GUIDE TO USING
DISPLAY INTEGRATED SOFTWARE
SYSTEM AND PLOTTING LANGUAGE (DISSPLA)

KEVIN G. BRADY

APPROVED FOR PUBLIC RELEASE:
DISTRIBUTION UNLIMITED
DESTRUCTION NOTICE — For classified documents, follow the procedures in DOD 5220.22M, Industrial Security Manual, Section II-9, or DOD 5200.1-R, Information Security Program Regulation, Chapter IX. For unclassified, limited documents, destroy by any method that will prevent disclosure of contents or reconstruction of the document.
Guide to Using Display Integrated Software System and Plotting Language (DISSPLA)

Brady, Kevin Gerard

This document contains a brief discussion and examples using Display Integrated Software System and Plotting Language (DISSPLA) at Central Computer Facility.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Conventions</td>
<td>1</td>
</tr>
<tr>
<td>Devices</td>
<td>2</td>
</tr>
<tr>
<td>Illegal Unit Numbers</td>
<td>3</td>
</tr>
<tr>
<td>Example 1</td>
<td>5</td>
</tr>
<tr>
<td>The Level Structure</td>
<td>10</td>
</tr>
<tr>
<td>Example 2</td>
<td>13</td>
</tr>
<tr>
<td>Two Dimensional Graphing</td>
<td>14</td>
</tr>
<tr>
<td>Example 3</td>
<td>17</td>
</tr>
<tr>
<td>Example 4</td>
<td>21</td>
</tr>
<tr>
<td>Legends and Stories</td>
<td>22</td>
</tr>
<tr>
<td>Example 5</td>
<td>25</td>
</tr>
<tr>
<td>Example 6</td>
<td>27</td>
</tr>
<tr>
<td>Interpolation</td>
<td>28</td>
</tr>
<tr>
<td>Example 7</td>
<td>31</td>
</tr>
<tr>
<td>Example 8</td>
<td>35</td>
</tr>
<tr>
<td>Post-Processor</td>
<td>36</td>
</tr>
<tr>
<td>Example 9</td>
<td>37</td>
</tr>
<tr>
<td>3-Dimensional Plotting</td>
<td>38</td>
</tr>
<tr>
<td>Example 10</td>
<td>41</td>
</tr>
<tr>
<td>Example 11</td>
<td>45</td>
</tr>
<tr>
<td>Example 12</td>
<td>47</td>
</tr>
<tr>
<td>Graffiti Plots</td>
<td>48</td>
</tr>
<tr>
<td>Example 13</td>
<td>51</td>
</tr>
<tr>
<td>Example 14</td>
<td>52</td>
</tr>
<tr>
<td>Character Sets</td>
<td>53</td>
</tr>
<tr>
<td>Example 15</td>
<td>55</td>
</tr>
<tr>
<td>Instruction Alphabet</td>
<td>56</td>
</tr>
<tr>
<td>Example 16</td>
<td>59</td>
</tr>
</tbody>
</table>

(i)
Introduction

This document is intended to give the user a basic understanding of Display Integrated Software System and Plotting Language. All examples were taken from implementations at David Taylor Naval Ship R & D Center to show actual scientific uses. All examples were run, unless specifically stated, on the VAXcluster using version 9.2 of DISSPLA. This version will soon be available on the CYBERs and aside from certain machine dependencies, all examples will then run on the CYBERs. In interactive examples, underlined commands are typed by the user.

Conventions

Throughout this document the following argument naming conventions hold:

ARGUMENTS:

- X, IX, LX - Refer to the X-axis
- Y, LY, LY - Refer to the Y-axis
- I, J, K, M, N - Integer Values
- All Others - Real Values
- L - String of text
- RAY - Array of values

NOTATION:

- {...} - Within a program listing are comments
- (underline) - In examples are user type-ins
- p/s - parameter setting
Devices

The call to a device **MUST** be the first call in a DISSPLA program. The following devices are currently active; other devices may be activated at the user's request, these calls are supported by version 9.2 which is currently only on the VAXcluster and should soon be on the CYBERs.

Tektronix 4000 Series:

```plaintext
CALL TK4006 (icps)
CALL TK4010 (icps)
CALL TK4012 (icps)
CALL TK4013 (icps)
CALL TK4014 (icps, idevc)
CALL TK4015 (icps, idevc)
CALL TK4016 (icps, idevc)
CALL TK4025
CALL TK4027
CALL TK4051 (icps)
CALL TK4052 (icps)
CALL TK4054 (icps, idevc)
CALL TEKALL (imodel, icps, 0, idevc, 0)
CALL PTEKAL {will prompt for all information}
```

- **icps** - Baud rate (30, 120, 240, 480, 960)
- **idevc** - is 1 for high resolution (4096)
  - is 0 for low resolution (1024)

Tektronix 4100 Series:

```plaintext
CALL TK41 (IMODEL)
```

- **IMODEL** - 4105, 4107, 4109, 4112, 4113, 4114, 4115, 4116

```plaintext
CALL PTK41 {will prompt for model number}
```
DEC Terminals:

CALL REGIS ( ioptin, idevc )

ioptin - 1 ( VT125 )
- 2 ( VK100 )
- 3 ( VT240, VT241 )
- 4 ( DEC PRO 350 )

idevc - 0 (Monochrome Monitor)
- 1 (Color Monitor )

CALL PREGIS  {will prompt for information}

CALCOMP:

CALL CALCOMP (0, 0, 10)

The file CALCOMPOUT.DAT is created for processing to tape using the procedure VSYS:CALCD2T on the VAX.

The file TAPE10 is created for processing to tape on either of the CYBERs.

META FILE:

CALL COMPRS

The file DISPLOT.DAT is created for processing by the Post-Processor on the VAX. (RUN VSYS:DISPOP)

The file PLFILE is created for processing by the Post-Processor on either of the CYBERs. (DISPOST)

Illegal Unit Numbers

Unit Numbers used by DISSPLA which CANNOT be used in your program:

31, 32, 33  Scratch files
90, 91, 93  Mapping and Landblanking
95         META file for COMPRS
96         Font files
97         Scratch file
10         CALCOMP file
This is an example of running a DISSPLA program on the VAXcluster.

```
PROGRAM EXAMPLE1
{ This example is for the VAX }
DIMENSION X(10), Y(10)
DATA Y /1, 2, 3, 4, 5, 6, 7, 8, 9, 10/
DATA X /1, 2, 3, 4, 5, 6, 7, 8, 9, 10/
CALL COMPRS
CALL BGNPL (0)
CALL TITLE ('This is a Simple Graph$', 100, 'X-Axis$', 100,
'Y-Axis$', 100, 6., 8.)
CALL GRAF (0., 1., 10., 0., 1., 10.)
CALL CURVE (X, Y, 10, 1)
CALL ENDPPL (0)
CALL DONEPL
END
```

$ fortran myfile
$ dislink myfile
Other Libraries (y or n): n
$ run myfile

PLOTTING COMMENCING

NO. OF FIRST PLOT 0

END OF DISSPLA 9.2 -- 894 VECTORS IN 1 PLOTS.
RUN ON 1/17/86 USING SERIAL NUMBER 3105 AT DTNSRDC VAX
PROPRIETARY SOFTWARE PRODUCT OF ISSCO, SAN DIEGO, CA.
403 VIRTUAL STORAGE REFERENCES; 5 READS; 0 WRITES.
$ run vsys:dispop
ENTER DEVICE TYPE....
1=TEK 40xx, 2=CALCOMP 1051, 3=TEK 41xx, 4=REGIS (VT240), 5=PRINTER PLOT

