAL)

Name

`VSBRBZ` – to specify the base z value for bars in *SIMPLEPLOT ViSualization* 3-D barcharts.

Availability Section 7, released version 2-14.

Argument

`ZVAL`	`REAL` expression	Required base z value

Description

By default, *SIMPLEPLOT ViSualization* 3-D barcharts show each bar extending from $z = 0.0$, to a z value from an array. `VSBRBZ` can be called before `VSBRFL` or `VSBRTN` to set an alternative base z value.

Default

`CALL VSBRBZ(0.0)` restores the default.

See also

`VSBRFL` and `VSBRTN`.

SUBROUTINE VSBRFC (ICODE)

Name

`VSBRFC` – to specify how colour sequences should be used on subsequent *SIMPLEPLOT ViSualization* 3-D barcharts.

Availability Section 7, released version 2-14.

Arguments

`ICODE`	`INTEGER` expression	Specify use of colour (see below)

`ICODE`	*Different colours*
0	Faces of bars distinguished by pen pointers (default)
1	Data Rows distinguished by colour sequence
2	Data Columns distinguished by colour sequence
3	Data Sets distinguished by colour sequence
4	Data magnitudes distinguished by colour sequence

Description

`VSBRFL` fills a *SIMPLEPLOT ViSualization* 3-D barchart. `VSBRFC` can be called before `VSBRFL`, to specify how colours are used for the bars.

When `ICODE` $= 0$, three colour indices distinguish the three planes of each bar: the x plane uses the first colour index, the y plane uses the second, and the z plane uses the third.

When `ICODE` > 0, all planes of a bar are shaded with the same colour index. `SQSHAD` may be called to specify the sequence of colour indices. When `SQSHAD` has not been called, the colour indices are used in the natural sequence 1, 2, 3, ...

Diagnostics

`[VSBRFC: Integer` *I* `out of range]`[1]

Default

`CALL VSBRFC(0)` restores the default.

See also

`SQSHAD` and `VSBRFL`.

SUBROUTINE VSBRFL (Z, NX, NY, NSETS)

Name

`VSBRFL` – to fill a *SIMPLEPLOT ViSualization* 3-D barchart.

Availability Section 7, released version 2-14.

Arguments

`Z`	`REAL` 3-D array	`NX` × `NY` × `NSETS` array of user (z) values
`NX`, `NY`, `NSETS`	`INTEGER` expressions	Dimensions of 3-D data array

Description

A *SIMPLEPLOT ViSualization* 3-D barchart is drawn representing `NSETS` sets of values in array `Z`, stacked on top of each other. The sets may be stacked downward with the first set at the top, or upward with the first set at the bottom, as specified by `VSBRUP`.

Different colours may be used to distinguish between data columns, data rows, or data sets, or data magnitudes, as specified by `VSBRFC`.

Diagnostics

`[VSBRFL: No current ViSualization picture]`[1]
`[VSBRFL: Invalid array dimension]`[1]

See also

`QVBRFL`, `VSBRBZ`, `VSBRFC`, `VSBRSZ`, `VSBRTN` and `VSBRUP`.

SUBROUTINE VSBRSZ (X1, X2, Y1, Y2)

Name

`VSBRSZ` – to specify the widths of bars in *SIMPLEPLOT ViSualization* 3-D barcharts.

Availability Section 7, released version 2-14.

Arguments

`X1`, `X2`	`REAL` expressions	Limits of bars in x direction
`Y1`, `Y2`	`REAL` expressions	Limits of bars in y direction

Description

By default, each bar in *SIMPLEPLOT ViSualization* 3-D barcharts extends from 0.1 to 0.9 of the space available in both x and y. `VSBRSZ` can be called to specify alternative limits.

When `X2` is not greater than `X1`, or either lies outside the range 0.0 to 1.0, the default in x is restored; when `Y2` is not greater than `Y1`, or either lies outside the range 0.0 to 1.0, the default in y is restored.

Default

`CALL VSBRSZ(0.0, 0.0, 0.0, 0.0)` restores the default.

See also

`VSBRFL` and `VSBRTN`.

SUBROUTINE VSBRTN (Z, NX, NY)

Name

`VSBRTN` – to tint a *SIMPLEPLOT ViSualization* 3-D barchart.

Availability Section 7, released version 2-14.

Arguments

`Z`	`REAL` 2-D array	`NX`$\times$`NY` user (z) values
`NX`, `NY`	`INTEGER` expressions	Dimensions of 2-D data array

Description

`VSBRTN` draws a *SIMPLEPLOT ViSualization* 3-D barchart representing the values in array `Z`. The colours representing different z values are controlled by `VSUTOC`, and by default map onto colour indices 1 to 15.

Diagnostics

`[VSBRTN: No current ViSualization picture]` ^{1}
`[VSBRTN: Invalid array dimension]` ^{1}

See also

`VSBRBZ`, `VSBRFL`, `VSBRSZ` and `VSUTOC`.

SUBROUTINE VSBRUP (IUP)

Name

`VSBRUP` – to specify the stacking direction for *SIMPLEPLOT ViSualization* 3-D barcharts.

Availability Section 7, released version 2-14.

Arguments

`IUP`	`INTEGER` expression	Whether bars stack upwards

`IUP`	*Direction*
0	Downward (default)
1	Upward

Description

`VSBRUP` is called before `VSBRFL` to specify whether multiple data sets should be stacked downward with the first set at the top, or upward with the first set at the bottom:

Diagnostics

`[VSBRUP: Integer` *I* `out of range]`[1]

Default

`CALL VSBRUP(0)` restores the default.

See also

`VSBRFL` and `VSBRTN`.

SUBROUTINE VSCGS (RFROM, RTO, ICIX1, ICIX2)

Name

VSCGS – to set the palette for a range of colour indices to a grey scale suitable for light-sourcing.

Availability Section 7, released version 2-15.

Arguments

RFROM, RTO	REAL expressions	Range of grey levels [0.0-1.0]
ICIX1, ICIX2	INTEGER expressions	Range of colour indices

Description

VSCGS changes the *SIMPLEPLOT ViSualization* palette, setting a grey scale from grey level RFROM at index ICIX1 to grey level RTO at index ICIX2, where 0.0 represents Black and 1.0 represents White.

Diagnostics

[VSCGS: Colour index *I* out of range]^1
[VSCGS: Colour specification *R* out of range]^1

See also

VSCIM and VSIS.

SUBROUTINE VSCHLS (ICIX, HUE, BRIGHT, SATN)

Name

`VSCHLS` – to set hue, lightness and saturation levels for the *SIMPLEPLOT ViSualization* palette.

Availability Section 7, released version 2-13.

Arguments

`ICIX`	`INTEGER` expression	Colour index to be changed
`HUE`	`REAL` expression	Hue (degrees)
`BRIGHT`	`REAL` expression	Lightness (0.0–1.0)
`SATN`	`REAL` expression	Saturation (0.0–1.0)

Description

SIMPLEPLOT ViSualization maintains its own target palette for output. Palette requests on devices with limited colour capabilities are rendered using Error Diffusion techniques.

`BRIGHT` and `SATN` must be between 0.0 and 1.0. If they are not, a diagnostic is issues and the palette is unchanged.

`VSCHLS` sets the device palette if palette control is available.

Diagnostics

`[VSCHLS: Colour index` *`I`* `out of range]`^{1}
`[VSCHLS: Colour specification` *`R`* `out of range]`^{1}

See also

`VSCRGB` and `VSCRNG`.

SUBROUTINE VSCIM (METHOD)

Name

`VSCIM` – to switch the method of interpolating colour sequences.

Availability Section 7, released version 2-15.

Arguments

`METHOD`	`INTEGER` expression	Method ID (see below)

`METHOD`	*Interpolation method*
0	Linear interpolation
1	Non-linear gamma corrected interpolation

Description

`VSCIM` controls which method of interpolation is used by `VSCGS` and `VSCRNG`. Linear interpolation gives colour values with equally spaced intensities as passed to the display hardware. Non-linear gamma corrected interpolation provides a range of colours with equally spaced intensities as perceived by the human eye. By default linear interpolation is used.

Diagnostics

`[VSCIM: Integer` *I* `out of range]`[1]

Default

`CALL VSCIM(0)` restores the default.

See also

`VSCGS`, and `VSCRNG`.

SUBROUTINE VSCRGB (ICIX, RED, GREEN, BLUE)

Name

`VSCRGB` – to specify red, green and blue levels for the *SIMPLEPLOT ViSualization* palette.

Availability Section 7, released version 2-13.

Arguments

`ICIX`	`INTEGER` expression	Colour index to be specified
`RED`	`REAL` expression	Proportion of Red (0.0–1.0)
`GREEN`	`REAL` expression	Proportion of Green (0.0–1.0)
`BLUE`	`REAL` expression	Proportion of Blue (0.0–1.0)

Description

SIMPLEPLOT ViSualization maintains its own target palette for output. Palette requests on devices with limited colour capabilities are rendered using Error Diffusion techniques.

`RED`, `GREEN` and `BLUE` must be between 0.0 and 1.0. If they are not, a diagnostic is issued and the palette is unchanged.

`VSCRGB` sets the device palette if palette control is available.

Diagnostics

`[VSCRGB: Colour index` *I* `out of range]`[1]
`[VSCRGB: Colour specification` *R* `out of range]`[1]

See also

`VSCHLS` and `VSCRNG`.

SUBROUTINE VSCRNG (ICIX1, ICIX2)

Name

`VSCRNG` – to set a graded range of colours in the *SIMPLEPLOT ViSualization* palette.

Availability Section 7, released version 2-14.

Arguments

`ICIX1`	`INTEGER` expression	first pen index of graded range
`ICIX2`	`INTEGER` expression	last pen index of graded range

Description

`VSCRNG` sets the colours for all indices between `ICIX1` and `ICIX2`. The existing values of colours `ICIX1` and `ICIX2` are left unchanged and a graded range of colours is interpolated for all indices between them.

See also

`VSCHLS` and `VSCRGB`.

SUBROUTINE VSEDGC (ICIX)

Name

VSEDGC – to specify the colour index for the edge of filled areas.

Availability Section 7, released version 2-13.

Argument

ICIX	INTEGER expression	Colour index for edge of filled areas

Description

VSEDGC sets the colour index for the edge of *SIMPLEPLOT ViSualization* filled areas. This value is then applied if the edge visibility is set to ON by VSEDGV(1). In this release of software, data may be mapped on to colour indices 0 to 95 inclusive.

If colour index −1 is specified, subsequent edges are omitted. This differs from selecting colour index 0, which draws the edge in background, and from VSEDGV(0) which fills the interior up to and including the edge.

Diagnostics

[VSEDGC: Colour index *I* out of range][1]

Default

CALL VSEDGC(1) restores the default.

See also

VSBLFL, VSBLTN, VSBRFL, VSBRTN, VSEDGV, VSPGFL, VSRG, VSRGU, VSPGTN, VSXYZ, VSXYZU, VSZ and VSZU.

SUBROUTINE VSEDGI (IDIM, IOFF, INCR)

Name

`VSEDGI` – to specify an offset and increment for edging of gridded data elements to form a grid over surfaces.

Availability Section 7, released version 2-15.

Arguments

`IDIM`	`INTEGER` expression	Dimension number (see below)
`IOFF`	`INTEGER` expression	Offset for first edge
`INCR`	`INTEGER` expression	Increment between edges

`IDIM`	*dimension(s) affected*
0	All
1	x
2	y
3	z (Not relevant for 2–D arrays)

Description

`VSEDGI` controls the offset and increment of drawn edges for one of the dimensions of gridded data. `IDIM` indicates which dimension is being specified, 1, 2, or 3, or `IDIM`=0 may be used to specify all dimensions. `VSEDGI` calls with `IDIM=1` and `IDIM`=2 affect all ViSualization plots drawn from gridded data; `IDIM`=3 applies to plotting from 3-dimensional arrays, but not to plotting from 2-dimensional arrays. By default, when edge visibility is ON (see `VSEDGV`) all edges of all elements are drawn. By selecting a less frequent density of edges, a coarser grid can be shown on the surface. The grid lines are offset by `IOFF` from the first data point. All edges in one dimension can be suppressed by calling `VSEDGI` with `INCR`=0.

Diagnostics

`[VSEDGI: Integer` *I* `out of range]`[1]

Default

`CALL VSEDGI(0, 0, 1)` restores the defaults.

See also

`VSEDGC`, `VSEDGV`, `VSIS`, `VSISU`, `VSRG`, `VSRGU`, `VSXYZ`, and `VSXYZU`.

SUBROUTINE VSEDGV (LEVEL)

Name

`VSEDGV` – to specify whether the edge of polygons is to be drawn.

Availability Section 7, released version 2-13.

Argument

`LEVEL`	`INTEGER` expression	Whether to draw the edge in edge colour index

`LEVEL`	*Drawn*
0	OFF – no edge drawn
1	ON – edge drawn

Description

By default, the edge visibility is OFF, and the edge of filled areas is drawn in the same colour as the interior. By setting the edge visibility ON, the edge of each polygon is drawn in the edge colour index (see `VSEDGC`). By drawing the edge of polygons separately from the interior, the underlying data structure can be displayed.

Diagnostics

`[VSEDGV: Integer` *I* `out of range]`[1]

Default

`CALL VSEDGV(0)` restores the default.

