New Dislin Features since Version 10.0 to 11.0

This article describes new features and options of Dislin which are added to the software since version 10.0 to version 11.0.

Chapter 2: Basic Concepts and Conventions

Programming in C++

Some Dislin distributions contain additional C++ libraries for using Dislin from C++. All Dislin routines are implemented as methods of the class Dislin, so that the description of the routines in the Dislin manual is also valid for C++. Here is a short example of a Dislin C++ program:

```
#include <iostream>
#include "discpp.h"
main()
{  Dislin g;
   g.metafl ("cons");
   g.disini ();
   g.messag ("This is a test", 100, 100);
   g.disfin ();
}
```

Chapter 3: Introductory Routines

S Y M B O L

The following constants can be used for symbol numbers in C and Fortran 90/95 programs:

SYMBOL_SQUARE	0	SYMBOL_OCTAGONCROSS	13
SYMBOL_OCTAGON	1	SYMBOL_SQUARETRIANGLE	14
SYMBOL_TRIANGLE_UP	2	SYMBOL_CIRCLE	15
SYMBOL_PLUS	3	SYMBOL_SQUARE_FILLED	16
SYMBOL_CROSS	4	SYMBOL_OCTAGON_FILLED	17
SYMBOL_DIAMOND	5	SYMBOL_TRIANGLE_UP_FILLED	18
SYMBOL_TRIANGLE_DOWN	6	SYMBOL_DIAMOND_FILLED	19
SYMBOL_SQUARECROSS	7	SYMBOL_TRIANGLE_DOWN_FILLED	20
SYMBOL_STAR	8	SYMBOL_CIRCLE_FILLED	21
SYMBOL_DIAMONDPLUS	9	SYMBOL_DOT	21
SYMBOL_OCTAGONPLUS	10	SYMBOL_HALFCIRCLE	22
SYMBOL_DOUBLETRIANGLE	11	SYMBOL_HALFCIRCLE_FILLED	23
SYMBOL_SQUAREPLUS	12		

I N C F I L

The routine INCFIL includes a GKSLIN or CGM metafile created by DISLIN, or general PNG, BMP, GIF and TIFF files into a graphics.

The call is:	CALL INCFIL (CFIL)	level 1, 2, 3
or:	void incfil (char *cfil);	

CFIL — is a character string that contains the filename.

Additional notes:

- For including PNG, BMP, GIF or TIFF files, the output format must be a raster, PostScript or PDF format.
- The routine FILBOX (NX, NY, NW, NH) defines a rectangular area on the page where the file will be included. (NX, NY) are the plot coordinates of the upper left corner, (NW, NH) are the width and length of the box in plot coordinates. By default, the entire page will be used. If the file is a bitmap and the output format a raster format, the file will be included at the point (NX, NY) while NW and NH will be ignored by default. This means that images are copied 1:1 to the screen. With the option FILOPT ('ON', 'SCALE'), images will be scaled. If the output format is PostScript or PDF, the image file will be scaled into the box defined by the parameters NX, NY, NW and NH. Therefore, NW and NH should have the same ratio as the width and height of the image file.
- The routine FILWIN (NX, NY, NW, NH) defines a rectangle of the image that will be included instead of the full image. The parameters in FILWIN must be specified as pixels.

F I L S I Z

The routine FILSIZ returns the size on an image file.

The call is: CALL FILSIZ (CFIL, NWIDTH, NHEIGHT, IRET) level 0, 1, 2, 3

or: int filsiz (char *cfil, int *nwidth, int *mheight);

CFIL — is a character string that contains the filename.

NWIDTH — is the returned width of the image in pixel.

NHEIGHT — is the returned height of the image in pixel.

IRET — contains a returned status that can have the values:
1: BMP file, 2: GIF file, 3: TIFF file, 4: PNG file, 0: undefined format, -1: error.

F I L T Y P

The routine FILTYP returns the type of a file.

The call is: CALL FILTYP (CFIL, ITYP) level 0, 1, 2, 3

or: int filtyp (char *cfil);

CFIL — is a character string that contains the filename.

ITYP — contains a returned status that can have the values:
1: BMP file, 2: GIF file, 3: TIFF file, 4: PNG file, 5: Dislin Image, 6: GKSL, 7: CGM, 8: WMF, 9: HPGL, 10: PostScript, 11: PDF, 12: Aldus WMF, 0: unknown format, -1: error.

Chapter 5: Plotting Curves

L E G P A T

The additional constants SYMBOL_EMPTY, LINE_NONE and SHADING_NONE can be used instead of the value -1 in C and Fortran 90/95 programs for symbols, line styles and shading patterns.

L E G T Y P

Legend entries can be plotted in vertical or horizontal direction depending on the option in LEGTYP. The routine must be called before LEGINI.

The call is:	CALL LEGTYP (COPT)	level 1, 2, 3
or:	void legtyp (const char *copt);	
COPT	is a character string that can have the values 'VERT' and 'HORI'. Default: COPT = 'VERT'.	

L E G B G D

The routine LEGBGD sets the background colour of legends.

The call is:	CALL LEGBGD (NCLR)	level 1, 2, 3
or:	void legbgd (int nclr);	
NCLR	is a colour value. The default value -1 means that no background is plotted. Default: NCLR = -1.	

L E G S E L

The routine LEGSEL selects legend lines that are plotted by LEGEND.

The call is:	CALL LEGSEL (IRAY, N)	level 1, 2, 3
or:	void legsel (const int *iray, int n);	
IRAY	is an integer array that contains legend lines between 1 and NLIN, where NLIN is the parameter in LEGINI.	
N	is the number of elements in IRAY. If N = -1, all legend lines are plotted. Default: N = -1.	

Chapter 6: Parameter Setting Routines

D E L G L B

DELGLB frees space that is allocated by DISLIN for global parameters. You can call this routine after DISFIN or WGFIN if you don't want to use any other DISLIN routine after the call.