3
ENTER MODEL NUMBER { Enter your device }
4105
ENTER POST-PROCESSOR DIRECTIVES
<CR>
PLOT FILE GENERATED BY Kevin Brady
AT DTNSRDC VAX
ON JAN 17, 1986 8:46
Example 1

This is a Simple Graph
This command procedure on the VAXcluster will do the entire process with no user interaction:

```
$ on error then exit
$ again:
$ if p1 eqs. "" then inquire p1 "Filename"
$   1 Keep prompting for filename
$ if p1 eqs. "" then goto again
$ fortran 'p1'
$   1 libraries linked twice because of nowrap search
$ link 'p1',vsys:dispintf/library,disspla/library,-
   dispintf/library,disspla/library
$ run 'p1'
$ run vsys:dispop
3
4105   { This line must be changed to your device }
$ exit
```
The same program run interactively on either of the CYBERs to create a CALCOMP plot:

```
PROGRAM EXAMPLE1
  DIMENSION X(10), Y(10)
  DATA Y/1, 2, 3, 4, 5, 6, 7, 8, 9, 10/
  DATA X/1, 2, 3, 4, 5, 6, 7, 8, 9, 10/
  CALL COMPRS
  CALL BGNPL (0)
  CALL HEIGHT (0.2)
  CALL TITLE ("THIS IS A SIMPLE GRAPH", 22, "X-AXIS", 6, 22, "Y-AXIS", 6, 6, 8.)
  CALL GRAF (0., 1., 10., 0., 1., 10.)
  CALL CURVE (X, Y, 10, 1)
  CALL ENDP (0)
  CALL DONEP {Double quotes are required on CYBERs}
END
```

```
COMMAND-CONNECT, OUTPUT
COMMAND-ATTACH, INFILE, MYPLOT, ID-CAKB
  AT CY= 001 SN=SYSSET
COMMAND-ATTACH, DISSPLA
  PFN IS DISSPLA
  AT CY= 006 SN=SYSSET1
COMMAND-REQUEST, TAPE10,*PF
COMMAND-ATTACH, NSRDC
  PFN IS NSRDC
  AT CY= 105 SN=SYSSET1
COMMAND-LIBRARY, DISSPLA, NSRDC
COMMAND-FTNS,I=INFILE,L=O
  56100 CM STORAGE USED.
  0.020 CP SECONDS COMPILATION TIME.
COMMAND-LGO
  PLOTTING COMMENCING
  .......................

..... DISSPLA VERSION 8.2 ......
NO. OF FIRST PLOT  0

END OF DISSPLA 8.2 -- 897 VECTORS GENERATED IN 1 PLOT FRAMES.
- ISSCO- 4186 SORRENTO VALLEY BLVD., SAN DIEGO CALIF. 92121

DISSPLA IS A CONFIDENTIAL PROPRIETARY PRODUCT OF ISSCO AND ITS USE IS SUBJECT TO A NONDISSEMINATION AND NONDISCLOSURE AGREEMENT.
NON-FATAL LOADER ERRORS - SEE MAP
END PLOTIT
61300 MAXIMUM EXECUTION FL.
0.132 CP SECONDS EXECUTION TIME.
COMMAND-RETURN, DISSPLA, NSRDC
COMMAND-ATTACH, DISPOST
PFN IS DISPOST
AT CY= 007 SN=SYSSET1
COMMAND-LIBRARY, DISPOST
COMMAND-POP936

* * * * * * * * * * * CALCOMP POST-PROCESSOR * * * * * * * * * *
INPUT DIRECTIVES
{ Input space then <CR> }

PLOT FILE GENERATED BY CAKB0E2 AT 10.05.27 ON 01/17/86
1 THE NUMBER OF CALCOMP BLOCKS IN THIS FILE IS 3
1 PLOTS HAVE BEEN PROCESSED

.............................. END OF POSTPROCESSOR ...............................
END POP936
COMMAND-CATALOG, TAPE10, ID=CAKB
NEWCYCLE CATALOG
RP = 030 DAYS
CT ID= CAKB PFN=TAPE10
CT CY= 002 000000016 PRUS $0000.04 /DAY
CT SN= SYSSET
COMMAND-
The following procedure, stored in the user's permanent file PLOTIT, will let him view his output on a Tektronix terminal:

```
.PROC,PLOTIT.
ATTACH,INFILE,MYPLOT,ID=CAKB.
ATTACH,DISSPLA.
REQUEST,PLFILE,*PF.
ATTACH,NRSRC.
LIBRARY,DISSPLA,NRSRC.
FTN5,I=INFILE,L=0.
LGO.
CATALOG,PLFILE,ID=CAKB.
RETURN,DISSPLA,NRSRC.
ATTACH,DISPOST.
LIBRARY,DISPOST.
TEK300.
REVERT.
```

To execute type:

