

New Dislin Features since Version 10.0

This article describes new features and options of Dislin which are added to the software since version 10.0 and not covered by the Dislin book. The current version number of Dislin is 10.2.1.

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.metapl ("cons");
  g.disini ();
  g.messag ("This is a test", 100, 100);
  g.disfin ();
}
```

Chapter 3: Introductory Routines

SYMBOL

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		

Chapter 5: Plotting Curves

LEGPAT

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.

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

LINMOD

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 (char *cmod, 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')

SHDPAT

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		

Chapter 8: Elementary Plot Routines

TRIFLC

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 (float *xray, float *yray, 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.

Chapter 10: Business Graphics

FBARS

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 (float *xray, float *y1ray, float *y2ray, float *y3ray, 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 financial 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 (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 12: 3-D Graphics

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 (float *xray, float *yray, float *zray, 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 (float *xray, int ixdim, float *yray, int iydim, float *zmat, 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.

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.

Chapter 13: Geographical Projections and Plotting Maps

M A P I M G

The routine MAPIMG plots a 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:	<code>void mapimg (char *cfil, float x1, float x2, float x3, float x4, float x5, float x6);</code>
CFIL	is a character string that contains the name of a BMP or GIF 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.

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 S E P

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

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 (char *ctype, 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.
- = '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.

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 = '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'.

For CLASS = 'TEXT', CTYPE can have the values 'EDIT' and 'NOEDIT'.

Defaults: ('HORI', 'SEPARATOR'), ('OPEN', 'FILE'), ('MENU', 'POPUP'), ('RESIZE', 'MAIN'), ('EDIT', 'TEXT').

SWGOPT

The routine SWGOPT sets widget options.

The call is: `CALL SWGOPT (COPT, CKEY)`

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

COPT is a character string containing an option.

CKEY is a character string containing a keyword:

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

= '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.

Defaults: ('STANDARD', 'SEPARATOR'), ('STANDARD', 'DIALOG').

SWG I O P

The routine SWGIOP sets integer options for widgets.

The call is: `CALL SWGIOP (N, CKEY)`

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

N is an integer number.

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.

Defaults: (8, 'TABLE'), (8, 'LIST'), (0, 'DLIST').

DWGERR

The routine DWGERR returns a status for the routines DWGFIL, DWGTXR 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 DWG-FIL, DWGTXN and DWGLIS is pressed. Otherwise, the CANCEL button is pressed, or an error occurred.

GWGSIZ

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.

Chapter 16: Quick Plots

QPLCRV

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 (float *xray, float *yray, int n, 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'.

QPLSCL

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, 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'.