See also

`VSBLFL`, `VSBLTN`, `VSBRFL`, `VSBRTN`, `VSEDGC`, `VSEDGI`, `VSIS`, `VSISU`, `VSPGFL`, `VSPGTN`, `VSRG`, `VSRGU`, `VSXYZ`, `VSXYZU`, `VSZ` and `VSZU`.

SUBROUTINE VSEQX (XSTART, XSTEP)

Name

`VSEQX` − to specify the equally-spaced x values to be associated with *SIMPLEPLOT ViSualization* gridded 3-D data.

Availability Section 7, released version 2-13.

Arguments

`XSTART`	`REAL` expression	x value of first data column
`XSTEP`	`REAL` expression	Interval between data columns

Description

By default, data on a regular grid drawn by `VSRG`, `VSRGU`, `VSIS` or `VSISU` are scaled to fit within the current Limiting Box specified by `VS3DLM`.

`VSEQX` specifies the x values corresponding to the data grid, and may be used in conjunction with `VS3DLM` to position the $y-z$ plane in 3-D space. `XSTART` specifies the x coordinate to be associated with the first column of the data array, and `XSTEP` specifies the interval between columns.

`VSEQX` has no effect if `VS3DLM` has not been called to set the *SIMPLEPLOT ViSualization* 3-D scales explicitly.

Default

Calling `VSEQX` with `XSTEP`=0.0 restores the default.

See also

`VSEQY`, `VSEQZ`, `VSRG`, `VSRGU`, `VSIS`, and `VSISU`.

SUBROUTINE VSEQY (YSTART, YSTEP)

Name

`VSEQY` – to specify the equally-spaced y values to be associated with *SIMPLEPLOT ViSualization* gridded 3-D data.

Availability Section 7, released version 2-13.

Arguments

`YSTART`	`REAL` expression	y value of first data row
`YSTEP`	`REAL` expression	Interval between data rows

Description

By default, data on a regular grid drawn by `VSRG`, `VSRGU`, `VSIS` or `VSISU` are scaled to fit within the current Limiting Box specified by `VS3DLM`.

`VSEQY` specifies the y values corresponding to the data grid, and may be used in conjunction with `VS3DLM` to position the $x - z$ plane in 3-D space. `YSTART` specifies the y coordinate to be associated with the first row of the data array, and `YSTEP` specifies the interval between rows.

`VSEQY` has no effect if `VS3DLM` has not been called to set the *SIMPLEPLOT ViSualization* 3-D scales explicitly.

Default

Calling `VSEQY` with `YSTEP`=0.0 restores the default.

See also

`VSEQX`, `VSEQZ`, `VSRG`, `VSRGU`, `VSIS` and `VSISU`.

SUBROUTINE VSEQZ (ZSTART, ZSTEP)

Name

`VSEQZ` – to specify the equally-spaced z values to be associated with *SIMPLEPLOT ViSualization* gridded 3-D data.

Availability Section 7, released version 2-15.

Arguments

`ZSTART`	`REAL` expression	z value of first data plane
`ZSTEP`	`REAL` expression	Interval between data planes

Description

By default, data on a regular grid drawn by `VSIS` or `VSISU` are scaled to fit within the current Limiting Box specified by `VS3DLM`.

`VSEQZ` specifies the z values corresponding to the data grid, and may be used in conjunction with `VS3DLM` to position the data in the 3-D space. `ZSTART` specifies the first z value of the grid, and `ZSTEP` specifies the z interval of the grid.

`VSEQZ` has no effect unless `VS3DLM` has been called to set the *SIMPLEPLOT ViSualization* 3-D scales explicitly.

Default

Calling `VSEQZ` with `ZSTEP`=0.0 restores the default.

See also

`VSEQX`, `VSEQY`, `VSIS` and `VSISU`.

SUBROUTINE VSFILC (ICIX)

Name

`VSFILC` – to specify the colour index to be used by *SIMPLEPLOT ViSualization* area fill subroutines.

Availability Section 7, released version 2-13.

Argument

`ICIX`	`INTEGER` expression	Colour index for filling

Description

`VSFILC` sets the colour index for subsequent filling by `VSPGFL` or `VSBLFL`. In this release of software, data may be mapped on to colour indices 0 to 95 inclusive. If colour index -1 is specified, subsequent areas are not filled.

Diagnostics

`[VSFILC: Colour index` *I* `out of range]`[1]

Default

`CALL VSFILC(1)` restores the default.

See also

`VSBLFL`, `VSCHLS`, `VSCRGB` and `VSPGFL`.

SUBROUTINE VSFIT (CHAXES)

Name

`VSFIT` – to specify which axis scales are measured in similar units.

Availability Section 7, released version 2-13.

Argument

`CHAXES`	`STRING` expression	axes which are related

`CHAXES`	*similar axes*
`'XYZ'`	all scales in same units (default)
`'XY'`	x and y scales are similar
`'XZ'`	x and z scales are similar
`'YZ'`	y and z scales are similar
`' '`	all scales are unrelated

Description

2-D Simpleplot scales are set to accommodate all data within the three-dimensional box specified by `VS3DLM`. By default, each of these scales is measured in the same units. If the ranges of these scales are of different magnitudes, the resulting picture is distorted.

`VSFIT` specifies which scales are comparable. If scales are unrelated, each axis defines the edge of a cube which contains the data. If only 2 scales are similar, the length of the third axis is equal to the longer of the other two.

`CHAXES` may be in upper or lower case, but if it does not correspond to one of the strings above, an error message is written and nothing is altered.

If non-default scales are used, the relationship between axes is affected by the transformation routines, `VSMAG`, `VSROT` and `VSTRAN`.

Diagnostics

`[VSFIT: Invalid string` *`XX`*`]`[1]

Default

`CALL VSFIT('XYZ')` restores the default.

See also

`VS3DLM` and `VSFITP`.

SUBROUTINE VSFITP (XLEN, YLEN, ZLEN)

Name

`VSFITP` – to specify the relative lengths of the x, y and z axes.

Availability Section 7, released version 2-14.

Argument

`XLEN`	`REAL` expression	Length of x-axis
`YLEN`	`REAL` expression	Length of y-axis
`ZLEN`	`REAL` expression	Length of z-axis

Description

By default, the lengths of the *SIMPLEPLOT ViSualization* scales are proportional to the data range they cover. This means that if the ranges are not comparable, the picture is unduly distorted.

`VSFITP` specifies that the axis lengths are in proportion to each other. After `CALL VSFITP(1.0,1.0,1.0)`, for example, all axes are of the same length. After `CALL VSFITP(1.0,2.0,4.0)`, the x-axis is half the length of the y-axis, which is half the length of the z-axis.

If any argument is non-zero, the default is restored – scales are proportional to data range.

Default

`CALL VSFITP(0.0,0.0,0.0)` restores the default.

See also

`VS3DLM` and `VSFIT`.

SUBROUTINE VSFULL (MODE)

Name

VSFULL – to specify how the Limiting Box is to fill the SIMPLEPLOT picture.

Availability Section 7, released version 2-13.

Argument

MODE	INTEGER expression	Scaling Mode used

Description

By default, the 2-D SIMPLEPLOT scales are set to accommodate the extent of the Limiting Box. The underlying SIMPLEPLOT scales are Linear Cartesian with 1 unit in x the same as 1 unit in y.

If a point or object is *Transformed* (Rotated, Translated or Magnified), its (x, y, z) coordinates change. This means that it may no longer fit on the picture, even though before transformation it lies within the limits specified by VS3DLM. After CALL VSFULL(1), VSNEW sets scales to allow all possible rotations. The relative size of the image is comparable for successive pictures.

After CALL VSFULL(2), VSNEW sets all points within the 3-D limits to be plottable after the current requested Transformations. The resulting images are larger than those produced by Scaling Mode 1, but the scales of successive pictures are not directly comparable.

VSFULL does not take effect until the next call of VSNEW.

Diagnostics

[VSFULL: Integer *I* out of range][1]

Default

CALL VSFULL(0) restores the default.

See also

VS3DLM, VSINIT, VSMAG, VSROT and VSTRAN.

SUBROUTINE VSINIT (LEVEL)

Name

VSINIT – to reset *SIMPLEPLOT ViSualization* defaults.

Availability Section 7, released version 2-13.

Argument

LEVEL	INTEGER expression	Level of initialization

LEVEL	*Initializes*
1	Rotation, Translation and Magnification
2	Scales
4	Drawing attributes

Description

VSINIT may be called to restore some or all defaults of attributes set in *SIMPLEPLOT ViSualization*:

The following subroutines are affected by VSINIT:

LEVEL	*Subroutines affected*
1	VSMAG, VSROT, VSTRAN
2	VS3DLM, VSEQX, VSEQY, VSEQZ, VSFIT, VSFITP, VSFULL, VSVRTP, VSVXYZ
4	VSAXFC, VSAXGC, VSBRBZ, VSBRFC, VSBRSZ, VSBRUP, VSCRGB, VSEDGC, VSEDGC, VSEDGV, VSFILC, VSCHLS, VSLBOX, VSLINC, VSPMC, VSPMMG, VSUTOC

VSINIT(2) takes effect at the next call to VSNEW. VSINIT(1) and VSINIT(4) take effect immediately.

Multiple categories of attributes may be set by adding the appropriate values of LEVEL, *eg.* CALL VSINIT(3) is equivalent to calling VSINIT(1) and VSINIT(2). CALL VSINIT(7) restores all defaults.

Diagnostics

[VSINIT: Integer *I* out of range][1]

See also

VSNEW.

SUBROUTINE VSIS (V, NX, NY, NZ, RVAL)

Name

`VSIS` – Draw the light-sourced IsoSurface of a specified data level from `REAL` data in a 3-D array in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-15.

Arguments

`V`	`REAL` 3-D array	Input data
`NX`, `NY`, `NZ`	`INTEGER` expressions	Dimensions of 3-D array
`RVAL`	`REAL` expression	Data level of surface

Description

`VSIS` draws a surface representing the data level `RVAL` in the array `V`. The lightness at each point on the surface is a value from 0.0 (no light) to 1.0 (maximum light) calculated from the gradient of the surface relative to the direction of the light source. By default, the range of lightnesses is mapped on to colour indices 1 to 15, which by default are a sequence of contrasting colours; `VSUTOC` may be called to specify a different range of indices, and `VSCGS` is available to set the palette for a chosen range of indices to a grey scale.

Diagnostics

`[VSIS: No current ViSualization picture]`^{1}
`[VSIS: Invalid array dimension]`^{1}

See also

`VSCGS`, `VSEDGV`, `VSEQX`, `VSEQY`, `VSEQZ`, `VSISU`, `VSISND`, `VSLSLD` and `VSLSSM`.

SUBROUTINE VSISND (ICODE)

Name

`VSISND` – specify whether IsoSurfaces should exclude six end planes of the data (see below).

Availability Section 7, released version 2-15.

Arguments

`ICODE`	`INTEGER` expression	Code number (see below)

`ICODE`	*End plane treatment*
0	End planes drawn without true gradients
1	End planes missed out

Description

Light-sourcing of IsoSurfaces depends on surface gradients at data points, which are derived from differences between data values on both sides. On each of the 6 boundary planes of the data, the differences cannot be calculated from points on both sides for one of the components of the gradient. When drawn surfaces extend to the end planes, a modified formula is used to calculate end plane gradients. Very large data sets can be rendered in stages by calling `VSIS` repeatedly to plot parts separately. When surfaces are built up in stages, discontinuities may be visible at the junctions between parts. The discontinuities can be eliminated by calling `VSISND(1)` to omit end planes, and providing overlapping data to the separate stages. Although `VSISND` is provided specifically for use with light-sourcing, for consistency, it affects `VSIS` and `VSISU` similarly, even though `VSISU` does not create the gradients which cause the problem.

Diagnostics

`[VSISND: Integer` *I* `out of range]`[1]

Default

`CALL VSISND(0)` restores the default.

See also

`VSIS`, and `VSISU`.

SUBROUTINE VSISU (V, U, NX, NY, NZ, RVAL)

Name

`VSISU` – draw contours of `REAL` data from one 3-D array on the IsoSurface of a specified data level from `REAL` data in another 3-D array in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-15.

Arguments

`V`, `U`	`REAL` 3-D arrays	Input data
`NX`, `NY`, `NZ`	`INTEGER` expressions	Dimensions of 3-D arrays
`RVAL`	`REAL` expression	Data level of surface

Description

`VSISU` draws contours of `REAL` data from the 3-D array `U` on the IsoSurface of data level `RVAL` from `REAL` data in the 3-D array `V` in a *SIMPLEPLOT ViSualization* image. The colours representing different contour levels are controlled by `VSUTOC`.

Diagnostics

`[VSIS: No current ViSualization picture]`[1]
`[VSIS: Invalid array dimension]`[1]

See also

`VSEDGV`, `VSEQX`, `VSEQY`, `VSEQZ`, `VSIS`, and `VSUTOC`.

SUBROUTINE VSK7H (VCHAR, HCHAR, ULEFT, URIGHT, CAP)

Name

`VSK7H` – to draw a horizontal key.

Availability Section 7, released version 2-14.

Arguments

`VCHAR`	`CHARACTER*1`	Vertical position of key
`HCHAR`	`CHARACTER*1`	Horizontal position of key
`ULEFT`	`REAL` expression	u value at left of key
`URIGHT`	`REAL` expression	u value at right of key
`CAP`	`STRING` expression	Single caption for key

Description

`VSK7H` draws a key to the current patterns representing user data values on tinted *SIMPLEPLOT ViSualization* plots. A horizontal rectangle gives a sample of colours, and a u-axis, labelled with the caption `CAP`, gives the data values.