The call is:	CALL DELGLB	level 0
or:	void delglb (void);	

M E T A F L

The keywords 'GL' and 'IPE' are added to the list of output formats. 'GL' defines an OpenGL window for graphical output. 'IPE' defines an XML format, which can be interpreted by the graphics editor Ipe.

Additional notes:

- By default, DISLIN plots to the OpenGL back buffer, which is copied to the front buffer by the routines DISFIN and SENDBF. If the option X11MOD ('NOSTORE') is set, DISLIN plots directly to the front buffer (graphics window).
- Multiple windows and hardware fonts are not supported by OpenGL windows.

F I L O P T

The routine FILOPT modifies rules for creating file version names, or or sets options for including files with INCFIL.

The call is:	CALL FILOPT (COPT, CKEY)	level 0, 1, 2, 3
or:	void filopt (const char *copt, const char *ckey);	
CKEY	is a character string that can have the values 'SEPARATOR', 'NUMBER', 'DIGITS' and 'SCALE'. The keywort 'SCALE' enables or disables scaling of images files imported by INCFIL. COPT can have the values 'ON' and 'OFF'. The default value is 'OFF'.	

D I S E N V

This routine sets the DISLIN environment within a program. If the DISLIN environment is already defined outside of the program, a call to DISENV has no affect.

The call is:	CALL DISENV (CPATH)	level 0, 1, 2, 3
or:	void disenv (const char *cpath);	
CPATH	is a character string that contains the path to the DISLIN installation directory.	

W I N J U S

WINJUS is an alternative routine to WINDOW for setting the position of the graphics window.

The call is:	CALL WINJUS (CJUS)	level 0, 1, 2, 3
or:	void winjus (const char *cjus);	
CJUS	is a character string that can have the values	
= 'CENTER'	means the center of the screen.	
= 'RBOTTOM'	means the lower right corner.	
= 'RTOP'	means the upper right corner.	
= 'LTOP'	means the upper left corner.	
= 'LBOTTOM'	means the lower left corner.	
	Default: CJUS = 'RBOTTOM'.	

W I N T Y P

The routine WINTYP defines the type of the graphics window. A graphics window with frames and a title bar can be used, or a window without any decorations.

The call is:	CALL WINTYP (CTYP)	level 0
or:	void wintyp (const char *ctyp);	
CJUS	is a character string that can have the values	
= 'STAND'	means a window with frames and a title bar.	
= 'POPUP'	means a window without any decorations.	
	Default: CTYP = 'STAND'.	

W I N I C O

The routine WINICO loads an icon from a file that is displayed in the title bar of the graphics window (only Windows).

The call is: CALL WINICO (CFIL) level 1, 2, 3

or: void winico (const char *cfil);

CFIL is a filename containing the icon. The format of the file must be a Windows .ico format.

Default: a standard icon is used.

W I N C B K

The routine WINCBK defines a user written callback routine which is called in DISFIN if the size of the graphics window is changed. The event loop is terminated in DISFIN before the callback routine is called.

The call is: CALL WINCBK (ROUTINE, 'SIZE') level 0, 1, 2, 3

or: void wincbk (routine, "SIZE");

ROUTINE is the name of a routine defined by the user. In Fortran, the routine must be declared as EXTERNAL. The parameters passed to ROUTINE are (ID, NX, NY, NW, NH), where ID is the ID of the window (see OPNWIN), NX, NY are the coordinates of the upper left corner and NW and NH the width and height of the changed window in pixels.

L A B E L S

The new keywords "MAPNDG' and 'XEXP' are added to the list of keywords. 'MAPNDG' is added for plotting geographical labels without a degree symbol. 'XEXP' has the same meaning as 'EXP', but a times symbol is used as an operator instead of an asterisk.

A X S E R S

AXSERS erases the contents of an axis system.

The call is: CALL AXSERS level 2, 3

or: void axsers (void);

C O L O R

New keywords 'GRAY' and 'HALF' are available for COLOR. 'HALF' sets a new foreground colour with the half intensity of the current foreground colour.

T X T J U S

Addtional keywords 'TOP', 'BOTTOM' and 'MIDDLE' are added for vertical alignment.

T X T B G D

TXTBGD defines a background colour for text and numbers.

The call is: CALL TXTBGD (NCLR) level 1, 2, 3

or: void txtbgd (int nclr);

NCLR — is a colour number. The default value -1 means that no background is plotted. The margin between background border and text is (LINESP-1) * NHCHAR, where LINESP is the value in LINESP.

Default: NCLR = -1.

M A R K E R

The symbol value -1 is now allowed in MARKER and means that the symbol is not plotted in routines such as CURVE and ERRBAR.

G A P S I Z

GAPSIZ defines a data gap used in the routine CURVE. This routine is an extension to GAPCRV and can be used for X- and Y-coordinates.

The call is: CALL GAPSIZ (XGAP, CAX) level 1, 2, 3

or: void gapsiz (float xgap, const char *cax);

XGAP — is the gap value.

CAX — is a character string that defines the axes. CAX can have the values 'X', 'Y', 'XY' and 'RESET'.

N A N C R V

The routine NANCRV can be used to enable the checking for undefined values (NaN) in curves. NaN values will be plotted as gaps and their count is reported in the DISLIN protocol.

The call is: CALL NANCRV (CMODE) level 1, 2, 3

or: void nancrv (const char *cmode);

CMODE — is a character string that can have the values 'ON' and 'OFF'. Default: CMODE = 'OFF'.

L I N T Y P

The following constants can be used for line styles in C and Fortran 90/95 programs:

LINE_SOLID	0	LINE_DASHM	4
LINE_DOT	1	LINE_DASHL	5
LINE_DASH	2	LINE_DOTL	6
LINE_CHNDOT	3		

L I N C L R

The routine LINCLR defines colour values for the pen-downs in line styles. The colours are ignored for solid lines.