```
COMMAND-ATTACH,PLOTIT,ID=xxxx
COMMAND-PLOTIT
```

**NOTE:** Currently on the CYBERs there are two versions of the Tektronix post-processor, the one attached above is used for color plots and **WILL NOT** stop between plots if there are more than one. To view multiple plots with a pause between each one, the other version of the post processor must be used. Instead of the "ATTACH,DISPOST" and "LIBRARY,DISPOST", `MSFETCH` the file `TEK300" ("MSFETCH,TEK300,UN=CSYS") and continue.
Level Structure

The axis system cannot be drawn until the sub-plot area and physical origin are defined. Similarly, curves cannot be drawn until the axis system is established. Thus, DISSPLA has a level structure to ensure that all necessary information is present during the construction of a plot. An error message is printed if a routine is called at the wrong level.

The DISSPLA levels are:

0 - before device initialization
1 - after device initialization
2 - after page border, physical origin, and subplot area defined
3 - after axis system is defined

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL DEVICE</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>CALL AREA2D (XAXIS, YAXIS)</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>CALL XNAME (LXNAME, YXNAME)</td>
<td>2</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL YNAME (LYNAME, IYNAME)</td>
<td>2</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL HEADIN (LHEAD, IHEAD, HTMULT, NLINES)</td>
<td>2, 3</td>
<td>same</td>
</tr>
<tr>
<td>CALL GRAF (XORG, XSTP, XMAX, YORG, YSTP, YMAX)</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>CALL CURVE (XARRAY, YARRAY, NPTS, IMARK)</td>
<td>3</td>
<td>same</td>
</tr>
<tr>
<td>CALL ENDPL (Iplot)</td>
<td>2, 3</td>
<td>1</td>
</tr>
<tr>
<td>CALL DONEPL</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Subroutine

To change page size from 8.5 x 11 inch default:

CALL PAGE (XPAGE, YPAGE) 1 p/s

To position the physical origin:

CALL PHYSOR (XPHYS, YPHYS) 1 p/s

To re-position the origin relative to PHYSOR:

CALL OREL (XOREL, YOREL) 1 p/s

To end the plot and remain on same page:

CALL ENDGR (IPlot) 2,3 1

To frame the subplot area:

CALL FRAME 2,3 same

To understand the concept of setting up the plot, consider the following example. The page will be set to 24" X 11", and will contain two subplots. The lower left corner of the first plot is positioned at the point (3",2") by PHYSOR. AREA2D then sets the current plotting area to 8" X 6". Now the plotting can begin (although no plots are actually done in this example). The first plot is then ended by ENDGR, which tells DISSPLA you are done with the first plot but will remain on the same page to do another plot. The lower left corner of the second plot is then put at (13",2") by OREL (moved relative to previous origin). AREA2D is called again to define the second plot area, and plotting could commence. This plot is ended by ENDGR again then ENDPL terminates the plot on this page.
PROGRAM EXAMPLE2
CALL COMPRS
CALL BGNPL (0)
CALL PAGE (24., 11.)
CALL PHYSOR (3., 2.)
CALL AREA2D (8., 6.)
CALL FRAME
CALL HEADIN ('THIS IS SUBPLOT # 1', 100, 3., 1)
CALL ENDGR (0)
CALL OREL (10., 0.)
CALL AREA2D (8., 6.)
CALL FRAME
CALL HEADIN ('THIS IS SUBPLOT # 2', 100, 3., 1)
CALL ENDGR (0)
CALL DONEPL
END

NOTE: Programs are for VAXcluster only!
Example 2

![Diagram showing two subplots: subplot #1 with dimensions 6" x 8" and subplot #2 with dimensions 6" x 8".](image)
Two Dimensional Graphing

Once the subplot area has been defined, the next step is to define the axis system. Below are some of the major routines, refer to the User's Manual for a more complete description.

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL GRAF (XORG, XSTP, XMX, YORG, YSTP, YMX)</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>CALL XLOG (XORIG, XCYCLE, YORIG, YSTEP)</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>CALL YLOG (XORIG, XSTEP, YORIG, YSTEP)</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>CALL LOGLOG (XORIG, XCYCLE, YORIG, YCYCLE)</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>CALL POLAR (THEFAC, RSTEP, XDIST, YDIST)</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>CALL GRID (IXGRID, IYGRID)</td>
<td>3</td>
<td>Same</td>
</tr>
</tbody>
</table>

To position messages in INCHES:

| CALL MESSAG (LMESS, IMESS, XPOS, YPOS) | 2, 3 | Same |
| CALL REALNO (ANUM, IPLACE, XPOS, YPOS) | 2, 3 | Same |
| CALL INTNO (INUM, XPOS, YPOS)         | 2, 3 | Same |
| CALL VECTOR (XFROM, YFROM, XTO, YTO, IVEC) | 2, 3 | Same |

To position messages in DATA units:

| CALL RLMESS (LMESS, IMESS, XVAL, YVAL) | 3   | Same |
| CALL RLREAL (ANUM, IPLACE, XVAL, YVAL) | 3   | Same |
| CALL RLINT (INUM, XVAL, YVAL)         | 3   | Same |
| CALL RLVEC (XFROM, YFROM, XTO, YTO, IVEC) | 3   | Same |
Now let us reconsider example #2 as it actually appeared when ran. Example #3 is the same program with the sequence of commands used to draw the vectors (with arrowheads) and the integer distances included. It is actually three plots (the first two are the subplots, the third is the entire page) so that the vectors can be placed around the subplot areas but not off of the page. First the vector is drawn (in inches from the origin), then INTNO places the value currently into the variable INUM (integer) at the coordinates specified (in inches from the origin). Then MESSAG is called to place the inch symbol (notice that 'ABUT' is used for both coordinates) directly after the previous symbol it just wrote, in this case directly after what INTNO wrote.
PROGRAM EXAMPLE3
CALL COMPRESS