The rectangle is drawn as a *SIMPLEPLOT ViSualization* image, and `VSK7H` should not be called between `VSNEW` and `VSOUT`. A box is drawn round the key unless cancelled by `CALL BOXKY(.FALSE.)`.

`VCHAR` and `HCHAR` are the single character initial letters of vertical and horizontal positions of the key. The possible interpretations of `VCHAR` and `HCHAR` are listed in Appendix T of the *SIMPLEPLOT Supplement*.

Diagnostics

`[VSK7H: Key omitted - active ViSualization picture]`[1]

See also

`BOXKY`, `VSBLTN`, `VSBRTN`, `VSK7V`, `VSPGTN`, `VSPLTN`, `VSRG`, `VSRGU`, `VSUTOC`, `VSXYZ`, `VSXYZU`, `VSZ` and `VSZU`.

SUBROUTINE VSK7V (VCHAR, HCHAR, UTOP, UBOTTM, CAP)

Name

`VSK7V` – to draw a vertical key.

Availability Section 7, released version 2-14.

Arguments

`VCHAR`	`CHARACTER*1`	Vertical position of key
`HCHAR`	`CHARACTER*1`	Horizontal position of key
`UTOP`	`REAL` expression	u value at top of key
`UBOTTM`	`REAL` expression	u value at bottom of key
`CAP`	`STRING` expression	Single caption for key

Description

`VSK7V` draws a key to the current patterns representing user data values on tinted *SIMPLEPLOT ViSualization* plots. A vertical rectangle gives a sample of colours, and a u-axis, labelled with the caption `CAP`, gives the data values.

The rectangle is drawn as a *SIMPLEPLOT ViSualization* image, and `VSK7V` should not be called between `VSNEW` and `VSOUT`. A box is drawn round the key unless cancelled by `CALL BOXKY(.FALSE.)`.

`VCHAR` and `HCHAR` are the single character initial letters of vertical and horizontal positions of the key. The possible interpretations of `VCHAR` and `HCHAR` are listed in Appendix T of the *SIMPLEPLOT Supplement*.

Diagnostics

`[VSK7V: Key omitted - active ViSualization picture]`^{1}

See also

`BOXKY`, `VSBLTN`, `VSBRTN`, `VSK7H`, `VSPGTN`, `VSPLTN`, `VSRG`, `VSRGU`, `VSUTOC`, `VSXYZ`, `VSXYZU`, `VSZ` and `VSZU`.

SUBROUTINE VSKEYS (VCHAR, HCHAR, LABARR, NARR, CAP)

Name

`VSKEYS` – to draw a complete key to a sequence of distinct shading patterns.

Availability Section 7, released version 2-14.

Arguments

`VCHAR`	`CHARACTER*1`	Vertical position of area
`HCHAR`	`CHARACTER*1`	Horizontal position of area
`LABARR`	`STRING` array	Set of captions
`NARR`	`INTEGER` expression	Number of labels in `LABARR`
`CAP`	`STRING` expression	Header caption for key

Description

`VSKEYS` constructs a complete key to the first `NARR` shading patterns of the currently defined sequence (see `SQSHAD` and `SHPATT`).

The key is positioned on the SIMPLEPLOT page in the same way as other keys; `VCHAR` and `HCHAR` are single characters representing the initial letters of vertical and horizontal positions. The possible interpretations of `VCHAR` and `HCHAR` are listed in Appendix T of the *SIMPLEPLOT Supplement*.

`CAP` provides a heading which, by default, is centred at the top of the key. The key is associated only with the current picture but area is masked to prevent overdrawing.

`VSKEYS` should not be called between `VSNEW` and `VSOUT`. A box is drawn round the key unless cancelled by `CALL BOXKY(.FALSE.)`.

See also

`ADDJST`, `BOXKY`, `DEFKEY`, `DEFKYW`, `DEFPOS`, `SHPATT`, `SHSET` `SQSHAD`, `VSK7H`, `VSK7V`.

SUBROUTINE VSLBOX (LEVEL)

Name

`VSLBOX` – to specify whether Limiting Boxes are to be drawn around *SIMPLEPLOT ViSualization* images.

Availability Section 7, released version 2-13.

Argument

`LEVEL`	`INTEGER` expression	Whether images are to be boxed

`LEVEL`	*Action taken*
0	No box drawn
1	Box is drawn

Description

By default, images are not boxed. `VSLBOX` specifies whether or not a 3-D box is to be drawn around every subsequent image.

After `CALL VSLBOX(1)`, a 3-D box is drawn around the area defined by the Limiting Box whenever `VSNEW` is called to start a new image. The box is drawn in the current Line Colour Index set by `VSLINC`. If an image has already been started, no box is drawn until the next image.

Default

`CALL VSLBOX(0)` restores the default.

Diagnostics

`[VSLBOX: Integer` *I* `out of range]`^{1}

See also

`VS3DLM`, `VSFULL`, `VSLINC` and `VSNEW`.

SUBROUTINE VSLINC (ICIX)

Name

`VSLINC` – to specify the colour index to be used by line drawing subroutines.

Availability Section 7, released version 2-13.

Argument

`ICIX`	`INTEGER` expression	Colour index for line drawing

Description

`VSLINC` sets the colour index for subsequent line drawing by `VSLBOX` or `VSPLDR`. In this release of software, data may be drawn in colour indices 0 to 95 inclusive. If colour index -1 is specified, subsequent lines are omitted.

Diagnostics

`[VSLINC: Colour index` *I* `out of range]`[1]

Default

`CALL VSLINC(1)` restores the default.

See also

`VSCHLS`, `VSLBOX`, `VSPLDR` and `VSCRGB`.

SUBROUTINE VSLSLD (THETA, PHI)

Name

`VSLSLD` – to specify the direction of a light source for a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-15.

Arguments

`THETA`	`REAL` expression	Horizontal angle
`PHI`	`REAL` expression	Vertical angle

Description

`VSLSLD` sets a direction for the *SIMPLEPLOT ViSualization* IsoSurface light in terms of 2 angles.

`THETA` is the horizontal angle in degrees between the Light Direction and the Origin on the x-y plane; `PHI` is the vertical angle in degrees between this plane and the Light Direction. By default, `THETA`=0.0, and `PHI`=0.0.

`THETA` $= 0.0$ along the x-axis (*ie.* at $y = 0.0$). `PHI` $= 0.0$ on the x-y plane (*ie.* at $z = 0.0$).

Default

`CALL VSLSLD(0.0, 0.0)` restores the default.

See also

`VSLSSM` and `VSIS`.

SUBROUTINE VSLSSM (METHOD)

Name

`VSLSSM` – to specify the shading method for light-sourcing to be used in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-15.

Arguments

`METHOD`	`INTEGER` expression	Shading method for light-sourcing (see below)

`METHOD`	*Shading method*
1	Uniform colour for each facet
2	Tinting of each facet

Description

`VSLSSM` controls which method of shading is used by `VSIS` when drawing the *SIMPLEPLOT ViSualization* image.

Diagnostics

`[VSLSSM: Integer` *I* `out of range]`[1]

Default

`CALL VSLSSM(1)` restores the default.

See also

`VSLSLD` and `VSIS`.

SUBROUTINE VSMAG (XFACT, YFACT, ZFACT)

Name

VSMAG – to magnify *SIMPLEPLOT ViSualization* coordinates along any or all of the x-y-z axes.

Availability Section 7, released version 2-13.

Arguments

XFACT	REAL expression	Magnification factor along the x-axis
YFACT	REAL expression	Magnification factor along the y-axis
ZFACT	REAL expression	Magnification factor along the z-axis

Description

By default, *SIMPLEPLOT ViSualization* interprets (x, y, z) coordinates literally. However, following a call to VSMAG, a magnification factor is applied to subsequent coordinates.

For example, following a call to VSMAG(XFACT,1.0,1.0), the point (x, y, z) is interpreted as $(x \times \texttt{XFACT}, y, z)$, and all subsequent objects are magnified by XFACT along the x-axis.

Similar magnification can be applied to the y and z-axes by setting YFACT and ZFACT to values other than 1.0.

After a point has been magnified, its (x, y, z) position changes. This means that it may not fit on the current picture, even if it lies within the limits specified by VS3DLM. VSFULL(2) may be called before VSNEW to ensure that (x, y, z) points within the Limiting Box before transformation stay on the picture.

Transformation is cumulative, so two successive calls to VSMAG(2.0,2.0,2.0) quadruple the size along each axis.

The transformed coordinate is a combination of the affects of VSMAG, VSROT and VSTRAN, which means that default magnification cannot be restored on its own. All transformation variables are restored by CALL VSINIT(1).

See also

VSFULL, VSINIT, VSROT and VSTRAN.

SUBROUTINE VSNEW

Name

`VSNEW` – to start a new *SIMPLEPLOT ViSualization* picture.

Availability Section 7, released version 2-13.

Arguments

None.

Description

`VSNEW` starts a new *SIMPLEPLOT ViSualization* picture and updates the variables which affect scaling. These are attributes set by `VS3DLM`, `VSEQX`, `VSEQY`, `VSEQZ`, `VSFIT`, `VSFITP`, `VSFULL`, `VSVRTP` and `VSVXYZ`.

`VSNEW` sets the underlying SIMPLEPLOT scales which are used by the internal *SIMPLEPLOT ViSualization* subroutines when constructing and transmitting an image. If another type of graph is drawn after a call to `VSNEW`, the SIMPLEPLOT scales should be set explicitly by `SCALES` or `EQSCAL`.

Backdrops and grids specified by `VSAXFC` and `VSAXGC` are drawn into the raster image when `VSNEW` is called.

Diagnostics

`[VSNEW: Radius too short - no perspective applied]`[1]

See also

`VS3DLM`, `VSAXFC`, `VSAXGC`, `VSFIT`, `VSFITP`, `VSFULL`, `VSVRTP` and `VSVXYZ`.

SUBROUTINE VSOUT

Name

`VSOUT` – to flush the raster image.

Availability Section 7, released version 2-13.

Arguments

None.

Description

`VSOUT` sends the raster image to the display device. With this release of *SIMPLEPLOT ViSualization* the internal raster image is not retained for subsequent re-processing, and `VSNEW` must be called before starting a new image.

Diagnostics

`[VSOUT: *n* incomplete tasks]`[1]

See also

`VSNEW`.

SUBROUTINE VSPGFE (XARRAY, YARRAY, ZARRAY, IEDGAR, NPTS)

Name

`VSPGFE` – to fill an edge-flagged planar polygon with the current fill index in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-15.

Arguments

`XARRAY`	`REAL` array	`NPTS` x values
`YARRAY`	`REAL` array	`NPTS` y values
`ZARRAY`	`REAL` array	`NPTS` z values
`IEDGAR`	`INTEGER` array	`NPTS` edge flags
`NPTS`	`INTEGER` expression	Number of points in polygon

Description

The current transformations are applied to the x-y-z data. The interior area is filled with the current fill colour index, and the edge is drawn according to the values in `IEDGAR`. Successive edge points are joined by straight lines.

The ith element of `IEDGAR` specifies the colour index for the edge joining the ith point of the polygon to the next point. The final element of `IEDGAR` relates to the edge from the last point to the first point. An edge colour index of -1 specifies that a gap is left in place of an edge; a colour index of -2 specifies that the fill colour continues to that boundary of the polygon.

The presence and colours of edges drawn by `VSPGFE` are independent of the current edge visibility and colour set by `VSEDGV` and `VSEDGC`.

Polygons must be planar, convex, and between 3 and 200 points. If `NPTS` $<$ 3, a diagnostic is issued and nothing is drawn.

Hidden line and surface removal are applied to all drawing.

Diagnostics

`[VSPGFE: No current ViSualization picture]`[1]
`[VSPGFE: Invalid array dimension]`[1]

See also

`VSPGFL`, `VSPGTE`, and `VSPGTN`.

SUBROUTINE VSPGFL (XARRAY, YARRAY, ZARRAY, NPTS)

Name

`VSPGFL` – to fill a planar polygon with the current fill index in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

`XARRAY`	`REAL` array	`NPTS` x values
`YARRAY`	`REAL` array	`NPTS` y values
`ZARRAY`	`REAL` array	`NPTS` z values
`NPTS`	`INTEGER` expression	Number of points in polygon

Description

The current transformations are applied to the x-y-z data. Successive points are joined by a straight line, and the interior area is filled with the current fill colour index. If the Edge Visibility is ON, following a call to `VSEDGV(1)`, the edge is drawn in the current Edge Colour Index.

Polygons must be planar, convex and between 3 and 200 points. If `NPTS` < 3, a diagnostic is issued and nothing is drawn.

Hidden line and surface removal are applied to all drawing.

Diagnostics

`[VSPGFL: No current ViSualization picture]`[1]
`[VSPGFL: Invalid array dimension]`[1]

See also

`VSEDGC`, `VSEDGV`, `VSFILC` and `VSPGTN`.

SUBROUTINE VSPGTE (XARRAY, YARRAY, ZARRAY, UARRAY, IEDGAR, NPTS)

Name

VSPGTE – to tint an edge-flagged planar polygon from user data values in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-15.