The call is: CALL LINCLR (NRAY, N) level 1, 2, 3

or: void linclr (const int *nray, int n);

NRAY — is an array of colour values.

N — is the number of elements in NRAY (N $\leq$ 10). The default value N = 0 disables colours for line styles.

L I N M O D

The routine LINMOD enables anti-aliased lines in image formats such as PNG, BMP and TIFF. True colour mode is required for anti-aliased lines (see IMGFMT).

The call is:	CALL LINMOD (CMOD, CKEY)	level 1, 2, 3
or:	void linmod (const char *cmod, const char *ckey);	
CMOD	is a character string that can contain the modes 'ON' and 'OFF'.	
CKEY	is a character string that can have the value 'SMOOTH'. Default: ('OFF', 'SMOOTH').	

S H D P A T

The following constants can be used for shading patterns in C and Fortran 90/95 programs:

SHADING_EMPTY	0	SHADING_GRID_BOLD	14
SHADING_LINES	1	SHADING_FILLED	16
SHADING_LINES_BOLD	4	SHADING_DOTS	17
SHADING_GRID	10		

S H D F A C

The routine SHDFAC sets a scaling factor for the distance of scan lines for software shading.

The call is:	CALL SHDFAC (XFAC)	level 1, 2, 3
or:	void shdfac (float xfac);	
XFAC	is the scaling factor (> 0.0). Default: XFAC = 1.0	

T T F O N T

TTFONT loads a Windows TrueType font. The characters of the font can be used for all Dislin output devices. By default, only the outlines of the characters are plottted. After the statement CALL SHDCHA characters will be shaded.

The call is:	CALL TTFONT (CFILE)	level 1, 2, 3
or:	void ttfont (const char *cfile);	
CFILE	is a character string that contains the filename of a TrueType font. If the filename does not contain a full directory path, the file is searched in the current directory, in the Windows fonts and in the Dislin fonts directory. For Linux, the font is also searched in the directory '/usr/X11R6/lib/X11/fonts/truetype/'.	

Chapter 7: Parameter Requesting Routines

G E T S C M

This routine informs if automatic scaling is used in a GRAF routine. A returned value of 1 means automatic scaling, otherwise 0 is returned. (see SETSCL).

The call is:	CALL GETSCM (IX, IY, IZ)	level 1, 2, 3
or:	void getscm (int *ix, int *iy, int *iz);	

Chapter 8: Elementary Plot Routines

V E C T O R

Two additional arrow forms are added to VECTOR. The form y = 4 means a sharp and filled arrow head, y = 5 means a sharp unfilled arrow head, where the digit y is part of the four digit number 'wxyz', that defines the appearance of arrows. The new options should work for all routines, where an array option can be specified.

T R I F L C

The routine TRIFLC plots solid filled triangles with interpolated colours.

The call is:	CALL TRIFLC (XRAY, YRAY, ICRAY, N)	level 1, 2, 3
or:	void triflc (const float *xray, const float *yray, const int *icray, int n);	
XRAY	are floating point arrays containing triangle corners.	
ICRAY	are the colour values of the triangle corners.	
N	is the number of points in the arrays above. N should be a multiple of three. You can increase performance by passing multiple triangles to TRIFLC instead of calling TRIFLC several times.	

W I N D B R

The routine WINDBR for plotting wind speed symbols (wind barbs) is so modified that wind flags are plotted at the opposite site if the symbol length is specified as a negative number.

Chapter 9: Utility Routines

H I D W I N

The routine HIDWIN defines whether a graphics window is visible or not.

The call is:	CALL HIDWIN (ID, CMOD)	level 1, 2, 3
or:	void hidwin (int id, const char *cmod);	
ID	is the window number between 1 and 8.	
CMOD	is a character string that can have the values	
= 'YES'	the window is hided and not visible.	
= 'NO'	the windows is showed and visible.	
	Default: CMOD = 'NO'.	

C S R L I N

The routine CSRLIN is similar to CSRREC and returns the end points of a line created with mouse button 1.

The call is:	CALL CSRLIN (NX1, NY1, NX2, NY2)	level 1, 2, 3
or:	void csrlin (int *nx1, int *ny1, int *nx2, int *ny2);	

NX1, NY1, NX2, NY2 are the returned coordinates of the line end points.

E X P I M G

The routine EXPIMG copies an image from memory to a file.

The call is:	CALL EXPIMG (CFIL, COPT)	level 1, 2, 3
or:	void expimg (const char *cfil, const char *copt);	
CFIL	is the name of the output file. A new file version will be created for existing files (see FILMOD).	
COPT	defines the file format and can have the values 'PS', 'PDF', 'PNG', 'GIF', 'TIFF', 'PPM' and 'BMP'.	
Additional note:	For the options 'PNG', 'GIF', 'TIFF', 'PPM' and 'BMP', EXPIMG has the same meaning as the routines RPNG, RGIF, RTIFF, RPPM, and RBMP.	

L D I M G

The routine LDIMG loads an PNG, BMP, GIF or TIFF image from a file into an array. RapidEye satellite TIFF images are also supported by LDIMG.

The call is:	CALL LDIMG (CFIL, IRAY, NMAX, NC, N)	level 0, 1, 2, 3
or:	int ldimg (const char *cfil, unsigned short *iray, int nmax, int nc);	
CFIL	is a character string that contains the filename.	
IRAY	is an INTEGER*2 array containing the image data after the call to LDIMG.	
NMAX	is the number of elements in IRAY. If NMAX = 0, just the needed number of elements is returned in the variable N.	
NC	is the channel number. Normally, the red components are returned for NC = 1, the green values for NC = 2 and the blue values for NC = 3. RapidEye TIFF images contain 5 channels. If NC = 0, all channels are returned, stored after each other in IRAY. For NC = -1, the image is packed as byte values in IRAY. Three bytes contain the RGB values of a pixel.	
N	is the returned number of elements used in IRAY.	