CALL BGNPL (0)
CALL PAGE (24., 11.)
CALL PHYSOR (3., 2.)
CALL AREA2D (8., 6.)
CALL FRAME
CALL HEADING ('THIS IS SUBPLOT # 1$'
,100, 3., 1.)
CALL ENDGR (0)
CALL OREL (10., 0.)
CALL AREA2D (8., 6.)
CALL FRAME
CALL HEADING ('THIS IS SUBPLOT # 2$'
,100, 3., 1.)
CALL ENDGR (0)
CALL PHYSOR (0., 0.)
CALL AREA2D (20., 11.)
CALL HEIGHT (.5)
CALL SETCLR ('RED')
CALL VECTOR (3., 0., 3., 2., 3402)
CALL INTNO (INUM, 3.5, .75)
CALL MESSAG ('""', 2, 'ABUT', 'ABUT')
CALL VECTOR (0., 4., 3., 4., 3402)
CALL INTNO (INUM, 21.5, 9.5)
CALL MESSAG ('""', 2, 'ABUT', 'ABUT')
CALL VECTOR (3., 2., 3., 11., 3, 3402)
CALL INTNO (INUM, 22.5, 4.5)
CALL MESSAG ('""', 2, 'ABUT', 'ABUT')
CALL ENDPL (0)
CALL MESSAG ('""', 2, 'ABUT', 'ABUT')
CALL ENDPL
Example 1

THIS IS SUBPLOT # 1

THIS IS SUBPLOT # 2
Once DISSPLA is in level 3, plotting of curves may begin. The call to CURVE may be made as many times as needed.

Subroutine

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>To draw a curve:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL CURVE (XARAY, YARAY, NPTS, IMARK)</td>
<td>3</td>
<td>Same</td>
</tr>
<tr>
<td>XARAY - array of X values</td>
<td></td>
<td></td>
</tr>
<tr>
<td>YARAY - array of Y values</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NPTS - number of points to plot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMARK - &lt; 0 symbols every Ith point, not connected</td>
<td></td>
<td></td>
</tr>
<tr>
<td>= 0 points connected, no symbols</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&gt; 0 symbols every Ith mark, connected</td>
<td></td>
<td></td>
</tr>
<tr>
<td>To blank an area around the curve:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL BLCURV (OFWDTH, OFLNTH)</td>
<td>1, 2, 3</td>
<td>p/s</td>
</tr>
<tr>
<td>To thicken curves:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL THKCRV (THKNSS)</td>
<td>1, 2, 3</td>
<td>p/s</td>
</tr>
<tr>
<td>To change the grace margin of subplot:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL GRACE (GRACEM)</td>
<td>1, 2, 3</td>
<td>p/s</td>
</tr>
<tr>
<td>To select a marker:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL MARKER (ISYM)</td>
<td>1, 2, 3</td>
<td>p/s</td>
</tr>
<tr>
<td>To blank symbol markers:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL BLSYM</td>
<td>1, 2, 3</td>
<td>p/s</td>
</tr>
<tr>
<td>To change size of markers:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL SCLPIC (FACTOR)</td>
<td>1, 2, 3</td>
<td>p/s</td>
</tr>
</tbody>
</table>
This example reads data points into two arrays, then plots the curve. This object must be drawn to scale, so PAGE is set to 23" x 11". AREA2D then sets the plot size to 20" x 8" (from within TITLE). SETDEV directs error messages and summary messages to unit # 3 (FOR003.DAT on the VAXcluster; TAPE# on the CYBERs), so that they will not appear on the screen and a hardcopy is kept. INTAXS will print only the integer values along the axes, and YAXANG will write the values along the Y axis horizontally. GRAF sets up the scale of the axes, the X axis will run from 0 to 20 by steps of 1, the Y axis will run from -4 to +4 by steps of 1. CURVE is then called twice, once to plot the upper portion, and the second time to plot the lower portion.
PROGRAM EXAMPLE4

DIMENSION X(10000), Y(10000)
OPEN (1, FILE='FOR001.DAT', STATUS='OLD')
OPEN (2, FILE='FOR002.DAT', STATUS='OLD')

CALL COMPRES  \{ Level 0 to Level 1 \}
CALL BGNPL (0)
CALL SETDEV (3, 3)
CALL SETCLR ('BLUE')
CALL PAGE (23.0, 11.0)
CALL HEIGHT (.3)
CALL SWISSB
CALL SHDCHR (90., 1, .002, 1)
CALL INTAXS
CALL YAXANG (0.)
CALL TITLE ('MLTA FOIL PROFILE$', 100, 'X (INCHES)$', 100, 'Z (INCHES)$', 100, 20., 8.) \{ Level 1 to Level 2 \}
CALL GRAF (0., 1., 20., -4., 1., 4.) \{ Level 2 to Level 3 \}
CALL SETCLR ('RED')
DO 20 I=1, 10000
   READ (1, '(E15.5,3X,E15.5)', END=21) X(I), Y(I)
20 CONTINUE

21 CALL CURVE (X, Y, 9500, 0)
DO 30 I=1, 10000
   READ (2, '(E15.5,3X,F15.5)', END=32) X(I), Y(I)
30 CONTINUE

32 CALL CURVE (X, Y, 9500, 0)
CALL ENDP1 (0) \{ Level 3 to Level 1 \}
CALL DONEPL \{ Level 1 to Level 0 \}
END
Example 4

MLTA FOIL PROFILE

X (inches)

Z (inches)
**Legends and Stories**

To draw a Legend for your graph, or write out a block of text (Story), use one of the following routines.

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL LINES (LSTRING, IPKARAY, ILINE)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>CALL LEGEND (IPKARAY, NLINES, XPOS, YPOS)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>J = LINESS (IPKRAY, NPKWRO, IMAX)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>CALL STORY (IPKARAY, NLINES, XPOS, YPOS)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>CALL LSTORY (IPKARAY, NLINES, XPOS, YPOS)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>CALL RSTORY (IPKARAY, NLINES, XPOS, YPOS)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>XLEN = XLEGEND (IPK, NLINES)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>YLEN = YLEGEND (IPK, NLINES)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>XLEN = XSTORY (IPK, NLINES)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
<tr>
<td>YLEN = YSTORY (IPK, NLINES)</td>
<td>1,2,3</td>
<td>Same</td>
</tr>
</tbody>
</table>
To get the length of messages:

\[ \text{XLEN} = \text{XMESS} (\text{LMESS}, \text{IMESS}) \]

To get the length of numbers:

\[ \begin{align*}
\text{XLEN} &= \text{XREAL} (\text{ANUM}, \text{IPLACE}) \{\text{REALS}\} \\
\text{XLEN} &= \text{XINT} (\text{INT}) \{\text{INTEGER}\}
\end{align*} \]

To get X,Y position in inches of a data point:

\[ \begin{align*}
A &= \text{XPOSN} (\text{XVAL}, \text{YVAL}) \\