Arguments

XARRAY	REAL array	NPTS x values
YARRAY	REAL array	NPTS y values
ZARRAY	REAL array	NPTS z values
UARRAY	REAL array	NPTS u values
IEDGAR	INTEGER array	NPTS edge flags
NPTS	INTEGER expression	Number of points in polygon

Description

The current transformations are applied to the x-y-z data. The interior area is tinted with values interpolated from UARRAY, and the edge is drawn according to the values in IEDGAR. Successive edge points are joined by straight lines.

The ith element of IEDGAR specifies the colour index for the edge joining the ith point of the polygon to the next point. The final element of IEDGAR relates to the edge from the last point to the first point. An edge colour index of -1 specifies that a gap is left in place of an edge; a colour index of -2 specifies that the interpolated fill colour continues to that boundary of the polygon.

The presence and colours of edges drawn by VSPGTE are independent of the current edge visibility and colour set by VSEDGV and VSEDGC.

Polygons must be planar, convex, and between 3 and 200 points. If NPTS $<$ 3, a diagnostic is issued and nothing is drawn.

Hidden line and surface removal are applied to all drawing.

Diagnostics

[VSPGTE: No current ViSualization picture][1]
[VSPGTE: Invalid array dimension][1]

See also

VSPGFE, VSPGFL, and VSPGTN.

SUBROUTINE VSPGTN (XARRAY, YARRAY, ZARRAY, UARRAY, NPTS)

Name

`VSPGTN` – to tint a planar polygon from user data values in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

`XARRAY`	`REAL` array	`NPTS` x values
`YARRAY`	`REAL` array	`NPTS` y values
`ZARRAY`	`REAL` array	`NPTS` z values
`UARRAY`	`REAL` array	`NPTS` user data values
`NPTS`	`INTEGER` expression	Number of points in polygon

Description

The current transformations are applied to the x-y-z data. Successive points are joined by a straight line, and the interior area is filled by changing colours interpolated from the corresponding user data values. If the Edge Visibility is ON, following a call to `VSEDGV(1)`, the edge is drawn in the current Edge Colour Index.

Polygons must be planar, convex and between 3 and 200 points. If `NPTS` < 3, a diagnostic is issued and nothing is drawn.

Hidden line removal is applied to all drawing.

Diagnostics

`[VSPGTN: No current ViSualization picture]`[1]
`[VSPGTN: Invalid array dimension]`[1]

See also

`VSEDGC`, `VSEDGV`, `VSPGFL` and `VSUTOC`.

SUBROUTINE VSPLDR (XARRAY, YARRAY, ZARRAY, NPTS)

Name

`VSPLDR` – to draw a polyline through a succession of (x, y, z) data points in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

`XARRAY`	`REAL` array	`NPTS` x values
`YARRAY`	`REAL` array	`NPTS` y values
`ZARRAY`	`REAL` array	`NPTS` z values
`NPTS`	`INTEGER` expression	Number of points in polyline

Description

The current transformations are applied to the input data and successive points are joined by a straight line, using the current line colour index. Hidden line removal is applied to all drawing.

If `NPTS` < 2, a diagnostic is issued and nothing is drawn.

Diagnostics

`[VSPLDR: No current ViSualization picture]`^{1}
`[VSPLDR: Invalid array dimension]`^{1}

See also

`VSLINC` and `VSPLTN`.

SUBROUTINE VSPLTN (XARRAY, YARRAY, ZARRAY, UARRAY, NPTS)

Name

`VSPLTN` – to tint a polyline based on user data values in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

`XARRAY`	`REAL` array	`NPTS` x values
`YARRAY`	`REAL` array	`NPTS` y values
`ZARRAY`	`REAL` array	`NPTS` z values
`UARRAY`	`REAL` array	`NPTS` user data values
`NPTS`	`INTEGER` expression	Number of points in polyline

Description

The current *SIMPLEPLOT ViSualization* transformations are applied to the x-y-z data. Successive points are joined by a straight line, using changing colours interpolated from the corresponding user data values.

If `VSPLTN` < 2, a diagnostic is issued and nothing is drawn.

Hidden line removal is applied to all drawing.

Diagnostics

`[VSPLTN: No current ViSualization picture]`[1]
`[VSPLTN: Invalid array dimension]`[1]

See also

`VSPLDR` and `VSUTOC`.

SUBROUTINE VSPMC (ICIX)

Name

`VSPMC` – to specify the colour index for subsequent markers.

Availability Section 7, released version 2-14.

Argument

`ICIX`	`INTEGER` expression	Colour index for markers

Description

`VSPMC` sets the colour index for subsequent marker symbols. In this release of software, data may be drawn in colour indices 0 to 95 inclusive. If colour index -1 is specified, subsequent markers are omitted.

Diagnostics

`[VSPMC: Colour index` *I* `out of range]`^{1}

Default

`CALL VSPMC(1)` restores the default.

See also

`VSCHLS`, `VSPMDR` and `VSCRGB`.

SUBROUTINE VSPMDR (XARRAY, YARRAY, ZARRAY, NPTS, MKTYPE)

Name

`VSPMDR` – to draw a set of markers of type `MKTYPE` in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-14.

Arguments

`XARRAY`	`REAL` array	`NPTS` x coordinates
`YARRAY`	`REAL` array	`NPTS` y coordinates
`ZARRAY`	`REAL` array	`NPTS` z coordinates
`NPTS`	`INTEGER` expression	Number of points in polymarker
`MKTYPE`	`INTEGER` expression	Marker type

Description

`VSPMDR` draws a polymarker – a set of marker symbols – of type `MKTYPE`, centred on each of the points held in the x-y-z arrays. The current transformations are applied to the x-y-z data.

Markers are drawn in the current marker colour with the current marker magnification factor applied by `VSPMMG`. Markers are drawn face on to the viewer, orthogonal to the line of sight.

By default, markers are scaled to occupy 2% of the smaller of the width or height of the picture size, with a minimum size of 0.14cm. When perspective is applied to the *SIMPLEPLOT ViSualization* image, the target marker size applies at a point half way between the front and back of the image. The size of markers decreases away from the viewer.

Diagnostics

`[VSPMDR: No Current ViSualization picture]`[1]
`[VSPMDR: Invalid array dimension]`[1]
`[VSPMDR: Integer` *I* `out of range]`[1]

See also

`VSPMC` and `VSPMMG`.

SUBROUTINE VSPMMG (FACTOR)

Name

`VSPMMG` – to specify the scaling factor for subsequent markers.

Availability Section 7, released version 2-14.

Argument

`FACTOR`	`REAL` expression	Factor by which marker size to be scaled

Description

By default, markers are scaled to occupy 2% of the smaller of the width or height of the picture size, with a minimum marker size of 0.14cm. When perspective is applied to the *SIMPLEPLOT ViSualization* image, the target marker size applies at a point half way between the front and back of the image. The size of markers decreases away from the viewer.

After `CALL VSPMMG(FACTOR)`, the size of markers is increased by a factor of `FACTOR`. `FACTOR` must be greater than 0.0. If it is not, the default is used.

Diagnostics

`[VSPMMG: Value` *R* `out of range]`1

Default

`CALL VSPMMG(0.0)` restores the default.

See also

`VSPMDR`.

SUBROUTINE VSPOP

Name

`VSPOP` – to recall a transformation matrix saved by `VSPUSH`.

Availability Section 7, released version 2-13.

Arguments

None.

Description

`VSPUSH` saves the current matrix of transformations, set by `VSMAG`, `VSROT` and `VSTRAN`. At the next call of `VSPOP`, subsequent calls to these subroutines are ignored. The modelling transformations active at the call of `VSPUSH` are applied.

`VSPOP` may be called as many times as there have been calls to `VSPUSH`. The first call to `VSPOP` recalls the matrix saved most recently, with subsequent calls remembering previous matrices. If `VSPUSH` has been called n times, the first matrix saved is recalled by the nth call to `VSPOP`.

If there is an attempt to recall more matrices than have been saved, a diagnostic is issued and the current matrix is unchanged.

Diagnostics

`[VSPOP: No more modelling transformations]`^{1}

See also

`VSMAG`, `VSPUSH`, `VSROT` and `VSTRAN`.

SUBROUTINE VSPUSH

Name

`VSPUSH` – to save current *SIMPLEPLOT ViSualization* transformation matrix.

Availability Section 7, released version 2-13.

Arguments

None.

Description

`VSPUSH` saves the current matrix of transformations, set by `VSMAG`, `VSROT` and `VSTRAN`. At the next call of `VSPOP`, subsequent calls to these subroutines are ignored. The modelling transformations active at the call of `VSPUSH` are applied.

`VSPUSH` may be called more than once. If a series of calls to `VSPUSH` are made, the first call to `VSPOP` recalls the matrix most recently pushed. Subsequent calls to `VSPOP` each recall the matrix most previously pushed but not yet popped.

Up to 20 matrices may be pushed at any one time. If an attempt is made to push more, a diagnostic is issued and the matrix is not remembered.

Diagnostics

`[VSPUSH: Too many modelling transformations]`^{1}

See also

`VSMAG`, `VSPOP`, `VSROT` and `VSTRAN`.

SUBROUTINE VSRG (Z2ARR, NX, NY)

Name

`VSRG` – to draw a perspective surface picture from a 2-D array in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

`Z2ARR`	`REAL` 2-D array	`NX`$\times$`NY` array of user (z) values
`NX`, `NY`	`INTEGER` expression	Dimensions of 2-D data array

Description

The data values held in the array `Z2ARR` are drawn as a filled surface. The colours representing different heights are controlled by `VSUTOC`.

By default, the x, y and z ranges of the data are mapped onto the Limiting Box. If `VS3DLM` has been called to set the scales explicitly, the z values map onto the specified limits, and the x and y scales can be controlled by `VSEQX` and `VSEQY`.

Diagnostics

`[VSRG: No current ViSualization picture]`[1]
`[VSRG: Invalid array dimension]`[1]

See also

`VS3DLM`, `VSEDGC`, `VSEDGV`, `VSEQX`, `VSEQY`, `VSRGU` and `VSUTOC`.

SUBROUTINE VSRGU (Z2ARR, U2ARR, NX, NY)

Name

VSRGU – to draw contours of data from one 2-D array on a surface picture from another 2-D array in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

Z2ARR	REAL 2-D array	NX×NY array of (z) values
U2ARR	REAL 2-D array	NX×NY array of data values
NX, NY	INTEGER expressions	Dimensions of 2-D arrays

Description

Contours representing U2ARR are drawn on the surface defined by Z2ARR. The colours representing different contour levels are controlled by VSUTOC.

By default, the x, y and z ranges of the data are mapped onto the Limiting Box. If VS3DLM has been called to set the scales explicitly, the z values map onto the specified limits, and the x and y scales can be controlled by VSEQX and VSEQY.

Diagnostics

```
[VSRGU: No current ViSualization picture]
[VSRGU: Invalid array dimension]
```
[VSRGU: No current ViSualization picture] ^{1}, [VSRGU: Invalid array dimension] ^{1}

See also

VS3DLM, VSEDGC, VSEDGV, VSEQX, VSEQY, VSRG and VSUTOC.

SUBROUTINE VSROT (CHAXIS, ROTATE)

Name

`VSROT` – to rotate *SIMPLEPLOT ViSualization* coordinates around one of the x-y-z axes.

Availability Section 7, released version 2-13.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`ROTATE`	`REAL` expression	Rotation (in degrees)

`CHAXIS`	*Type of axis*
`'X3'`	3-D perspective x-axis
`'Y3'`	3-D perspective y-axis
`'Z3'`	3-D perspective z-axis

Description

By default, *SIMPLEPLOT ViSualization* interprets (x, y, z) coordinates literally. However, following a call to `VSROT`, subsequent coordinates are rotated around one of the axes.

`CHAXIS` signifies axis type. If you are looking down the axis in a positive direction, rotation is clockwise.

Following a call to `VSROT('X3',45.0)`, the point (x, y, z) is rotated by 45° around the x-axis. Transformation is cumulative, so two successive calls to `VSROT('Y3',30.0)` rotate points 60° around the y-axis.

After a point has been rotated, its (x, y, z) position changes. This means that it may not fit on the current picture, even if it lies within the limits specified by `VS3DLM`. `VSFULL(1)` or `VSFULL(2)` may be called before `VSNEW` to ensure that (x, y, z) points within the Limiting Box before transformation stay on the picture.

The transformed coordinate is a combination of the effects of `VSMAG`, `VSROT` and `VSTRAN`, which means that default rotation cannot be restored on its own. All transformation variables are restored by `CALL VSINIT(1)`.

Diagnostics

`[VSROT: Invalid string` *`XX`*`]`[1]

See also

`VSFULL`, `VSINIT`, `VSMAG` and `VSTRAN`.

SUBROUTINE VSTRAN (XTRANS, YTRANS, ZTRANS)

Name

`VSTRAN` – to translate *SIMPLEPLOT ViSualization* coordinates along any or all of the x-y-z axes.

Availability Section 7, released version 2-13.

Arguments

`XTRANS`	`REAL` expression	Translation along the x-axis
`YTRANS`	`REAL` expression	Translation along the y-axis
`ZTRANS`	`REAL` expression	Translation along the z-axis

Description

By default, *SIMPLEPLOT ViSualization* interprets (x, y, z) coordinates literally. However, following a call to `VSTRAN`, a translation value is applied to subsequent coordinates.