F I T S O P N

The routine FITSOPN opens a FITS file for reading.

The call is:	CALL FITSOPN (CFILE, ISTAT)	level 0, 1, 2, 3
or:	int fitsopn (const char *cfile);	
CFILE	is a character string containing the file name.	
ISTAT	is the returned status (0: no errors).	

F I T S C L S

This routine closes a FITS file that was opened before with FITSOPN.

The call is:	CALL FITSCLS	level 0, 1, 2, 3

or: void fitscls (void);

F I T S H D U

The routine FITSHDU selects the FITS Header/Data Unit for the following FITS operations.

The call is: CALL FITSHDU (NHDU, IRET) level 0, 1, 2, 3

or: int fitshdu (int nhdu);

NHDU defines the current FITS HDU (≥ 1).

IRET is the returned type of the HDU:

0: Primary HDU
1: Image Extension
2: ASCII Table Extension
3: Binary Table Extension

IRET = -1 means that the HDU does not exist.

F I T S T Y P

FITSTYP returns the type of a key.

The call is: CALL FITSTYP (CKEY, ITYP) level 0, 1, 2, 3

or: int fitstyp (const char *ckey);

CKEY is a character string containing the key.

ITYP is the returned type. The possible types are:

0: Integer number
1: Float number
2: Null string
3: String
4: Logical True
5: Logical False
6: Complex Integer
7: Complex Float
8: Undefined

ITYP = -1 means that the key could not be found in the FITS file.

F I T S V A L

The routine FITSVAL returns the integer value of a key. Some of the required keys in FITS files are:

NAXIS the number of data axes

NAXISi the length of axis i, where $1 \leq i \leq n$

BITPIX the number of bits per data pixel.

The call is: CALL FITSVAL (CKEY, IVAL) level 0, 1, 2, 3

or: int fitsval (const char *ckey);

CKEY is a character string containing the key.

IVAL is the returned integer value.

F I T S F L T

FITSFLT returns the floatingpoint value of a key.

The call is:	CALL FITSFLT (CKEY, XVAL)	level 0, 1, 2, 3
or:	float fitsflt (const char *ckey);	
CKEY	is a character string containing the key.	
XVAL	is the returned floatingpoint value.	

F I T S S T R

FITSSTR returns the string value of a key.

The call is:	CALL FITSSTR (CKEY, CVAL, NMAX)	level 0, 1, 2, 3
or:	void fitsstr (const char *ckey, char *cval, int nmax);	
CKEY	is a character string containing the key.	
CVAL	is the returned string value.	
NMAX	is the maximal number of bytes that can be copied to CVAL.	

F I T S I M G

The routine FITSIMG copies the image data of a FITS file to a byte array.

The call is:	CALL FITSIMG (IRAY, NMAX, N)	level 0, 1, 2, 3
or:	int fitsimg (unsigned char *iray, int nmax);	
IRAY	is a byte array containing the returned image pixels.	
NMAX	defines how many bytes can be copied to IRAY. If NMAX = 0, the necessary number of bytes is returned in N without copying the image data.	
N	is the returned number of images bytes.	

Chapter 10: Business Graphics

F B A R S

FBARS plots financial bars for open, high, low and close prices. The bars are displayed as line bars or candlestick bars.

The call is:	CALL FBARS (XRAY, Y1RAY, Y2RAY, Y3RAY, Y4RAY, N)	level 2, 3
or:	void fbars (const float *xray, const float *y1ray, const float *y2ray, const float *y3ray, const float *y4ray, int n);	
XRAY	is an array of user coordinates defining the position of the bars on the X-axis.	
Y1RAY	is an array of user coordinates containing the open prices.	
Y2RAY	is an array of user coordinates containing the high prices.	
Y3RAY	is an array of user coordinates containing the low prices.	
Y4RAY	is an array of user coordinates containing the close prices.	

N	is the number of bars.	
Additional notes:	- The type of the financial bars can be selected with the routine BARTYP.	
	- BARCLR sets colours for finacial bars.	

B A R T Y P

The routine BARTYP defines vertical or horizontal bars, or the type of financial bars.

The call is:	CALL BARTYP (CTYP)	level 1, 2, 3
or:	void bartyp (const char *ctyp);	
CTYP	is a character string defining the bar type.	
= 'CANDLE'	defines candlestick bars for financial bars.	
= 'TICKS'	defines financial line bars with tick marks.	
	Default: CTYP = 'CANDLE'.	

B A R C L R

The routine BARCLR defines the colours of bars. Different colours can be defined for the sides of 3-D bars.

The call is:	CALL BARCLR (IC1, IC2, IC3)	level 1, 2, 3
or:	void barclr (int ic1, int ic2, int ic3);	
IC1, IC2, IC3	are colour values for the front, side and top planes of 3-D bars. The value -1 means that the corresponding plane is plotted with the current colour. For financial bars, IC1 is the colour of the line bars, IC2 the colour of the open ticks and IC3 the colour of the close ticks.	
	Default: (-1, -1, -1).	

Chapter 11: 3-D Colour Graphics

A U T R E S

With a call to AUTRES, the size of coloured rectangles will be automatically calculated by GRAF3 or CRVMAT.

The call is:	CALL AUTRES (IXDIM, IYDIM)	level 1, 2, 3
or:	void autres (int ixdim, int iydim);	
IXDIM, IYDIM	are the number of data points in the X- and Y-direction, or (0, 0). If IXDIM = 0 and IYDIM = 0, CRVMAT plots rectangles between neighbouring data points. This is useful for logarithmic axes, where the rectangles should have a different size. A negative value can be used for a logartithmic axis scaling, where the matrix in CRVMAT contains already a logarithmic grid.	