B &= \text{YPOSN} (\text{XVAL}, \text{YVAL})
\end{align*} \]

To get X,Y coordinate values:

\[ \begin{align*}
A &= \text{XINVRS} (\text{XINCH}, \text{YINCH}) \\
B &= \text{YINVRS} (\text{XINCH}, \text{YINCH})
\end{align*} \]

The next example uses a legend containing 9 lines which are packed by LINES and stored. MYLEGN changes the title on the legend to any string. BLREC uses the values of XLEGEND and YLEGEND to blank out the area where the legend will be written, to prevent curves from over writing it. Notice that LEGLIN is called directly before CURVE, the line type is stored with the legend titles. \textbf{NOTE:} the curves must be called in the same order as they appear in the legend. LEGEND then prints out the legend at the specified coordinates.

The second example is basically the same except SPCMOD is called. This subroutine calls MYSPEC and also stores the color of the line that was used.
PROGRAM EXAMPLE5
DIMENSION IPAX(10000), Y(10), Z(10)
OPEN(1, FILE='JETDATA', STATUS='OLD')
CALL COMPRS
CALL BGNPL (0)
CALL SETDEV (2, 2)
CALL SETCLR ('BLUE')
CALL PAGE (12.0, 11.0)
CALL SWISSB
CALL INTAXS
CALL YAXANG (0.)
CALL SHDCHR (90., 1, .002, 1)
CALL TITLE ('IMPACT WINDOWS', 100, 'Z - IMPACTS', 100, 'Y - IMPACTS', 100, 9., 7.)
CALL GRAF (-200., 20., 200., -200., 20., 200.)
CALL SETCLR ('RED')
CALL LINES (.0000 - .0625$, IPAX, 1)
CALL LINES (.0625 - .1220$, IPAX, 2)
CALL LINES (.1250 - .2500$, IPAX, 3)
CALL LINES (.2500 - .5000$, IPAX, 4)
CALL LINES (.5000 - .7500$, IPAX, 5)
CALL LINES (.7500 - 1.000$, IPAX, 6)
CALL LINES (1.000 - 2.000$, IPAX, 7)
CALL LINES (2.000 - 3.000$, IPAX, 8)
CALL LINES (3.000 - 4.000$, IPAX, 9)
CALL LINES (' > 4.000$', IPAX, 10)
CALL MYLEGN ('DEPTH IN INCHES$', 100)
XR = XLEGND (IPAX, 10) + .3
YR = YLEGND (IPAX, 10) + .3
CALL BLREC (7.-.3, 6.-.3, XR+.3, YR+.3, .02)
DO 20 1=1,500
   READ (1, '(70X,F5.2,2X,F9.2,2X,F9.2)', END=32) DEP, Y(1), Z(1)
   CALL MARKER (1)
   IF ((DEP.GE.0.0) .AND. (DEP.LT.0.0625)) CALL MARKER (2)
   IF ((DEP.GE.0.0625) .AND. (DEP.LT.0.1250)) CALL MARKER (3)
   IF ((DEP.GE.0.1250) .AND. (DEP.LT.0.2500)) CALL MARKER (4)
   IF ((DEP.GE.0.2500) .AND. (DEP.LT.0.5000)) CALL MARKER (5)
   IF ((DEP.GE.0.5000) .AND. (DEP.LT.0.7500)) CALL MARKER (6)
   IF ((DEP.GE.0.7500) .AND. (DEP.LT.1.0000)) CALL MARKER (7)
   IF ((DEP.GE.1.0000) .AND. (DEP.LT.2.0000)) CALL MARKER (8)
   IF ((DEP.GE.2.0000) .AND. (DEP.LT.3.0000)) CALL MARKER (9)
   IF ((DEP.GE.3.0000) .AND. (DEP.LT.4.0000)) CALL MARKER (10)
   CONTINUE
   CALL CURVE (Y, Z, 1, -1)
20 CONTINUE
32 CALL RESET ('BLNKS')
CALL LEGEND (IPAX, 10, 7., 6.)
CALL ENOPL (0)
CALL DONEPL
END
Example 5

IMPACT WINDOW

DEPTH in INCHES

- .000 - .0025
- .0025 - .0125
- .0125 - .2500
- .2500 - .5000
- .5000 - .7500
- .7500 - 1.000
- 1.000 - 2.000
- 2.000 - 3.000
- 3.000 - 4.000
- > 4.000
PROGRAM EXAMPLE6
DIMENSION X(50),Y(50),IPAK(1000)
OPEN (1,FILE='DAT',STATUS='OLD')
CALL COMPRS
CALL BGNPL (0)
CALL SETCLR ('BLUE')
CALL HEIGHT (.2)
CALL PAGE (12.0, 11.0)
CALL INTAXS
CALL YAXANG (0.)
CALL TITLE ('FUNCTIONS', 100, 'X - AXISS', 100, 'Y - AXISS', 100, 9., 7.)
CALL SETCLR ('GREEN')
CALL GRAF (-3., 1., 3., -4., 1., 10.)
CALL SETCLR ('GREEN')
CALL LINES ('INPUT POINTS$', IPAK, 1)
CALL LINES (' DEGREE 1$', IPAK, 2)
CALL LINES (' DEGREE 2$', IPAK, 3)
CALL LINES (' DEGREE 3$', IPAK, 4)
CALL LINES (' DEGREE 4$', IPAK, 5)
CALL LINES (' DEGREE 5$', IPAK, 6)
CALL LINES (' DEGREE 6$', IPAK, 7)
CALL MYLEGN ('POLY DEGREES', 100)
XR = XLEGND (IPAK, 7) + .3
YR = YLEGND (IPAK, 7) + .3
CALL BLREC (7.-.3, 6.-.3, XR+.3, YR+.3, .02)
DO 10 1=1,13
   READ (1,*) X(1),Y(1)
10  CONTINUE
CALL SPCMOD
CALL LEGLIN
CALL CURVE (X, Y, 13, 1)
DO 30 J=1,6
   DO 20 1=1,49
      READ (1,*) X(1),Y(1)
20  CONTINUE
30  CONTINUE
CALL BLMOVE (XR+.9, 0.)
CALL LEGEND (IPAK, 7, 7., 6.)
CALL ENDPL (0)
CALL DONEPL
END
SUBROUTINE MYSPEC (J)
  IF (J .EQ. 1) CALL SETCLR ('RED')
  IF (J .EQ. 2) CALL SETCLR ('MAGENTA')
  IF (J .EQ. 3) CALL SETCLR ('BLUE')
  IF (J .EQ. 4) CALL SETCLR ('CYAN')
  IF (J .EQ. 5) CALL SETCLR ('YELLOW')
RETURN
END
Example 6

FUNCTION

POLY DEGREE
- Input Points
- Degree 1
- Degree 2
- Degree 3
- Degree 4
- Degree 5
- Degree 6

Y - Axis

X - Axis
Interpolation

To use interpolation routines provided by DISSPLA:

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL SPLINE</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL PSPLIN</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL POLY3</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL POLY5</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL LINEAR</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL PARA3</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL PARA5</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL STEP</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL BARS (BARWTH)</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
</tbody>
</table>

For smoothing: (Supply error weighting factors in blank common YDLARAY)

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL SMOOTH</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL PSMTH</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
</tbody>
</table>

To change line texture:

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL DOT</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL DASH</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL CHNDOT</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL CHNDSH</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
<tr>
<td>CALL RESET ('DOT') {SOLID}</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
</tbody>
</table>