For example, following a call to `VSTRAN(XTRANS,0.0,0.0)`, the point (x, y, z) is interpreted as $(x + \texttt{XTRANS}, y, z)$, and all subsequent objects are translated by `XTRANS` along the x-axis.

Similar translation can be applied to the y and z-axes by setting `YTRANS` and `ZTRANS` to non-zero values.

After a point has been translated, its (x, y, z) position changes. This means that it may not fit on the current picture, even if it lies within the limits specified by `VS3DLM`. `VSFULL(2)` may be called before `VSNEW` to ensure that (x, y, z) points within the Limiting Box before transformation stay on the picture.

Transformation is cumulative, so two successive calls to `VSTRAN(2.0,2.0,2.0)` move points 4.0 units along each axis.

The transformed coordinate is a combination of the effects of `VSMAG`, `VSROT` and `VSTRAN`, which means that default translation cannot be restored on its own. All transformation variables are restored by `CALL VSINIT(1)`.

See also

`VSFULL`, `VSINIT`, `VSMAG` and `VSROT`.

SUBROUTINE VSUTOC (UMIN, UMAX, ICIX1, ICIX2)

Name

`VSUTOC` – to define mapping from user data values to colour indices.

Availability Section 7, released version 2-13.

Arguments

`UMIN`, `UMAX`	`REAL` expressions	Range of user data values to be mapped
`ICIX1`, `ICIX2`	`INTEGER` expressions	Corresponding range of colour indices

Description

By default, *SIMPLEPLOT ViSualization* maps the full range of user data given in output subroutines on to colour indices 1 to 15. `VSUTOC` selects different ranges of colour associated with a *SIMPLEPLOT ViSualization* picture.

`VSUTOC` maps data within a given range onto a specified set of pen indices. After `CALL VSUTOC(UMIN,UMAX,ICIX1,ICIX2)`, each index from `ICIX1` to (`ICIX2` − 1) is associated with an equal range of data between `UMIN` and `UMAX`. Each range measures $\frac{\texttt{UMAX-UMIN}}{\texttt{ICIX2-ICIX1}}$ units.

Index `ICIX2` represents all data values greater than or equal to `UMAX`. Index `ICIX1` is also used for data less than `UMIN`.

The default action may be restored by setting `UMIN` greater than `UMAX`.

In this release of software, data may be mapped on to colour indices 0 to 95 inclusive.

Diagnostics

`[VSUTOC: Colour index` *I* `out of range]`[1]

Default

`CALL VSUTOC(0.0, -1.0, 0, 0)` restores the default.

See also

`VSCRGB` and `VSCHLS`.

SUBROUTINE VSVRTP (RADIUS, THETA, PHI)

Name

`VSVRTP` – to set the Viewing Position in terms of a radius and 2 angles for a projected picture.

Availability Section 7, released version 2-13.

Arguments

`RADIUS`	`REAL` expression	Distance of the Viewing Position from the Origin
`THETA`	`REAL` expression	Horizontal angle
`PHI`	`REAL` expression	Vertical angle

Description

`VSVRTP` sets a Viewing Position for *SIMPLEPLOT ViSualization* in terms of a radius and 2 angles.

`RADIUS` is the distance between the Viewing Position and the Origin, where the maximum possible distance from the Origin to any point in the Limiting Box is $\sqrt{3}$. The Origin and Limiting Box are set by `VS3DLM`.

`RADIUS` is interpreted as being positive. A radius of $-R$ is equivalent to a radius of $+R$.

If the Viewing Position is too close to the Origin, points could be projected to infinity, or be otherwise unplottable. To prevent this happening, if the absolute value of `RADIUS` $\leq \sqrt{3}$, a diagnostic is issued at the next call to `VSNEW`, and no perspective is applied.

`THETA` is the horizontal angle in degrees between the Viewing Position and the Origin on the x-y plane; `PHI` is the vertical angle in degrees between this plane and the Line of Sight. By default, `THETA`=30.0, and `PHI`=15.0.

`THETA` $= 0.0$ along the x-axis (*ie.* at $y = 0.0$). `PHI` $= 0.0$ on the x-y plane (*ie.* at $z = 0.0$).

The change in Viewing Position does not take effect until the next call of `VSNEW`. `VSVRTP` overrides any previous call to `VSVXYZ`.

Default

`CALL VSVRTP(0.0, 30.0, 15.0)` restores the default.

See also

`VS3DLM`, `VSINIT`, `VSNEW` and `VSVXYZ`.

SUBROUTINE VSVXYZ (XVIEW, YVIEW, ZVIEW)

Name

`VSVXYZ` – to set the Viewing Position for a projected picture in terms of (x, y, z).

Availability Section 7, released version 2-13.

Arguments

`XVIEW`	`REAL` expression	x component of Viewing Position
`YVIEW`	`REAL` expression	y component of Viewing Position
`ZVIEW`	`REAL` expression	z component of Viewing Position

Description

`VSVXYZ` sets a Viewing Position for *SIMPLEPLOT ViSualization* in terms of (x, y, z).

`XVIEW`, `YVIEW` and `ZVIEW` are in the underlying three-dimensional coordinates before transformation.

If the Viewing Position is too close to the Origin, points could be projected to infinity, or be otherwise unplottable. To prevent this from happening, the distance from (`XVIEW`, `YVIEW`, `ZVIEW`) to the Origin should be greater than the radius of the Bounding Sphere specified by `VS3DLM`.

If

$$\mathtt{XVIEW}^2 + \mathtt{YVIEW}^2 + \mathtt{ZVIEW}^2 < \\ (\frac{\mathtt{XSTOP-XSTART}}{2})^2 + (\frac{\mathtt{YSTOP-YSTART}}{2})^2 + (\frac{\mathtt{ZSTOP-ZSTART}}{2})^2$$

a diagnostic is issued at the next call to `VSNEW`, and no perspective is applied.

The change in Viewing Position does not take effect until the next call of `VSNEW`. `VSVXYZ` overrides any previous call to `VSVRTP`.

Default

See `VSVRTP`.

See also

`VS3DLM`, `VSINIT`, `VSNEW` and `VSVRTP`.

SUBROUTINE VSXYZ (X2ARR, Y2ARR, Z2ARR, NX, NY)

Name

`VSXYZ` – to draw a surface picture from three 2-D arrays in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

`X2ARR`, `Y2ARR`, `Z2ARR`	`REAL` 2-D arrays	`NX`×`NY` arrays of user (x, y, z) values
`NX`, `NY`	`INTEGER` expression	Dimensions of 2-D arrays

Description

The quadrilateral data structure held in the x-y-z arrays is represented by a surface. Elements are constructed from each of the 4 adjacent elements in the 2-D arrays. The colours representing different z-values are controlled by `VSUTOC`.

Diagnostics

`[VSXYZ: No current ViSualization picture]`[1]
`[VSXYZ: Invalid array dimension]`[1]

See also

`VS3DLM`, `VSEDGC`, `VSEDGV`, `VSUTOC` and `VSXYZU`.

SUBROUTINE VSXYZU (X2ARR, Y2ARR, Z2ARR, U2ARR, NX, NY)

Name

VSXYZU – to draw contours of data in one 2-D array on a surface picture from three 2-D arrays in a *SIMPLEPLOT ViSualization* image.

Availability Section 7, released version 2-13.

Arguments

X2ARR, Y2ARR, Z2ARR	REAL 2-D arrays	NX×NY arrays of user (x, y, z) values
U2ARR	REAL 2-D array	NX×NY User data values
NX, NY	INTEGER expression	Dimensions of 2-D arrays

Description

The quadrilateral data structure held in the *x*-*y*-*z* arrays is represented by a surface. Contours from the corresponding U2ARR array are drawn on this surface. Elements are constructed from each of the four adjacent elements in the *x*-*y*-*z* arrays. The colours representing different *u* values are controlled by VSUTOC.

Diagnostics

[VSXYZU: No current ViSualization picture]^1
[VSXYZU: Invalid array dimension]^1

See also

VS3DLM, VSEDGC, VSEDGV, VSUTOC and VSXYZ.

SUBROUTINE VSZ (XARR, YARR, ZARR, NPTS, I2ARR, NNODES, NELS)

Name

`VSZ` – to draw a surface from 3-D data structured into elements.

Availability Section 7, released version 2-13.

Arguments

`XARR`, `YARR`, `ZARR`	`REAL` arrays	(x, y, z) coordinates of data values in parallel arrays
`NPTS`	`INTEGER` expression	Number of data points
`I2ARR`	`INTEGER` 2-D array	Data element structure
`NNODES`	`INTEGER` expression	Number of nodes per element
`NELS`	`INTEGER` expression	Number of elements

Description

`VSZ` draws a surface representing data structured into elements and held in parallel arrays, `XARR(NPTS)`, `YARR(NPTS)` and `ZARR(NPTS)`.

Before using `VSZ`, the element array `I2ARR(NNODES,NELS)`, must be set up. The SIMPLEPLOT-PLUS subroutines `ZZORDR` or `ZZORDN` can only be used for this purpose if the surface is such that there is a single function value z for any (x, y), otherwise the user must construct an index array identifying the indices in the x, y and z arrays to each node in each element.

The colours representing different contour levels are controlled by `VSUTOC`.

Diagnostics

`[VSZ: No current ViSualization picture]`[1]
`[VSZ: Invalid array dimension]`[1]

See also

`VSEDGC`, `VSEDGV`, `VSUTOC`, and `VSZU`.

SUBROUTINE VSZU (XARR, YARR, ZARR, UARR, NPTS, I2ARR, NNODES, NELS)

Name

VSZU – to draw contours of data in an array on a surface from 3-D data structured into elements.

Availability Section 7, released version 2-13.

Arguments

X2ARR, Y2ARR, Z2ARR	REAL arrays	(x, y, z) coordinates of data values in parallel arrays
U2ARR	REAL array	NPTS user data values
NPTS	INTEGER expression	Number of data points
I2ARR	INTEGER 2-D array	Data element structure
NNODES	INTEGER expression	Number of nodes per element
NELS	INTEGER expression	Number of elements

Description

VSZU draws a contour map of the data values in UARR(NPTS), on the surface representing data structured into planar elements and held in parallel arrays XARR(NPTS), YARR(NPTS) and ZARR(NPTS).

Before using VSZU, the element array I2ARR(NNODES,NELS) must have been set up. The SIMPLEPLOT-PLUS subroutines ZZORDR or ZZORDN can be used for this purpose if the surface is such that there is a single function value z for any (x, y). Otherwise, the user must construct an index array identifying the indices in the x, y and z arrays to each node in each element.

The colours representing different contour levels are controlled by VSUTOC.

Diagnostics

```
[VSZU: No current ViSualization picture]
[VSZU: Invalid array dimension]
```
[1] [1]

See also

VSEDGC, VSEDGV, VSUTOC and VSZ.

B. Brief specifications of other subroutines

This appendix gives brief formal specifications for the SIMPLEPLOT subroutines used in this manual. It also includes the AXis subroutines listed in Table 2.2.10 Full specifications are given in the *SIMPLEPLOT Reference manual*.

SUBROUTINE AXCLR (CHAXIS, ILEVEL)

Name

`AXCLR` – to specify the level of annotation drawn with the axis.

Availability Section 1, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`ILEVEL`	`INTEGER` expression	Level of axis annotation

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√

`ILEVEL`	*Effect*
0	Tick marks and annotation labels are drawn
1	Only axis line drawn

SUBROUTINE AXIS7 (CHAXIS, CAP)
SUBROUTINE AXIS (CHAXIS, CAP, NCAP)

Name

`AXIS7` – to draw an axis on the current picture.

Availability Section 1, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`CAP`	`STRING` expression	Axis caption
`NCAP`	`INTEGER` expression	Number of characters in `CAP`

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	√	√	√	√	√	√	×

SUBROUTINE AXLAB7 (CHAXIS, W, CAP)
SUBROUTINE AXLAB (CHAXIS, W, CAP, NCAP)

Name

`AXLAB7` – to draw a single axis annotation label and tick mark.

Availability Section 1, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`W`	`REAL` expression	Position along axis in units of the plotting scales
`CAP`	`STRING` expression	Annotation label
`NCAP`	`INTEGER` expression	Number of characters in `CAP`

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	×	√	√	√	√	√	×

SUBROUTINE AXLBAN (CHAXIS, ACHAR)

Name

`AXLBAN` – to specify the style of annotation labels on an axis.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`ACHAR`	`CHARACTER*1`	Position of annotation

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	√	√	√	√^{U}	√^{U}	√^{U}	×

U: `'P'`receding, `'F'`ollowing and `'I'`nside unavailable

`ACHAR`	*Style*
`'D'`	Default – towards the outside
`'F'`	Following axis (in direction of other axis)
`'I'`	Towards the inside of the picture
`'N'`	None – annotation labels omitted
`'O'`	Towards the outside of the picture
`'P'`	Preceding axis (in direction of other axis)

SUBROUTINE AXLBGP (CHAXIS, ILEVEL)

Name

`AXLBGP` – to specify the level of axis annotation near an intersection.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`ILEVEL`	`INTEGER` expression	Level of annotation at intersection

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	×	√	√	√	√	×	√	√	√	√	√	√

`ILEVEL`	*Effect*
0	All annotation drawn
1	Annotation omitted near intersection

SUBROUTINE AXLBSL (CHAXIS, SCHAR)

Name

`AXLBSL` – to specify the slope of used-defined axis annotation.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`SCHAR`	`CHARACTER*1`	Direction of slope of labels

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	×	√	√	√	√	√	√	√	×	×	×	×

`SCHAR`	*Direction of label*
`'H'`	Horizontal
`'V'`	Vertical
`'S'`	Sloping
`'D'`	Default (different for labels of different lengths)

SUBROUTINE AXLBSP (CHAXIS, CHAR1, CHAR2)

Name

`AXLBSP` – to specify separators between label components of *time-date* axis annotation.