P O S B A R

The routine POSBAR sets the position of colour bars. By default, colour bars are plotted in a vertical direction near the right Y-axis of an axis system.

The call is:	CALL POSBAR (COPT)	level 1, 2, 3

	or:	void posbar (const char *copt);
COPT		is a character value defining the position of colour bars.
= 'FIXED'		means that the colour bar is plotted in a vertical direction near the right Y-axis.
= 'RIGHT'		has nearly the same meaning as the keyword 'FIXED', but the colour bar is automatically moved if labels and an axis title is plotted at the right Y-axis.
= 'TOP'		means that the colour bar is plotted above the axis system in a horizontal direction.
= 'BOTTOM'		means that the colour bar is plotted below the axis system in a horizontal direction.
= 'LEFT'		means that the colour bar is plotted on the left side of the axis system.

Default: COPT = 'FIXED'.

J U S B A R

JUSBAR defines alignment of colour bars.

The call is:	CALL KUSBAR (COPT)	level 1, 2, 3
or:	void jusbar (const char *copt);	
COPT	is a character value defining the alignment of colour bars.	
= 'START'	means that the colour bar is plotted at the beginning of the X- or Y-axis.	
= 'CENTER'	means that the colour bar is centred at the X- or Y-axis.	
= 'END'	means that the colour bar is justified at the end of the X- or Y-axis.	

Default: COPT = 'START'.

F R M B A R

The routine FRMBAR defines the thickness of frames around colour bars.

The call is:	CALL FRMBAR (N)	level 1, 2, 3
or:	void frmbar (int n);	
N	is the thickness in plot coordinates.	

Default: N = 0

S P C B A R

The routine SPCBAR defines the space between colour bars and axis systems.

The call is:	CALL SPCBAR (N)	level 1, 2, 3
or:	void spcbar (int n);	
N	is the space in plot coordinates.	

Default: N = 85

Chapter 12: 3-D Graphics

P R O J 3 D

The routine PROJ3D defines a perspective or orthographic projection.

The call is:	CALL PROJ3D (COPT)	level 1
or:	void proj3d (const char *copt);	
COPT	is a character string that can have the values 'PERSPECTIVE' and 'ORTHO'. The default value is COPT = 'PERSPECTIVE'.	

V S C L 3 D

For an orthographic view the size of the projected 3-D box can be scaled by a factor defined with VSCL3D.

The call is:	CALL VSCL3D (XFAC)	level 1, 2, 3
or:	void vscl3d (float xfac);	
XFAC	defines the scaling factor. The default value is XFAC = 1.0.	

C U R V 4 D

The routine CURV4D plots coloured 3-D symbols.

The call is:	CALL CURV4D (XRAY, YRAY, ZRAY, WRAY, N)	level 3
or:	void curv4d (const float *xray, const float *yray, const float *zray, const float *wray, int n);	
XRAY	is an array containing the X-coordinates of data points.	
YRAY	is an array containing the Y-coordinates of data points.	
ZRAY	is an array containing the Z-coordinates of data points.	
WRAY	is an array of dimension N containing intensities. The minimum and maximum of WRAY are used for the colour scaling.	
N	is the number of data points.	
Additional notes:	- The statement CALL ZSCALE (ZMIN, ZMAX) defines an alternate range for calculating colours.	
	- The used 3-D symbol can be selected with the routine MARKER. The symbol numbers corresponds to the numbers in SYMB3D.	

C O N S H D 3 D

The routine CONSHD3D plots a shaded surface from a matrix where colour values are connected with contours.

The call is:	CALL CONSHD3D (XRAY, IXDIM, YRAY, IYDIM, ZMAT, ZLVRAY, NLEV)	level 3
or:	void conshd3d (const float *xray, int ixdim, const float *yray, int iydim, const float *zmat, const float *zlvray, int nlev);	
XRAY, YRAY	are arrays containing the X- and Y-user coordinates.	
ZMAT	is a matrix with the dimension (IXDIM, IYDIM) containing the function values.	
IXDIM, IYDIM	are the dimensions of ZMAT, XRAY and YRAY (≥ 2).	
ZLVRAY	is an array containing the levels.	

NLEV	is the number of levels.
Additional note:	The user is referred to the notes on SURSHD and CONSHD.

S U R S H C

The routine SURSHC is a similar routine to SURSHD with an extra matrix which is used for calculating surface colours.

The call is:	CALL SURSHC (XRAY, IXDIM, YRAY, IYDIM, ZMAT, WMAT)	level 3
or:	void surshc (const float *xray, int ixdim, const float *yray, int iydim, const float *zmat, const float *wmat);	
XRAY, YRAY	are arrays containing the X- and Y-user coordinates.	
IXDIM, IYDIM	are the dimensions of ZMAT, WMAT, XRAY and YRAY (≥ 2).	
ZMAT	is a matrix with the dimension (IXDIM, IYDIM) containing the function values.	
WMAT	is a matrix with the dimension (IXDIM, IYDIM) which is used to calculate surface colours.	
Additional note:	The user is referred to the notes on SURSHD.	

T R 3 A X S

The routine TR3AXS defines a rotation about an arbitrary axis.

The call is:	CALL TR3AXS (X, Y, Z, A)	level 3
or:	void tr3axs (float x, float y, float z, float a);	
X, Y, Z	define the axis which goes from the origin to the point (X, Y, Z).	
A	is a rotation angle in degrees. Rotation is done in a counter-clockwise direction when looking from the point (X, Y, Z) toward the origin.	

P L Y I N I

The routine PLYINI initializes the output of polygons to a PLY file. The polygons are sent to the output device and to the PLY file.

The call is:	CALL PLYINI (COPT)	level 3
or:	void plyini (const char *copt);	
COPT	is a character string that defines the format of the PLY file. COPT can have the value 'STANDARD'.	