To construct your own texture:

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL MRSCOD (TLENG, NMRKSP, RATRAY)</td>
<td>1,2,3 p/s</td>
<td></td>
</tr>
</tbody>
</table>
**PROGRAM EXAMPLE7**

**DIMENSION** 

\( X(13), Y(13), RATRAY(6) \)

**DATA** 

\( X/-3., -2.5, -2., -1.5, -1., -0.5, 0., 1, 1.5, 2., 2.5, 3./ \)

\( Y/-1., 4.5, -3., -6., -2., 3., 3., 8., -2., 5., -1., 5., -3./ \)

**DATA** 

\( RATRAY /5., 2., 2., 2., 7., 2./ \)

**CALL** 

**COMPRS**

**CALL** 

**BGNPL** (0)

**CALL** 

**HEIGHT** (1.)

**CALL** 

**SCMPLX**

**CALL** 

**PAGE** (24.0, 18.0)

**DO** 30 \( J=1, 6 \)

**CALL** 

**PHYSOR** (1., \( \text{FLOAT}(J-1)*3. \))

**CALL** 

**AREA2D** (15., 3.)

**CALL** 

**SETCLR** ('GREEN')

**CALL** 

**GRAF** (-3., 1., 3., -4., 2., 10.)

**CALL** 

**FRAME**

**CALL** 

**MESSAG** ('____________________5', 100, 15., 0.19)

**IF** ( \( J \text{ .EQ. } 1 \) ) **THEN**

**CALL** 

**LINEAR**

**CALL** 

**SETCLR** ('RED')

**CALL** 

**MESSAG** ('LINEARS', 100, 16., 1.)

**ELSE IF** ( \( J \text{ .EQ. } 2 \) ) **THEN**

**CALL** 

**POLY3**

**CALL** 

**SETCLR** ('BLUE')

**CALL** 

**MESSAG** ('POLY3S', 100, 16., 1.)

**CALL** 

**DOT**

**ELSE IF** ( \( J \text{ .EQ. } 3 \) ) **THEN**

**CALL** 

**STEP**

**CALL** 

**SETCLR** ('YELLOW')

**CALL** 

**MESSAG** ('STEPS', 100, 16., 1.)

**CALL** 

**DASH**

**ELSE IF** ( \( J \text{ .EQ. } 4 \) ) **THEN**

**CALL** 

**SPLINE**

**CALL** 

**SETCLR** ('MAGENTA')

**CALL** 

**MESSAG** ('SPLINES', 100, 16., 1.)

**CALL** 

**CHNDOT**

**ELSE IF** ( \( J \text{ .EQ. } 5 \) ) **THEN**

**CALL** 

**BARS** (1.)

**CALL** 

**SETCLR** ('CYAN')

**CALL** 

**MESSAG** ('BARS', 100, 16., 1.)

**CALL** 

**CHNDSH**

**ELSE IF** ( \( J \text{ .EQ. } 6 \) ) **THEN**

**CALL** 

**POLY5**

**CALL** 

**SETCLR** ('YELLOW')

**CALL** 

**MESSAG** ('POLY5S', 100, 16., 1.)

**CALL** 

**MRSCOD** (2., 6., \( RATRAY \))

**ENDIF**

**CALL** 

**CURVE** ( \( X, Y, 13, 1 \))

**CALL** 

**ENDEGR** (0)

**CONTINUE**

**CALL** 

**ENDPL** (0)

**CALL** 

**DONEPL**

**END**
Example 2

<table>
<thead>
<tr>
<th></th>
<th>POLY5</th>
<th>BARS</th>
<th>SPLINE</th>
<th>STEP</th>
<th>POLY3</th>
<th>LINEAR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example 8 shows a polar plot of flows around a prop. The data is read in from two data files containing angles and magnitudes of the flows. A vector will be drawn for each flow, with the length being the magnitude, the direction being the angle. The data did have to be adjusted to convert to the polar coordinate system. Notice that subroutine TITLE is used even though it is an obsolete routine it is still supported by DISSPLA. In all calls to MESSAG, the position (XVAL, YVAL) is given in polar coordinates (i.e., ANGLE, MAGNITUDE). In the call to POLAR you determine whether angles are in radians or degrees.
This page intentionally left blank
PROGRAM EXAMPLE
OPEN (1, FILE='TUBE1.DAT', STATUS='OLD')
OPEN (2, FILE='TUBE2.DAT', STATUS='OLD')
CALL COMPRS
CALL SETDEV (9, 9)
CALL BGNPL (1)
CALL PAGE (11., 1.)
CALL SETCLR ('BLUE')
CALL TITLE ('', 1, 'RADIUS', 6, 1H, 1, 10.5, 10.5)
CALL POLAR (3.14159/180., .25, 5.25, 5.25)
CALL HEIGHT (0.2)
DO 105, K=1,2
IF (K.EQ.1) THEN
  CALL SETCLR ('RED')
  READ (1, '(5X, 18X, I3)') INUM
  CALL RLMESS (' TUBE = ', 8.60, 1.3)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
  CALL RLMESS (' TUBE = ', 8.135, 1.6)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
  CALL RLMESS (' TUBE = ', 8.220, 1.6)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
  CALL RLMESS (' TUBE = ', 8.310, 1.35)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
END IF
IF (K.EQ.2) THEN
  CALL SETCLR ('BLUE')
  READ (2, '(5X, 18X, I3)') INUM
  CALL RLMESS (' TUBE = ', 8.50, 1.3)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
  CALL RLMESS (' TUBE = ', 8.140, 1.5)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
  CALL RLMESS (' TUBE = ', 8.220, 1.8)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
  CALL RLMESS (' TUBE = ', 8.290, 1.3)
  CALL INTNO (INUM, 'ABUT', 'ABUT')
END IF
END
DO 50 I=1,2500
IF (K.EQ.1) READ (1, 11, END=105) TH, R1, TH1
IF (K.EQ.2) READ (2, 11, END=999) TH, R1, TH1
IF (K.EQ.1) R = 1.25
IF (K.EQ.2) R = .75
X = R * COSD (TH)
Y = R * SIND (TH)
X1 = R1 * COSD (180. + TH - TH1)
Y1 = R1 * SIND (180. + TH - TH1)
XNEW = X + X1
YNEW = Y + Y1
RNEW = SQRT ((XNEW**2) + (YNEW**2))
THNEW = ATAND (YNEW/XNEW)
IF ((TH.GT.0).AND. (TH.LE.90)) THEN
  CALL SETDEV (9, 9)
END IF
IF ((TH.GT.180).AND. (TH.LE.270)) THEN
  ANG = 2.
  ! X = R1*COSD(TH-TH1-ANG*(90.))
  ! Y = R1*SIND(TH-TH1-ANG*(90.))
END IF
IF ((TH.GT.180).AND. (TH.LE.270)) THEN
  CALL SETCLR ('GREEN')
  IF (X1.LT.0).AND. (Y1.LT.0) THEN
    CALL GRID (1, 1)
    Y1 = -Y1
    X1 = -X1
  END IF
  CALL POLAR (3.14159/180., .25, 5.25, 5.25)
  CALL HEIGHT (0.2)
  CALL TITLE ('', 1, 'RADIUS', 6, 1H, 1, 10.5, 10.5)
  CALL BGNPL (1)
END IF
END
Post-Processor

When COMPRS is used as the device, DISSPLA creates a device independent META file (called DISPLOT.DAT on VAX and PLFILE on the CYBERS). When you run the post processor, you can choose any device and make certain modifications. A plot can be scaled up or down, certain ones drawn, or a small portion of the plot "zoomed" in on.

The following command used on the preceding example would produce the following graph:

MODI = 1( WINDOW = UPPER( 5.75), LOWER( 0.75),
RIGHT( 5.75), LEFT ( 0.75) * SIZE = 11,8)

NOTE: The graph cannot be changed (i.e., viewpoint), you can only work with the plot as it exists!
Example 9

TUBE = 1

BE = 2
3-Dimensional Plotting