Availability Section plus, released version 2-12.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`CHAR1`	`CHARACTER*1`	Separator between first pair
`CHAR2`	`CHARACTER*1`	Separator between second pair

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	×	√	√	√	√	√	×

SUBROUTINE AXLBTM (CHAXIS, COMP1, COMP2, COMP3)

Name

`AXLBTM` – to specify components for *time-date* axis annotation.

Availability Section plus, released version 2-12.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`COMP1`	`CHARACTER*2`	Component for first label position
`COMP2`	`CHARACTER*2`	Component for second label position
`COMP3`	`CHARACTER*2`	Component for third label position

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	×	√	√	√	√	√	×

SUBROUTINE AXLBTP (CHAXIS, CHTYPE)

Name

`AXLBTP` – to specify *numeric* or *time-date* axis annotation.

Availability Section plus, released version 2-12.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`CHTYPE`	`CHARACTER*1`	Annotation type – `'N'`umeric, `'D'`ate or `'T'`ime

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	×	√	√	√	√	√	×

`CHTYPE`	*Annotation type*
`'N'`	Numeric (default)
`'D'`	Date
`'T'`	Time

SUBROUTINE AXLOCN (CHAXIS, LCHAR)

Name

`AXLOCN` – to specify the location of an axis relative to the picture.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`LCHAR`	`CHARACTER*1`	Location of axis

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√

`LCHAR`	*Location of axis*
`'P'`	Preceding the other axis
`'F'`	Following the other axis
`'D'`	Default

SUBROUTINE AXMAJ (CHAXIS, W)

Name

`AXMAJ` – to draw a single major axis subdivision.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`W`	`REAL` expression	Position along axis in units of the plotting scales

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	×	√	√	√	√	√	×

SUBROUTINE AXMIN (CHAXIS, W)

Name

`AXMIN` – to draw a single minor axis subdivision.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`W`	`REAL` expression	Position along axis in units of the plotting scales

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$

SUBROUTINE AXRNGE (CHAXIS, START, STOP)

Name

`AXRNGE` – to specify the subrange over which an axis is to be drawn.

Availability Section 1, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`START`	`REAL` expression	Start of axis in units of scale
`STOP`	`REAL` expression	End of axis in units of scale

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$

SUBROUTINE AXSBDV (CHAXIS, OFFSET, DELTA)

Name

`AXSBDV` – to specify the interval at which a linear axis is to be subdivided and annotated.

Availability Section 1, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`OFFSET`	`REAL` expression	Offset value for subdivisions
`DELTA`	`REAL` expression	Interval between subdivisions of axis in units of the scale

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	$\surd^L$	$\surd^L$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$

L: Linear scales only

SUBROUTINE AXSBMN (CHAXIS, OFFSET, DELTA)

Name

`AXSBMN` – to specify the minor subdivisions on a linear axes independently of major subdivisions.

Availability Section 4, released version 2-12.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`OFFSET`	`REAL` expression	Offset value for minor subdivisions in units of the plotting scale
`DELTA`	`REAL` expression	Interval between minor subdivisions in units of the plotting scale

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√^{L}	√^{L}	√	√	√	√	√	√	×	√	√	√	√	√	√

L: Linear scales only

SUBROUTINE AXSBTK (CHAXIS, TCHAR)

Name

`AXSBTK` – to specify the style of tick marks on an axis.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`TCHAR`	`CHARACTER*1`	Direction of tick marks

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
`CHAXIS`	`XC`	`YC`	`RP`	`AP`	`XI`	`YI`	`ZI`	`NB`	`LB`	`NW`	`LW`	`X3`	`Y3`	`Z3`	`U3`
	√	√	√	√	√	√	√	√	√	√	√	√^{U}	√^{U}	√^{U}	√^{U}

U: `'P'`receding, `'F'`ollowing and `'I'`nside unavailable

`TCHAR`	*Meaning*
`'A'`	Straddling across the axis
`'D'`	Default – towards the outside
`'F'`	Following axis
`'I'`	Towards the inside of the picture
`'N'`	None – tick marks omitted
`'O'`	Towards the outside of the picture
`'P'`	Preceding axis

SUBROUTINE AXSUBS (CHAXIS, NMAJOR, NMINOR)

Name

`AXSUBS` – to specify the numbers of major and minor subdivisions on linear axes.

Availability Section 4, released version 2-11.

Arguments

`CHAXIS`	`CHARACTER*2`	Axis type
`NMAJOR`	`INTEGER` expression	Number of major subdivisions
`NMINOR`	`INTEGER` expression	Number of minor subdivisions

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
CHAXIS	XC	YC	RP	AP	XI	YI	ZI	NB	LB	NW	LW	X3	Y3	Z3	U3
	$\surd^L$	$\surd^L$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$

L: Linear scales only

SUBROUTINE AXTXT7 (CHAXIS, W1, STEP, LABARR, NARR)
SUBROUTINE AXTXT (CHAXIS, W1, STEP, LABARR, NARR, NCAP)

Name

AXTXT7 – to draw a set of axis annotation labels and tick marks.

Availability Section 4, released version 2-11.

Arguments

CHAXIS	CHARACTER*2	Axis type
W1	REAL expression	Position for first label
STEP	REAL expression	Interval between labels
LABARR	STRING array	Set of labels
NARR	INTEGER expression	Number of labels in LABARR
NCAP	INTEGER expression	Maximum number of characters in any label

axis type	Cartesian		Polar		Isometric			Bars		Water		ViSualization			
CHAXIS	XC	YC	RP	AP	XI	YI	ZI	NB	LB	NW	LW	X3	Y3	Z3	U3
	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\times$

SUBROUTINE BOXPIC (TORF)

Name

BOXPIC – to specify whether boxes are to be drawn around individual pictures.

Availability Section 4, released version 2-7.

Argument

TORF	LOGICAL expression	Whether pictures are to be boxed

SUBROUTINE CP7LB (X, Y, CAP)
SUBROUTINE CAPLB (X, Y, CAP, NCAP)

Name

CP7LB – to draw a caption at (x, y).

Availability Section 4, released version 2-6.

Arguments

X, Y	REAL expressions	Coordinates of a point, specified in units of the plotting scales
CAP	STRING expression	Caption
NCAP	INTEGER expression	Number of characters in CAP

SUBROUTINE CP7PT (X, Y, MKTYPE, CAP)
SUBROUTINE CAPPT (X, Y, MKTYPE, CAP, NCAP)

Name

`CP7PT` – to draw a marker symbol at (x, y) with a caption.

Availability Section 1, released before version 2-5.

Arguments

`X, Y`	`REAL` expressions	Coordinates of a point specified in units of the plotting scales
`MKTYPE`	`INTEGER` expression	Type of marker symbol
`CAP`	`STRING` expression	Caption
`NCAP`	`INTEGER` expression	Number of characters in `CAP`

SUBROUTINE ENDPLT

Name

`ENDPLT` – to close the plotting device and SIMPLEPLOT at the end of plotting.

Availability Section 1, released before version 2-5.

Arguments

None.

SUBROUTINE GROUP (NHORIZ, NVERT)

Name

`GROUP` – to specify how pictures are to be grouped on the SIMPLEPLOT page.

Availability Section 1, released before version 2-5.

Arguments

`NHORIZ`	`INTEGER` expression	Number of pictures to be placed horizontally
`NVERT`	`INTEGER` expression	Number of pictures to be placed vertically

SUBROUTINE KTREAL (UNITS, IVAL1, IVAL2, RVAL, VALUE)

Name

`KTREAL` – to convert *time-date* value from external *time-date* triple to internal `REAL` form.

Availability Section plus, released version 2-12.

Arguments

`UNITS`	`CHARACTER*1`	`'T'`ime, `'M'`onths or `'W'`eeks
`IVAL1`	`INTEGER` expression	*Hours* or *years*
`IVAL2`	`INTEGER` expression	*Minutes*, *months* or *week number*
`RVAL`	`REAL` expression	*Seconds*, *date* or *day of week*
`VALUE`	`REAL` variable	To receive encoded *time-date* value

`UNITS`	`IVAL1`	`IVAL2`	`RVAL`
`'T'`ime	Hours	Minutes	Seconds
`'M'`onths	Year	Month	Date
`'W'`eeks	Year	Week no.	Day of week

SUBROUTINE LABJST (VJUST, HJUST)

Name

`LABJST` – to specify the justification of labels drawn by `CP7LB`, `CP7LBM`, `CP7XC` or `CP7YC`.

Availability Section 4, released version 2-6.

Arguments

`VJUST`	`CHARACTER*1`	Vertical justification of label
`HJUST`	`CHARACTER*1`	Horizontal justification of label

`VJUST`	*Vertical justification*
`'D'`	Default
`'T'`	Top of letters
`'C'`	Halfway between `'T'` and `'B'`
`'B'`	Bottom of letters (not including descenders)

`HJUST`	*Horizontal justification*
`'D'`	Default
`'L'`	At the left (beginning) of the label
`'C'`	Halfway between `'L'` and `'R'`
`'R'`	At the right (end) of the label
`'P'`	Preceding the label
`'F'`	Following the label

SUBROUTINE LIMEXC (DARR, NARR, VARMIN, VARMAX)

Name

`LIMEXC` – to find the minimum and maximum values in a `REAL` array.

Availability Section 1, released before version 2-5.

Arguments

`DARR`	`REAL` array	Data values
`NARR`	`INTEGER` expression	Number of elements of `DARR` to be examined
`VARMIN`	`REAL` variable	To receive minimum value
`VARMAX`	`REAL` variable	To receive maximum value

SUBROUTINE MARGIN (CMS)

Name

`MARGIN` – to specify the overall size of the margin around individual pictures.

Availability Section 1, released before version 2-5.

Argument

`CMS`	`REAL` expression	Width of margin (in cms)

SUBROUTINE PAGMRG (CMS, RCMS, BCMS, TCMS)

Name

`PAGMRG` – to specify the size and distribution of peripheral margins.

Availability Section 4, released version 2-11.

Arguments

`CMS`	`REAL` expression	Width of left margin (in cms)
`RCMS`	`REAL` expression	Width of right margin (in cms)
`BCMS`	`REAL` expression	Width of bottom margin (in cms)
`TCMS`	`REAL` expression	Width of top margin (in cms)

SUBROUTINE SHDEBX (X1, Y1, X2, Y2, ISHADE)

Name

`SHDEBX` – to draw a shaded box.

Availability Section 4, released version 2-8.

Arguments

`X1`, `Y1`, `X2`, `Y2`	`REAL` expressions	Coordinates of opposite corners of the box, specified in units of the plotting scales
`ISHADE`	`INTEGER` expression	Shading pattern number

SUBROUTINE SQSHAD (IARR, NARR)

Name

`SQSHAD` – to specify a sequence of shading patterns.

Availability Section 4, released version 2-12.

Arguments

`IARR`	`INTEGER` array	Pattern numbers for each shaded area
`NARR`	`INTEGER` expression	Number of elements in `IARR` (1–32)

`IARR(`*i*`)`	*Shading pattern*
−1	an empty area
0	solid fill with background colour
1, 2, 3 . . .	hardware/software patterns

SUBROUTINE TITLE7 (VCHAR, HCHAR, CAP)
SUBROUTINE TITLE (VCHAR, HCHAR, CAP, NCAP)

Name

`TITLE7` – to draw a text string as a title to the picture, group or page.

Availability Section 1, released before version 2-5.

Arguments

`VCHAR`	`CHARACTER*1`	Vertical position of title
`HCHAR`	`CHARACTER*1`	Horizontal position of title
`CAP`	`STRING` expression	Caption
`NCAP`	`INTEGER` expression	Number of characters in `CAP`

SUBROUTINE ZZORDR (XARR, YARR, NPTS, I2ARR, N2ARR, NVAR, ISIZE)

Name

`ZZORDR` – to reconfigure (x, y) coordinates into triangular elements and an array of neighbours.

Availability Section 2, released version 2-5.

Arguments

`XARR`, `YARR`	`REAL` arrays	(x, y) coordinates of data values
`NPTS`	`INTEGER` expression	Number of data points
`I2ARR`	`INTEGER` 2-D array	To receive data element structure, `I2ARR(3,ISIZE)`
`N2ARR`	`INTEGER` 2-D array	To receive neighbour array, `N2ARR(3,ISIZE)`
`NVAR`	`INTEGER` variable	To receive number of elements
`ISIZE`	`INTEGER` expression	second dimension of `I2ARR` and `N2ARR`

C. Changes to SIMPLEPLOT

This manual is based on *SIMPLEPLOT ViSualization* Version 2-15. The following changes have been made since the previous edition of the manual.