P L Y F I N

The routine PLYFIN terminates the output of polygons to a PLY file.

The call is:	CALL PLYFIN (CFIL, CSTR)	level 3
or:	void plyini (const char *cfil, const char *cstr);	
CFIL	is a character string that contains the name of the PLY file.	
CSTR	is a character string that is written as a comment to the PLY file.	
Additional note:	Backface culling should be disabled with the routine SHDMOD. Otherwise, polygons written to the PLY file depend on the viwepoint.	

Chapter 13: Geographical Projections and Plotting Maps

S H D M A P

The new options 'SEA', 'LSEA', 'ISEA', 'HSEA' and 'LAND' are added to SHDMAP. The keyword 'SEA' means that all oceans and lakes are shaded. For 'LSEA', 'ISEA' and 'HSEA' the GSHHS map coordinates 'gshhs_l.b', 'gshhs_i.b' and 'gshhs_h' are used for sea shading. 'LAND' means that all land is shaded without lakes. A solid shading is automatically used for this new options. With version 10.4.1 the options 'GSHL', 'GSHI' and 'GSHH' are added for using the GSHHS map coordinates for shading continents.

M A P I M G

The routine MAPIMG plots a PNG, TIFF, BMP or GIF raster image to an axis system. Some parameters which describe the location, scale and rotation of the map are passed to MAPIMG. The parameters have the same meaning as the attributes of the ESRI World File Format.

The call is:	CALL MAPIMG (CFIL, X1, X2, X3, X4, X5, X6)	level 2
or:	void mapimg (const char *cfil, float x1, float x2, float x3, float x4, float x5, float x6);	
CFIL	is a character string that contains the name of the image file.	
X1	is the pixel size in the X-direction in map units per pixel.	
X2	is the rotation about the Y-axis.	
X3	is the rotation about the X-axis.	
X4	is the pixel size in the Y-direction in map units per pixel. This value is normally a negative number.	
X5	is the X-coordinate of the centre of the upper left pixel.	
X6	is the Y-coordinate of the centre of the upper left pixel.	

P T 2 P O S

The routine PT2POS is the inverse routine to POS2PT and converts plot coordinates to map coordinates. The plot coordinates should be located in the current axis system.

The call is:	CALL PT2POS (XP, YP, XLONG, YLAT)	level 2
or:	void pos2pt (float xp, float yp, float *xlong, float *ylat);	
XP, YP	are the plot coordinates.	
XLONG, YLAT	are the map coordinates calculated by PT2POS.	

Chapter 15: Widget Routines

W G B O X, W G L I S, W G D L I S

The routines accept now the value 0 for the pre-selected element, which means that no element is pre-selected.

W G P O P B

The routine WGPOPB creates a popup menu in the menu bar of the main widget, or a popup submenu of a popup menu. WGPOPB uses an image as title instead of a character string.

The call is:	CALL WGPOPB (IP, IRAY, NW, NH, ID)
or:	int wgpopb (int ip, const unsigned char *iray, int nw, int nh);
IP	is the index of the parent widget.
IRAY	is a byte array containing the image. Each pixel must be represented as three bytes with the RGB values.
NW, NH	is the width and height of the image.
ID	is the returned widget index.

W G A P P B

The routine WGAPPB creates an image entry in a popup menu.

The call is:	CALL WGAPPB (IP, IRAY, NW, NH, ID)
or:	int wgappb (int ip, const unsigned char *iray, int nw, int nh);
IP	is the index of the parent widget.
IRAY	is a byte array containing the image. Each pixel must be represented as three bytes with the RGB values.
NW, NH	is the width and height of the image.
ID	is the returned widget index.

W G S E P

The routine WGSEP separates widgets by drawing horizontal or vertical lines, or menu items by drwing horizontal lines.

The call is:	CALL WGSEP (IP, ID)
or:	int wgsep (int ip);
IP	is the index of the parent widget.
ID	is the returned widget index.
Additional notes:	- WGSEP draws by default horizontal lines. Vertical lines can be defined with the routine SWGTYP.
	- Several line drawing styles can be selected with the routine SWGOPT.

W G I C O N

The routine WGICON creates a label widget with an icon as label.

The call is:	CALL WGICON (IP, CLAB, NW, NH, CFIL, ID)

or:	int wgicon (int ip, const char *clab, int nw, int nh, const char *cfil);
IP	is the index of the parent widget.
CLAB	is a character string that will be displayed if the mouse is moved over the label. This feature is only supported on Windows.
NW, NH	is the width and height of the icon. If NW = 0 and NH = 0, the height and width is calculated from the icon file.
CFIL	is a filename containing the icon.
ID	is the returned widget index.

W G I M G

The routine WGIMG creates a label widget with an image as label.

The call is:	CALL WGIMG (IP, CLAB, IRAY, NW, NH, ID)
or:	int wgimg (int ip, const char *clab, const unsigned char *iray, int nw, int nh);
IP	is the index of the parent widget.
CLAB	is a character string that will be displayed if the mouse is moved over the label. This feature is only supported on Windows.
IRAY	is a byte array containing the image. Each pixel must be represented as three bytes with the RGB values.
NW, NH	is the width and height of the image.
ID	is the returned widget index.

W G P I C O N

The routine WGPICON creates a push button widget with an icon as label.

The call is:	CALL WGPICON (IP, CLAB, NW, NH, CFIL, ID)
or:	int wgpicon (int ip, const char *clab, int nw, int nh, const char *cfil);
IP	is the index of the parent widget.
CLAB	is a character string that will be displayed if the mouse is moved over the label. This feature is only supported on Windows.
NW, NH	is the width and height of the icon. If NW = 0 and NH = 0, the height and width is calculated from the icon file.
CFIL	is a filename containing the icon.
ID	is the returned widget index.