Three-dimensional plotting in DISSPLA is similar to two-dimensional plotting except that some subroutines are changed to accommodate the Z coordinate arguments. There are also some additional routines to cope with 3-D plotting properties. The user wishing to use 3-D is advised to read the User's Manual.

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL VOLM3D (X3AXIS, Y3AXIS, Z3AXIS)</td>
<td>2</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL VUABS (XVU, YVU, ZVU)</td>
<td>2</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL VUANG (PHI, THETA, RADIUS)</td>
<td>2</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL GRAF3D (X3ORIG, X3STEP, X3MAX, Y3ORIG, Y3STEP, Y3MAX, Z3ORIG, Z3STEP, Z3MAX)</td>
<td>2, 3</td>
<td>Same</td>
</tr>
<tr>
<td>CALL CURV3D (XARAY, YARAY, ZARAY, N, I)</td>
<td>3</td>
<td>Same</td>
</tr>
</tbody>
</table>

N - Number of points to be plotted

1 - < 0  Symbol every i-th mark not connected
= 0  No symbols just a line
> 0  Symbol every i-th mark and connected

To draw 3-D Vector:

CALL VECTR3 (XFROM, YFROM, ZFROM, XTO, YTO, ZTO, IVEC) 2, 3 Same

CALL RLVEC3 (XROM, YFROM, ZFROM, XTO, YTO, ZTO, IVEC) 2, 3 Same
Example #10 shows a simple 3-D plot of a conic section. Notice that along with AREA2D, AXES3D is called to define the 3-D work area. It is defined in "absolute units", so that VUABS can define the view angle relative to the work area. Four subplots will be put onto the same page, each one using a different viewpoint. The X, Y, and Z arrays are filled with the coordinates to be plotted using the equation of the conic.
PROGRAM EXAMPLE 10

DIMENSION X(500), Y(500), Z(500), ZX(250)

CALL COMPRS
CALL BGNPL (0)
CALL PAGE (14., 11.)
CALL HEIGHT (.3)
CALL INTAXS
CALL ZAXANG (90.)
DO 100 IRUNS=1,4
    CALL SETCLR ('BLUE')
    CALL ENDGR (0)
    IF (IRUNS.EQ.1) THEN
        CALL PHYSOR (.5, .5)
        CALL AREA2D (6., 5.)
        CALL VUABS (3., -30., 3.)
    ELSE IF (IRUNS.EQ.2) THEN
        CALL PHYSOR (7.5, .5)
        CALL AREA2D (6., 5.)
        CALL VUABS (30., 30., 30.)
    ELSE IF (IRUNS.EQ.3) THEN
        CALL PHYSOR (7.5, 5.5)
        CALL AREA2D (6., 5.)
        CALL VUABS (-30., -30., -30.)
    ELSE IF (IRUNS.EQ.4) THEN
        CALL PHYSOR (7.5, 5.5)
        CALL AREA2D (6., 5.)
        CALL VUABS (30., 30., 30.)
    ENDIF
CALL AXES3D ('X', 'Y', 'Z', 1, 3., 3., 3.)
CALL GRAF3D (-30., 15., 30., -30., 15., 30., -30., 15., 30.)
DO 10 1=1,60
    ICOUNT = 1
    DO 20 J=1,60
        ZZ = FLOAT (1-J30)
        YY = FLOAT (J-30)
        XX = (((ZZ**2)/16) - (YY**2)/4) + 1)*4
        IF (XX .GE. 0) THEN
            X (ICOUNT) = SQRT (XX)
            Y (ICOUNT) = YY
            Z (ICOUNT) = ZZ
            ZX (ICOUNT) = -SQRT (XX)
            ICOUNT = ICOUNT+1
        ENDIF
    CONTINUE
    ICOUNT = ICOUNT-1
    CALL SETCLR ('GREEN')
    DO 500 IX=1,ICOUNT
        Z (ICOUNT+IX) = Z (ICOUNT+1-IX)
        X (ICOUNT+IX) = ZX (ICOUNT+1-IX)
        Y (ICOUNT+IX) = Y (ICOUNT+1-IX)
    CONTINUE
Example 10
Example #11 shows how to plot a surface defined by a function of two variables. Once the work box is set up VUANGL is called to define the viewpoint. This has the same effect as VUABS, only it uses spherical coordinates. The equation to be plotted is put in an EXTERNAL function with two arguments. The call to SURFUN contains the name of the function and will automatically compute the points to be plotted. The figure is plotted 15 times, each from a different viewpoint to simulate rotation.

Example #12 shows a single surface plot, the call to VUABS will draw the surface as if you were looking at it from (-10, 4, 20). ZAXANG(0.) will label the Z-axis horizontally the same way YAXANG works in 2-D.

NOTE: When defining the viewpoint, you must draw the object from the outside looking in. DISSPLA will print an error message if you are within the work box (i.e., looking from the inside out).
This page intentionally left blank.
PROGRAM EXAMPLE11
EXTERNAL PYRR, PYRR1
CALL COMPRS
CALL BGNPL (0)
THETA = 0
PHI = -180.
CALL PAGE (35.5, 30.)
DO 20 I=1,5
   DO 30 J=1,3
      CALL SETCLR ('BLUE')
      CALL PHYSOR (FLOAT(I-1)*(35.5/5.), FLOAT(J)*10-10)
      CALL AREA2D (35.5/5., 30./3.)
      CALL FRAME
      CALL BLSUR
      CALL AXES3D (0, 0, 0, 0, 0, 5., 5., 5., 5.)
      CALL VUANGL (PHI, THETA, 20.)
      CALL GRAF3D (-1000., 500., 1000., -1000., 500., 1000., -1000.,
                    500., 1000.)
      CALL SETCLR ('CYAN')
      CALL SURFUN (PYRR, 1., 100., 1., 100., WORK)
      CALL SETCLR ('RED')
      CALL SURFUN (PYRR1, 1., 100., 1., 100., WORK)
      CALL ENDGR (0)
      IF (J.EQ.2) THEN
         THETA = 90. - I*10.
      ELSE
         PHI = PHI + 90.
      ENDIF
   CONTINUE
PHI = -180.
THETA = 1 * 10.
20 CONTINUE
CALL ENDPLO (0)
CALL DONEPL
END

FUNCTION PYRR(X,Y)
PYRR = SQRT(ABS((3*Y**2-16*X)/12))
RETURN
END

FUNCTION PYRR1(X,Y)
PYRR1 = -SQRT(ABS((3*Y**2-16*X)/12))
RETURN
END
PROGRAM EXAMPLE12

EXTERNAL PYRR

CALL COMPRS

CALL BGNPL (0)

CALL PAGE (14., 11.)

CALL HEIGHT (.3)

CALL INTAXS

CALL ZAXANG (90.)

CALL SETCLR ('BLUE')

CALL AREA2D (10., 8.)

CALL HEADIN ('3-D SURFACE PLOTS', 100, 1.2, 1)

CALL AXES3D ('X', 1, 'Y', 1, 'Z', 1, 5., 5., 5.)

CALL VUABS (-10., 4., 20.)

CALL GRAF3D (-2., 1., 2., -2., 1., 2., 0., 1., 2.)

CALL SETCLR ('RED')

CALL SURFUN (PYRR, 2, .1, 2, .1, WORK)

CALL ENDPL (0)

CALL DONEPL

END

FUNCTION PYRR(X,Y)

PYRR = (X**2+2*Y**2)*EXP(1-X**2-Y**2)

RETURN

END
Example 12

3-D Surface Plot
**Graffiti Plots**

Graffiti plots allow the user to project a 2-D plot onto a plane in 3-D. The call below defines the plane by giving three points that are on the plane.