C.1 New subroutines

The following subroutines were introduced at 2-15:

`VSCIM`[7]	switch the method of interpolating colour sequences
`VSCGS`[7]	set the palette for a range of colour indices to a grey scale
`VSEDGI`[7]	specify the offset and increment for edging of gridded data
`VSEQZ`[7]	specify the equally-spaced z values for gridded 3-D data
`VSIS`[7]	draw the light-sourced IsoSurface of a specified data level
`VSISND`[7]	specify whether IsoSurfaces should exclude six end planes
`VSISU`[7]	draw contours of one 3-D array on the IsoSurface of another
`VSLSLD`[7]	specify the direction of a light source
`VSLSSM`[7]	specify the shading method for light-sourcing
`VSPGFE`[7]	fill an edge-flagged planar polygon
`VSPGTE`[7]	tint an edge-flagged planar polygon

D. Graphic Details

This appendix illustrates the graphical details of SIMPLEPLOT.

D.1 Marker symbols

Most SIMPLEPLOT subroutines use symbols from those listed in Figure D.2.

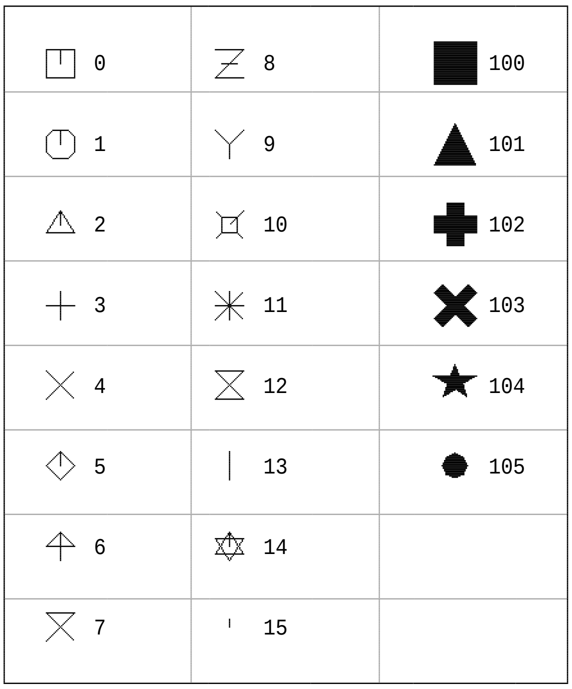


Figure D.1 *SIMPLEPLOT ViSualization* Polymarker symbols

The `VSPM*` subroutines for drawing polymarkers use the symbols listed in Figure D.1.

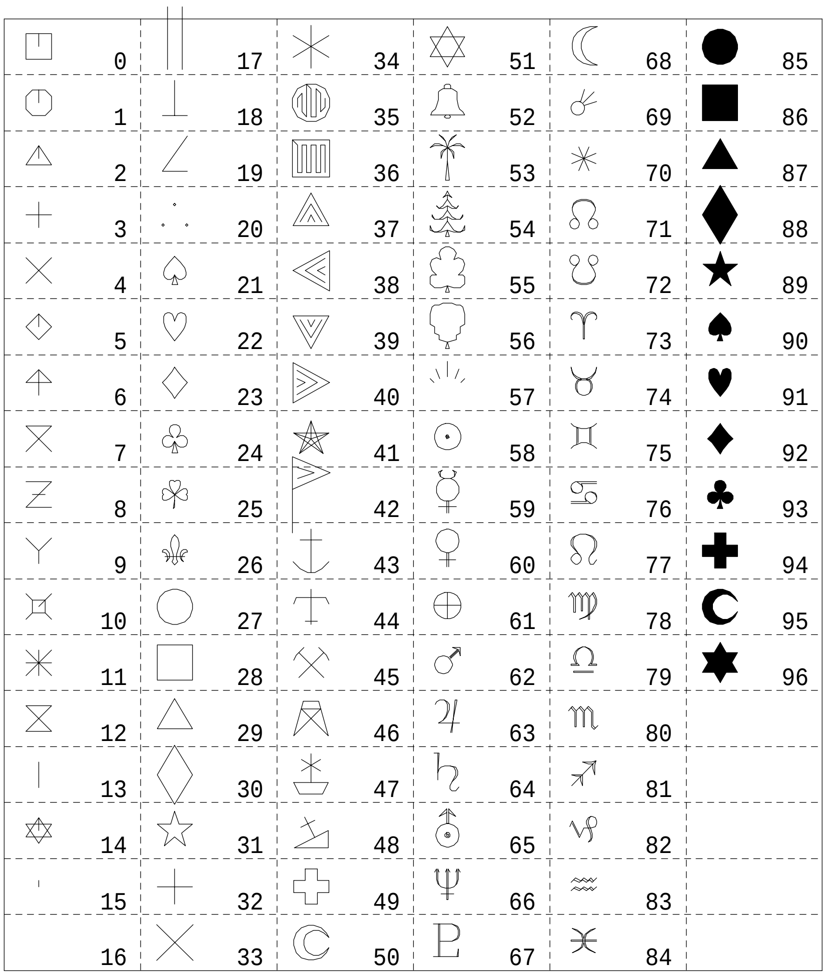


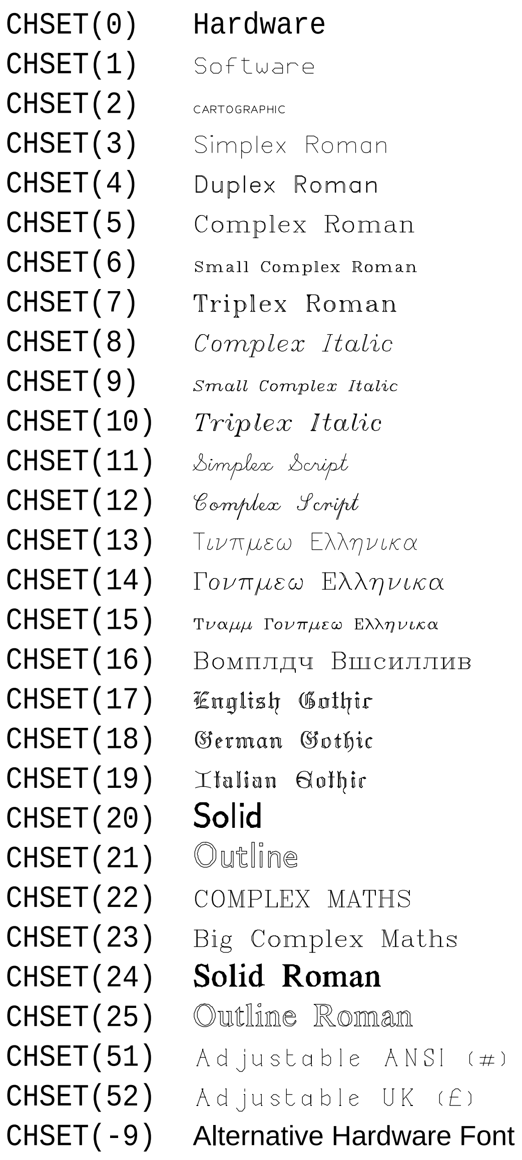
Figure D.2 SIMPLEPLOT marker symbols

D.2 Fonts

By default, SIMPLEPLOT uses the most appropriate hardware characters available on a graphics device to write text. In addition to hardware text, a set of simple software characters, proportionally spaced fonts (Hershey characters) and an adjustable fixed width font are available. Please note:

- *Hardware* fonts differ between graphics devices, therefore lettering which fits comfortably on one device may be smaller or larger on another.
- The *simple software* font is designed always to be clearly readable and may appear relatively larger on some low resolution graphics devices.
- Other software fonts are drawn independently of the resolution of the graphics device and may be illegible on some devices.

Figure D.3 illustrates the character sets available. CHSELECT is a utility program distributed with SIMPLEPLOT version 2-13 which can be used to generate font tables for any of the character sets.



CHSET(0)	Hardware
CHSET(1)	Software
CHSET(2)	CARTOGRAPHIC
CHSET(3)	Simplex Roman
CHSET(4)	Duplex Roman
CHSET(5)	Complex Roman
CHSET(6)	Small Complex Roman
CHSET(7)	Triplex Roman
CHSET(8)	*Complex Italic*
CHSET(9)	*Small Complex Italic*
CHSET(10)	*Triplex Italic*
CHSET(11)	*Simplex Script*
CHSET(12)	*Complex Script*
CHSET(13)	Τινπμεω Ελληνικα
CHSET(14)	Γονπμεω Ελληνικα
CHSET(15)	Τναμμ Γονπμεω Ελληνικα
CHSET(16)	Вомплдч Вшсиллив
CHSET(17)	English Gothic
CHSET(18)	German Gothic
CHSET(19)	Italian Gothic
CHSET(20)	Solid
CHSET(21)	Outline
CHSET(22)	COMPLEX MATHS
CHSET(23)	Big Complex Maths
CHSET(24)	**Solid Roman**
CHSET(25)	Outline Roman
CHSET(51)	Adjustable ANSI (#)
CHSET(52)	Adjustable UK (£)
CHSET(-9)	Alternative Hardware Font

Figure D.3 SIMPLEPLOT character sets

G. Glossary

Affine transformation: Changing the position of a point or object by *Magnification*, *Rotation* or *Translation*.

Bounding Sphere: The Sphere containing all points within the *Limiting Box* at all possible rotations. Its radius is the distance between the *Origin* and any corner of the *Limiting Box*.

Depth: The distance from an (x, y, z) point to the *Viewing Plane*. By default, all points within the *Limiting Box* have a depth between $-\sqrt{3}$ and $\sqrt{3}$.

Points between the *Viewing Position* and the *Viewing Plane* have negative depths. Points the other side of the *Viewing Plane* have positive depths.

Gamma correction: A function applied to colours to correct for the nonlinear response of the eye to linear changes colour intensity values.

Horizontal angle: The angle from which an object is viewed in the horizontal plane. Default is 30.0. Also known as θ (Theta).

IsoSurface: A 3-dimensional surface connecting points of equal value.

Limiting Box: The 3-dimensional box containing all the points which are plottable from the current *Viewing Position*. Specified by its corners (`XMIN`, `YMIN`, `ZMIN`) and (`XMAX`, `YMAX`, `ZMAX`).

Line of Sight: Line passing through the *Viewing Position* and the *Origin*. When *Perspective* is not applied, objects are viewed from infinity down the *Line of Sight*, at an angle (θ, ϕ).

Magnification: Magnifying a point or object by a given factor along an axis. Sometimes known as Scaling.

Origin: Point at the centre of the *Limiting Box*. The Origin is the point on the *Viewing Plane* which is closest to the *Viewing Position*.

Perspective: Method of viewing an object so that areas which are nearer to the *Viewing Position* appear larger than equivalent areas which are further away.

Phi (ϕ): see *Vertical Angle*.

Radius: Distance from the *Viewing Position* to the *Viewing Plane* in terms of the underlying scales.

Rotation: Moving a point or object around an axis or point so that its distance from that axis is preserved throughout.

Scaling: See *Magnification*.

Scaling Mode: Method of determining the relationship between the *Limiting Box* and the underlying 2-D scales.

Theta (θ): see *Horizontal Angle*.

Translation: Moving a point or object a given distance along an axis.

Vertical angle: The angle from which an object is viewed in the vertical plane. Default is 15.0. Also known as ϕ (Phi).

Viewing Plane: The plane to which *perspective* is applied. It is perpendicular to the *Line of Sight* and contains the *Origin*.

Viewing Position: The point from which the object is viewed. Expressed in terms of 2 angles, θ and ϕ. If *perspective* is applied, it is in terms of $(Radius, \theta, \phi)$.

The Viewing Position can also be expressed as (x, y, z) in terms of the underlying 3-dimensional coordinates.

T. Affine Transformation

Affine transformation entails changing the position, shape and size of an object, with all straight lines remaining straight. All such changes can be expressed as a combination of *Translation*, *Magnification* and *Rotation*.

The basic effects of these transformations can be explained with reference to the simpler 2-dimensional case before extending the same principles to 3-dimensions.

T.1 Translation

Translation]

Translation (subroutine `VSTRAN`) involves moving a point along an axis. Take the point A (XA, YA) on Figure T.1. Translation in x means moving it a fixed distance along the x-axis.

Translating the point a distance of XT in x moves it to BT $(XA + XT, YA)$. A similar translation of YT along the y-axis moves it to point CT $(XA + XT, YA + YT)$.

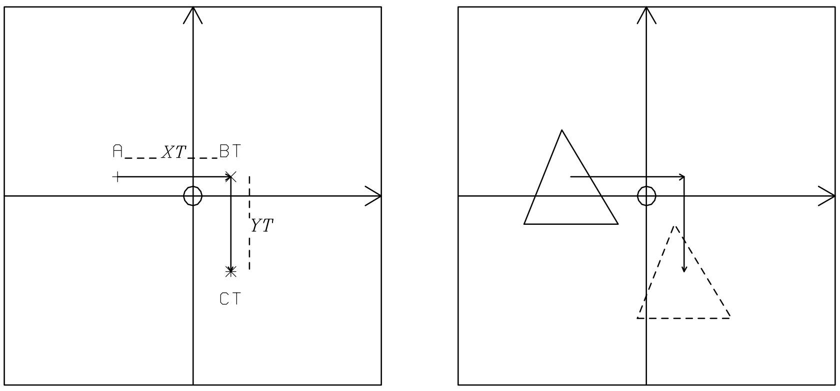


Figure T.1 Translation along x and y in 2 dimensions

To apply these translations to objects, every point on the object is translated by a given distance along each axis. The dotted triangle in Figure T.1 shows the effect of translating an object by (XT, YT).