W G P I M G

The routine WGPIMG creates a push button widget with an image as label.

The call is:	CALL WGPIMG (IP, CLAB, IRAY, NW, NH, ID)
or:	int wgpimg (int ip, const char *clab, const unsigned char *iray, int nw, int nh);
IP	is the index of the parent widget.
CLAB	is a character string that will be displayed if the mouse is moved over the label. This feature is only supported on Windows.

IRAY — is a byte array containing the image. Each pixel must be represented as three bytes with the RGB values.

NW, NH — is the width and height of the image.

ID — is the returned widget index.

S W G S C L

The routine SWGSCL changes the value of a scale widget, and the value of scrollbars in draw widgets.

The call is: CALL SWGSCL (ID, XVAL)

or: void swgscl (int id, float xval);

ID — is a widget ID of a scale or draw widget. If the widget ID is passed as it's negative value, the vertical scrollbar of a draw widget is changed. Otherwise, the horizontal scrollbar.

XVAL — is a floating point number containing the new value of the scrollbar.

S W G O P T

The routine SWGOPT sets widget options.

The call is: CALL SWGOPT (COPT, CKEY)

or: void swgopt (const char *copt, const char *ckey);

COPT — is a character string containing an option.

CKEY — is a character string containing a keyword:

= 'CODING' — defines the coding of strings in widgets. COPT can have the values 'ANSI' (default), 'UTF8', 'RUSSIAN' and 'GREEK'. This option is only available on Windows systems, but not for X11.

= 'DIALOG' — Dialog widgets created by the DWG routines described in paragraph 15.5 can have the topmost attribute, so that they are not overplotted by other windows. COPT can have the values 'STANDARD' and 'TOP'. This option is only available on Windows, not on X11 systems.

= 'SEPARATOR' — This option selects a line style for WGSEP. COPT can have the values 'STANDARD', 'SINGLE', 'DOUBLE', 'DASH' and 'DDASH'.

= 'SLIDER' — Specifies whether a label for the current slider value is displayed or not. COPT can have the values 'VALUE' (default) and 'NOVALUE'.

Defaults: ('ANSI', 'CODING'), ('STANDARD', 'DIALOG'), ('STANDARD', 'SEPARATOR'), ('VALUE', 'SLIDER').

S W G T Y P

The routine SWGTYP modifies the appearance of certain widgets.

The call is: CALL SWGTYP (CTYPE, CLASS)

or: void swgtyp (const char *ctype, const char *class);

CTYPE — is a character string containing a keyword:

= 'VERT' — means that list elements in box widgets or scale widgets will be displayed in vertical direction. Lines plotted by WGSEP will have a vertical orientation.

= 'HORI' — means that box widgets, scale widgets and progress bars will be displayed in horizontal direction. Lines plotted by WGSEP will have a horizontal orientation.

= 'GRID' means that box widgets will be displayed in matrix form.

= 'SCROLL' means that scrollbars will be created in list, table, draw and main widgets.

= 'NOSCROLL' means that no scrollbars will be created in list, table, draw and main widgets.

= 'OPEN' means that a file selection box for reading files is defined.

= 'SAVE' means that a file selection box for saving files is defined.

= 'STRING' means that a popup menu can be directly connected with a callback routine. Normally, menu entries in a popup menu can be connected with callback routines.

= 'NORESIZE' means that the size of the main widget cannot be changed with the mouse. The default behaviour is 'RESIZE'.

= 'NOEDIT' defines non editable text widgets.

= 'PASS' enables hidden password input for text widgets.

CLASS is a character string containing the widget class where CLASS can have the values 'LIST', 'BOX', 'SCALE', 'PBAR', 'TABLE', 'DRAW', 'FILE', 'SEPARATOR', 'POPUP', 'MAIN' and 'TEXT'.
If CLASS has one of the values 'BOX', 'SCALE', 'PBAR' and 'SEPARATOR', CTYPE can have the values 'VERT' and 'HORI'. The class 'BOX' can have the additional value 'GRID'.
If CLASS = 'FILE', CTYPE can have the values 'OPEN' and 'SAVE'.
If CLASS = 'POPUP', CTYPE can have the values 'STRING' and 'MENU'.
For CLASS = 'MAIN', CTYPE can have the values 'RESIZE' and 'NORESIZE', or 'SCROLL' and 'NOSCROLL'.
For CLASS = 'TEXT', CTYPE can have the values 'EDIT', 'NOEDIT' and 'PASS'.
Defaults: ('HORI', 'SEPARATOR'), ('OPEN', 'FILE'), ('MENU', 'POPUP'), ('RESIZE', 'MAIN'), ('NOSCROLL', 'MAIN'), ('EDIT', 'TEXT').

S W G A T T

The routine SWGATT sets widget attributes.

The call is: CALL SWGATT (ID, CATT, CKEY)

or: void swgatt (int id, const char *catt, const char *ckey);

ID is a widget ID.

CATT is a character string containing an attribute.

CKEY is a character string containing a keyword:

= 'CLOSE' The keyword 'CLOSE' can have the attributes 'ACTIVE' and 'INACTIVE'. It allows a user to disable the close button of the main widget.

= 'MENU' The menu header line in a main widget can be disabled with the attribute 'OFF'. The default value is 'ON'. ID should contain the widget ID of the main widget.

= 'MINI' This option allows to disable the minimize button in the header line of a main widget. CATT can have the values 'OFF' and 'ON'.

= 'MAXI' This option allows to disable the maximize button in the header line of a main widget. CATT can have the values 'OFF' and 'ON'.

= 'ICON' The icon in the header line of a main widget can be replaced by an icon in a .ico file (only Windows). CATT should contain the name of the .ico file.

S W G I O P

The routine SWGIOP sets integer options for widgets.

The call is: CALL SWGIOP (N, CKEY)

or: void swgiop (int n, const char *ckey);

N is an integer option.

CKEY is a character string containing a keyword:

= 'TABLE' means that N defines the number of visible rows in scrolled table widgets.

= 'LIST' means that N defines the number of visible entries in scrolled list widgets.

= 'DLIST' means that N defines the width of the list in dropping list widgets. For N = 0, the list has the same width as the widget. A negative value sets the width of the list in pixel, a positive value the width in number of characters.

= 'HMARGIN' sets the horizontal margin in text and push button widgets (only X11). The default margins are 5 for text widgets and 2 for push button widgets. N = -1 defines this values.

= 'VMARGIN' sets the vertical margin in text and push button widgets (only X11). The default margins are 5 for text widgets and 2 for push button widgets. N = -1 defines this values.

= 'ICON' sets an icon ID for the routines WGICON, WGPICON, and for the header line of a main widget. The icon is taken from the program resource. N = -1 resets this option. (Only Windows).

Defaults: (8, 'TABLE'), (8, 'LIST'), (0, 'DLIST'), (-1, 'HMARGIN'), (-1, 'VMARGIN'), (-1, 'ICON').

S W G C B 3

The routine SWGCB3 defines callback routines for mouse wheel events in draw widgets.

The call is: CALL SWGCB3 (ID, ROUTINE)

or: void swgcb3 (int id, (void) (*routine) (int id, int ival));

ID is a widget ID.

IVAL is the name of a routine defined by the user. In Fortran, the routine must be declared as EXTERNAL. The parameters passed to the callback routine are the widget ID and an integer variable, that can have the values 1 und -1. A positive value means that the wheel was rotated forward away from the user, a negative value indicates that the wheel was rotated backward.

S W G B G D

The routine SWGBGD changes the background colour of a widget.

The call is: CALL SWGBGD (ID, XR, XG, XB)

or: void swgbgd (int id, float xr, float xg, float xb);

ID	is the widget ID.
XR, XG, XB	are the RGB colour values between 0 and 1.

S W G F G D

The routine SWGBGD changes the foreground colour of a widget.

The call is:	CALL SWGFGD (ID, XR, XG, XB)
or:	void swgfgd (int id, float xr, float xg, float xb);
ID	is the widget ID.
XR, XG, XB	are the RGB colour values between 0 and 1.

S W G T X T

The routine SWGTXT changes the value of a text widget and the text string of label and push button widgets including the labels of the widgets created by WGOK and WGQUIT.

D W G E R R

The routine DWGERR returns a status for the routines DWGFIL, DWGTXT and DWGLIS. The routine can be used to check directly after the routines above if the OK button is pressed in the routines.

The call is:	CALL DWGERR (ISTAT)
or:	int dwgerr (void);
ISTAT	is a returned status. If ISTAT = 0, the OK button in the routines DWGFIL, DWGTXT and DWGLIS is pressed. Otherwise, the CANCEL button is pressed, or an error occured.

G W G S C L

The routine GWGSCL returns the value of a scale widget, or the value of a scrollbar in draw widgets.

The call is:	CALL GWGSCL (ID, XVAL)
or:	float gwgscl (int id);
ID	is a widget ID of a scale or draw widget. If the widget ID is passed as it's negative value, the value of the vertical scrollbar of a draw widget is returned. Otherwise, the value of the horizontal scrollbar.
XVAL	is the returned value.

G W G S I Z

The routine GWGSIZ returns the size of widgets.

The call is:	CALL GWGSIZ (ID, NW, NH)
or:	void gwgsiz (int id, int *nw, int *nh);
ID	is the index of a widget, which must not be a parent, base or popup widget.
NW, NH	are the returned width and height of the widget in pixels.

G W G G U I

The routine GWGGUI returns the used GUI of the Dislin library.

The call is:	CALL GWGGUI (IRET)
or:	int gwggui (void);
IRET	identifies the used GUI. IRET = 1 means OpenMotif, IRET = 2 GTK and IRET = 3 Windows API.

I T M N C A T

The routine ITMNCAT concatenates an element to a list string.

The call is:	CALL ITMNCAT (CLIS, N, CITEM)
or:	void itmncat (char *clis, int n, char *item);
CLIS	is a character string that contains the list elements (s. WGLIS).
N	is the maximal number of characters that can be stored in CLIS.
CITEM	is a character string that will be concatenated to CLIS. If CLIS is blank, CITEM will be the first element in CLIS.
Additional notes: -	Trailing blanks in CLIS and CITEM will be ignored.
-	ITMNCAT is a replacement of ITMCAT with the additional parameter N for avoiding buffer overflows. ITMCAT is still in the library for providing compatibility.

F R E E P T R

The routine FREEPTR deallocates space that is allocated in a Dislin routine before. The routine is only useful for C.

The call is:	void freeptr (void *ptr);
ptr	is a pointer to a memory address.

Chapter 16: Quick Plots

Q P L C R V

QPLCRV is a similar routine to QPLOT, but can display multiple curves.

The call is:	CALL QPLCRV (XRAY, YRAY, N, COPT)	level 0, 1
or:	void qplcrv (const float *xray, const float *yray, int n, const char *copt);	
XRAY, YRAY	are arrays that contain X- and Y-coordinates.	
N	is the number of data points.	
COPT	is a character string that describes the meaning of the curve. COPT can have the values 'FIRST', 'NEXT' and 'LAST'.	

Q P L S C L

QPLSCL overwrites the automatic scaling of quick plots.

The call is:	CALL QPLSCL (A, E, OR, STEP, CAX)	level 0, 1
or:	void qplscl (float a, float e, float or, float step, const char *cax);	
A, E	are the lower and upper limits of the axis.	
OR, STEP	are the first axis label and the step between labels.	
CAX	is a character string that defines the axes. CAX can contain the characters 'X', 'Y' and 'Z'.	