Translation can take place along a number of axes, or just one. When translation is performed over more than one axis, the result is independent of the order in which axes are selected.

T.2 Magnification

Magnification]

Magnification (subroutine `VSMAG`) is a similar process to Translation, except a multiplying factor is applied to the coordinate, rather than simple addition.

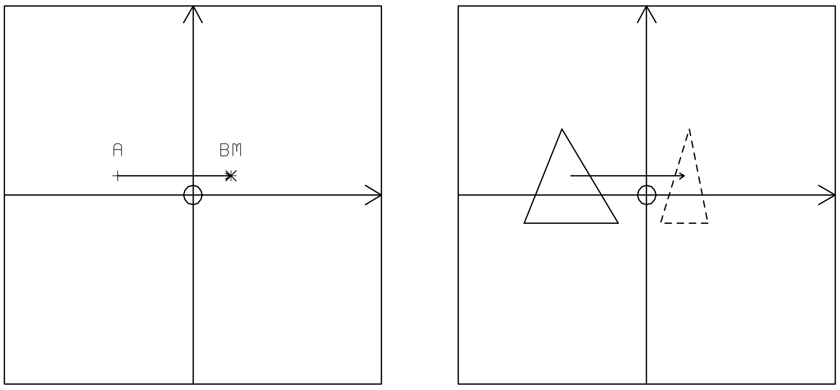


Figure T.2 Magnification along x

In Figure T.2, point A is magnified by a factor of XM along the x-axis, moving to BM $(XA \times XM, YA)$.

In this example, the magnification factor is negative, so the shape is reflected across the appropriate axis. The absolute value of the magnification factor is less than 1.0, so the shape is contracted. Unequal magnification factors on different axes change the aspect ratio of the shape.

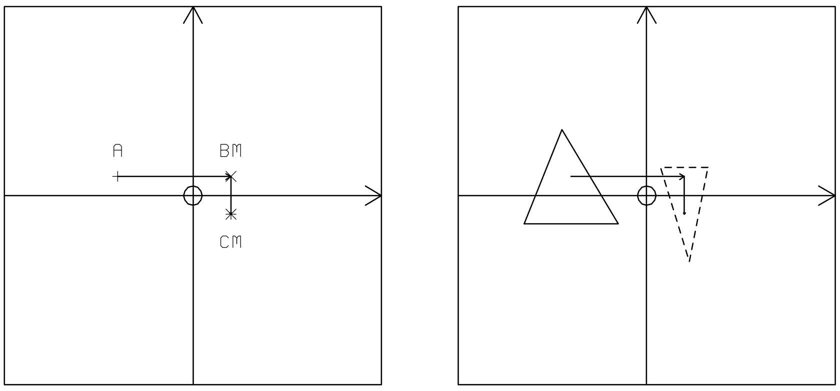


Figure T.3 Magnification along x and y

Magnification can take place along a number of axes, or just one. When magnification is performed over more than one axis, the result is independent of the order in which axes are selected.

In Figure T.3, an additional magnification factor of -1.0 is applied along the y-axis. This means that the triangle is inverted.

T.3 Translation and Magnification together

Although the order of combined Translations or combined Magnifications does not affect the result, when Translation is combined with Magnification, the order is significant.

For example, when the point (XA, YA) is translated by (XT, YT), then magnified by (XM, YM), it is moved to $(XM \times (XA + XT), YM \times (YA + YT))$, point CT' in Figure T.4. But when the same point is magnified by (XM, YM) before it is translated by (XT, YT), it is moved to $((XA \times XM) + XT, (YA \times YM) + YT)$, point CM' in Figure T.4.

The second picture in Figure T.4 shows the equivalent effect on objects.

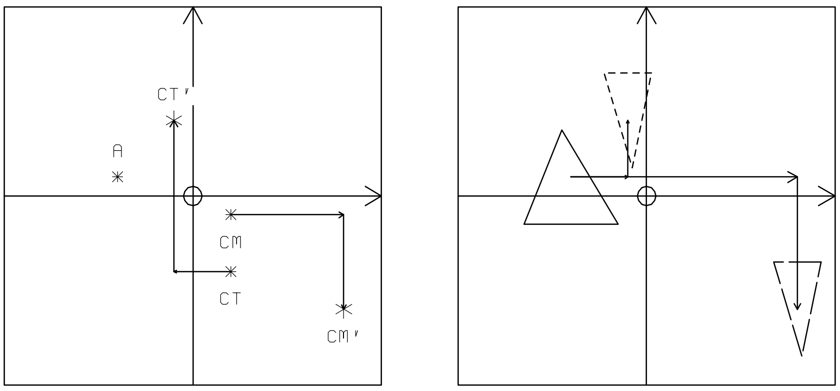


Figure T.4 Translation and magnification along x and y

This simple example illustrates the need to be sure about the order in which points are moved. This becomes particularly important when complex transformations, such as rotation, are applied to the object.

T.4 Rotation

Rotation]

Translation and Magnification are performed *along* an axis. 3-Dimensional Rotation is performed *around* an axis. In the simpler 2-dimensional case, rotation takes place around a point, the Origin (0,0).

Figure T.5 shows a simple 2-dimensional plane. A is an arbitrary point with coordinates (x, y). If the axes cross at (0,0), the distance from A to the Origin is $\sqrt{x^2 + y^2}$.

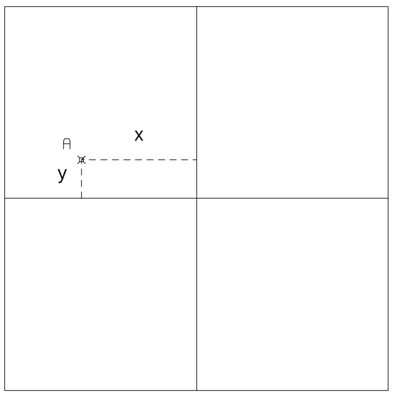


Figure T.5 Point A in a 2-dimensional plane

When a point is rotated, distance from the Origin is maintained. This means that the rotation path of A defines a circle of radius $\sqrt{x^2 + y^2}$ (Figure T.6).

If A is rotated by an anti-clockwise angle θ, the position of the new point A' can be easily calculated. If this rotation is accompanied by Translation and Magnification, the order in which these Transformations are applied is significant (see above).

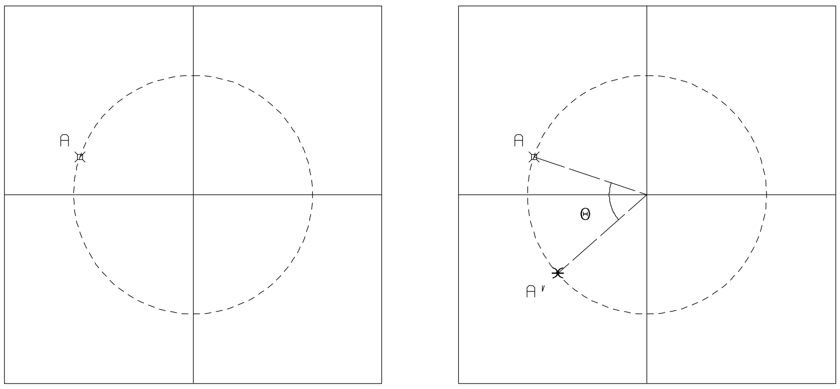


Figure T.6 Rotation path for point A around the z-axis

T.5 Three-dimensional Translation and Magnification

Translation and Magnification in 3 dimensions use the same principles as in 2 dimensions. A point (x, y, z) can be Translated to $(x + XT, y + YT, z + ZT)$, or magnified to $(x \times XM, y \times YM, z \times ZM)$. Translation can take place along 1 or more axes, and the order in which these operations are performed is not significant.

The first picture in Figure T.7 shows a cube which has been Translated along the x-axis. The second picture shows the Transformed cube Translated in y and z. Translation does not change the shape of the cube.

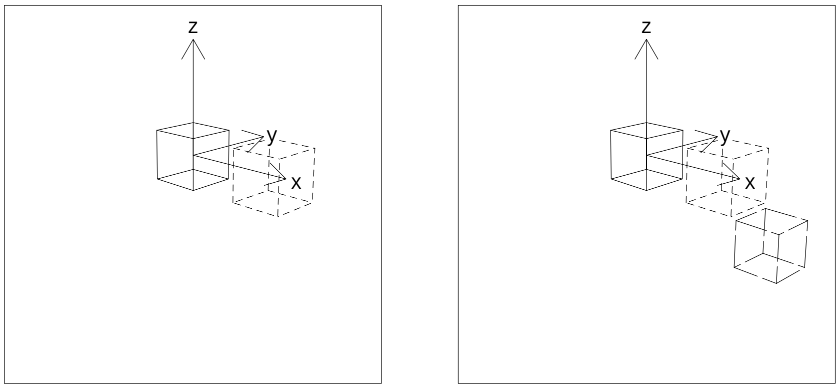


Figure T.7 Cube translated in x, y and z

Figure T.8 shows the same cube Magnified by different amounts along each axis. Note that, just as a magnified rectangle changed its aspect ratio, the new cuboid has quite different proportions.

T.6 Three-dimensional Rotation

All 2-dimensional rotation takes place around a point, usually the origin. 3-d rotation is around an axis, and differs according to which axis is used.

The first picture in Figure T.9 shows the path of a point which is rotated around the y-axis. The distance from the point to the axis remains the same. The second picture shows the same point rotated around z.

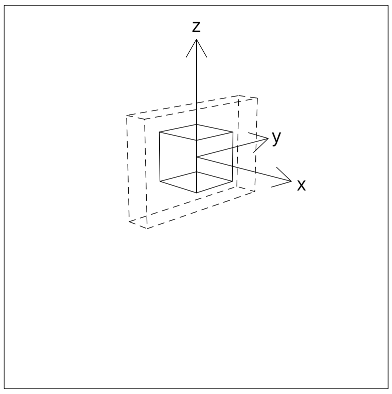


Figure T.8 Cube magnified in x, y and z

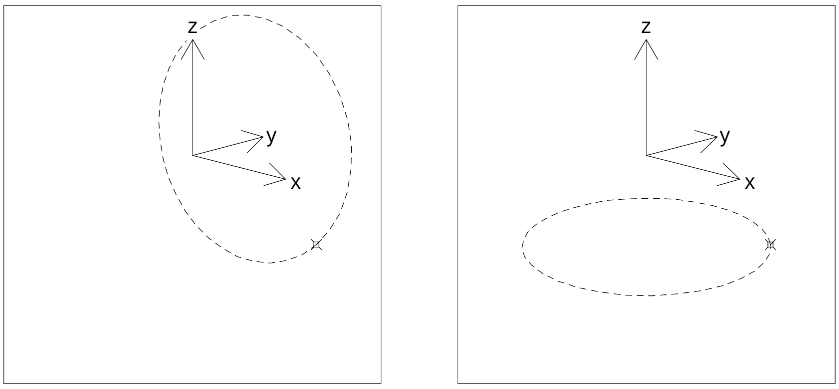


Figure T.9 Point rotated around y and z

The same principle can be extended to objects. Figure T.10 shows cuboids which have been rotated around the y and z axis. Figure T.11 shows the same cuboid rotated around x, y and z.

The order in which Rotation takes place about different axes is significant. Rotating an Object 10° in x then 20° in y is not the same as rotating 20° in y then 10° in x. This is why SIMPLEPLOT Visualization software only allows rotation to take place around any one axis at a time.

Any cube can be Translated, Magnified and Rotated so that it occupies any space at any angle. However, x, y and z axes remain perpendicular. This means that, although the shape may change, it will always be some sort of cuboid.

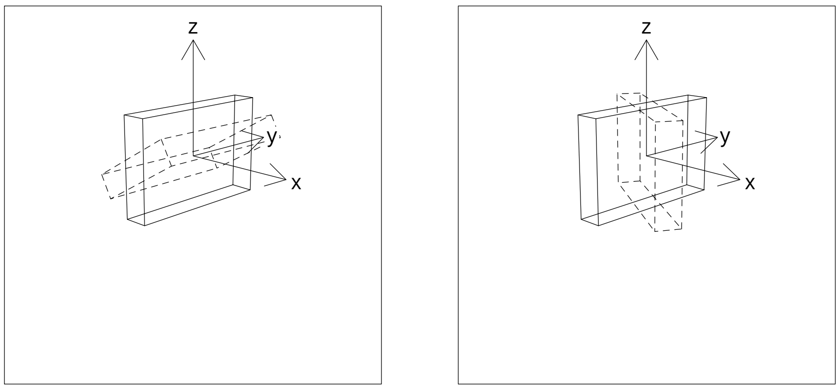


Figure T.10 Cuboid rotated around y

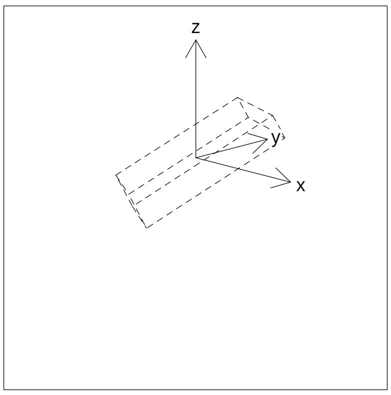


Figure T.11 Cuboid rotated around x, y and z

Index