```call
GRFITI (XLCORN, YLCORN, ZLCORN,
        XBASEX, YBASEX, ZBASEX,
        XOTHER, YOTHER, ZOTHER)
```

- **XLCORN, YLCORN, ZLCORN** - coordinates of the lower left corner of the plane in absolute workbox units.
- **XBASEX, YBASEX, ZBASEX** - coordinates of a point on the X-axis in absolute workbox units.
- **XOTHER, YOTHER, ZOTHER** - coordinates of a point in the plane above the X-axis in absolute workbox units.

Example #13 shows a 3-D graffiti plot. It is important to note that when drawing a graffiti plot, the 2-D plot must be finished (CALL END3GR) before 3-D plotting or another 2-D plot can be started. The workbox is defined to be 5 X 5 X 5, the first graffiti plot defines the 2-D plane (0,5,0) (5,5,0) (2,5,2) which is the Z-X plane when Y is equal to 5. A grid is drawn on the plane and a message written before it is ended by END3GR. Then the second plane is defined, the Y-Z plane when X is 5. A grid is again drawn on the plane and the message written before it is ended by END3GR. Finally the third plane is defined, the X-Y plane when Z is zero, and the same events occur. After the final call to END3GR the 3-D plotting can begin.

Example #14 shows a graffiti plot with actual graphs.
PROGRAM EXAMPLE 13
CALL TK41 (4105)
CALL BGNPL (0)
CALL PAGE (14., 11.)
CALL SETCLR ('BLUE')
CALL PHYSOR (.5, .5)
CALL HEIGHT (.25)
CALL INTAXS
CALL AREA2D (13., 10.)
CALL SETCLR ('RED')
CALL FRAME
CALL AXES3D ('X - AXIS', 8, 'Y - AXIS', 8, 'Z - AXIS', 8, 5, 5, 5.)
CALL VUABS (-10., -10., 10.)
CALL GRAF3D (-150., 50., 150., -150., 50., 150., -150., 50., 150.)
CALL GRFITI (0., 5., 0., 5., 0., 2., 5., 2.)
    CALL AREA2D (5., 5.)
    CALL GRAF (-150., 10., 150., -150., 10., 150.)
    CALL SETCLR ('GREEN')
    CALL MESSAG ('THIS IS THE $S', 100, 1., 2.5)
    CALL MESSAG ('Z - X PLANES', 100, 1., 2.)
    CALL BLREC (.7, 1.7, 3, 1.5, .08)
    CALL GRID (1, 1)
    CALL END3GR (0)
CALL GRFITI (5., 5., 0., 5., 0., 0., 5., 2., 2.)
    CALL AREA2D (5., 5.)
    CALL GRAF (-150., 10., 150., -150., 10., 150.)
    CALL SETCLR ('YELLOW')
    CALL MESSAG ('THIS IS THE $S', 100, 1., 2.5)
    CALL MESSAG ('Y - Z PLANES', 100, 1., 2.)
    CALL BLREC (.7, 1.7, 3, 1.5, .08)
    CALL GRID (1, 1)
    CALL END3GR (0)
CALL GRFITI (0., 0., 0., 5., 0., 0., 0., 5., 0.)
    CALL AREA2D (5., 5.)
    CALL GRAF (-150., 10., 150., -150., 10., 150.)
    CALL SETCLR ('BLUE')
    CALL MESSAG ('THIS IS THE $S', 100, 1., 2.5)
    CALL MESSAG ('X - Y PLANES', 100, 1., 2.)
    CALL BLREC (.7, 1.7, 3, 1.5, .08)
    CALL GRID (1, 1)
    CALL END3GR (0)
CALL SETCLR ('CYAN')
CALL BOX3D
CALL ENDPL (0)
CALL DONEPL END
Example 13

This is the X-Y Plane

This is the Y-Z Plane

This is the Z-X Plane
Example 14

USE OF GRAFITI PLOTS
Character Sets

DISSPLA contains numerous alphabets and character styles. Consult the User's Manual for examples of each.

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Level at call</th>
<th>Level after call</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL CARTOG</td>
<td>1,2,3</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL SIMPLX</td>
<td>1,2,3</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL SCMPLX</td>
<td>1,2,3</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL COMPLX</td>
<td>1,2,3</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL DUPLX</td>
<td>1,2,3</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL TRIPLX</td>
<td>1,2,3</td>
<td>p/s</td>
</tr>
<tr>
<td>CALL GOTHIC</td>
<td>1,2,3</td>
<td>p/s</td>
</tr>
</tbody>
</table>

Then to choose an alphabet:

CALL BASALF (LALPHA) 1,2,3 p/s

where LALPHA is:

"STANDARD" "L/CSTD" "GREEK" "L/CGREEK" "RUSSIAN" "L/CRUSSIAN" "HEBREW"

"ITALIC" "L/CITALIC" "SCRIPT" "SPECIAL" "MATHEMATIC" "INSTRUCTION"

Example #15 shows how to use several alphabets, and how to change from one to another. The user must call MXIALF with the alphabet chosen and a character to represent it. At this point, any time you write out text, this character causes DISSPLA to switch to the alphabet it represents. In each call to MESSAG, these special characters are not printed but the words appear in different alphabets. Each string must end in the BASALF to use the "$" string counter feature.
PROGRAM EXAMPLE15
CALL COMPRS
CALL SETDEV (1, 1)
CALL BGNPL (0)
CALL TRIPLEX
CALL BASALF ('STANDARD')
CALL MX1ALF ('STANDARD', '+'
CALL MX2ALF ('L/GREEK', '#')
CALL MX3ALF ('RUSSIAN', '/')
CALL MX4ALF ('SCRIPT', '&')
CALL MX5ALF ('GREEK', '$')
CALL MX6ALF ('ITALIC', 'I')
CALL SETEND (';', 1)
CALL HEIGHT (.5)
CALL PAGE (16., 8.5)
CALL PHYSOR (0., 0.)
CALL AREA2D (16., 8.5)
CALL SETCLR ('BLUE')
CALL MESSAG ('THIS IS AN !EXAMPLE+ USING DIFFERENT;', 100, .5, 7.5)
CALL MESSAG ('ALPHABETS, &IT IS QUITE EASY IT0 USE AND+;', 100, .5, 6.5)
CALL MESSAG ('THE 1ALPHABET CAN +BE CHANGED &AT ANY TIME+;', 100, .5, 5.5)
CALL MESSAG ('NOTICE IT THAT EACH STRING +ENDS IN THE +BASALF;', 100, .5, 4.5)
CALL SETCLR ('RED')
CALL MESSAG ('SOME MORE EXAMPLES:;', 100, 2., 3.5)
CALL MESSAG ('RUSSIAN -/RUSSIAN+ L/GREEK -#ABCD8F+;', 100, .5, 2.5)
CALL MESSAG ('GREEK - $ABCDEFGH +ITALIC -IITALIC+;', 100, .5, 1.5)
CALL SETCLR ('BLUE')
CALL MESSAG ('THE ISTRINGS+ ARE PLACED IN INCHES FROM ORIGIN;', 100, .1, .5)
999 CALL ENDPL (0)
CALL DONEPL
CLOSE (1, STATUS='DELETE')
STOP
END
Example 15

This is an example using different Alphabets, it is quite easy to use and the ALPHABET can be changed at any time. Notice that each string ends in the BASALF.

Some more examples:

RUSSIAN –ПИОССИАН L/GREEK –αβηδεφ
GREEK – ΑΒΗΔΕΦΓΧ ITALIC –ITALIC

The strings are placed in inches from origin.
This next example shows the power of the instruction alphabet. The user can choose any alphabet, spacing, height or font through commands similar to those of multiple alphabets. The commands are enclosed in parentheses and operate only on the line they appear in. Below are the commands that were used for example 16, for a complete list refer to the reference manual or page B-24 of the Pocket Guide.

Pi - set the i-th tab at the current position horizontally. (i<20)

Gi - move back to the i-th tab previously set by P. (i<20)

Er - move up (superscript) from base line a distance r.

Lr - move down (subscript) from base line a distance r.

Yr - skew character by factor r=x/y.

Ui - underline once from i-th P tab to current position.

Di - double underline from i-th P tab to current position.

Fi - Font changed to corresponding style:

0 - Default
1 - CARTOG
2 - SIMPLX
3 - SCMPLX
4 - COMPLX
5 - DUPLX
6 - TRIPLX
7 - GOTHIC
8 - FUTURA
9 - SERIF
10 - FASHON
11 - LOGO1
12 - SWISSL
13 - SWISSM
14 - SWISSB

 NOTE: If a shaded font is used it must be called prior to use.
Mi - Set character set to specified alphabet:

0 - STANDARD  
1 - L/CSTD  
2 - ITALIC  
3 - L/CITALIC  
4 - SCRIPT  
5 - L/CSCRIPT  
6 - GREEK  
7 - L/CGREEK  
8 - RUSSIAN  
9 - L/CRUSSIAN  
10 - SPECIAL  
11 - MATHEMATIC  
12 - HEBREW  
13 - ALFABET

X - as an argument, resets instruction to default value.

Example #16 writes out text in various styles using the instruction alphabet. Text can be slanted, underlined and at different heights. Also, the mathematical formula is written out using the mathematical character set (see manual). If time is taken to learn the instruction alphabet, the user can benefit by having every style of character DISSPLA can produce available to him throughout his plot.
NOTE: in the call to MESSAG, when ABUT is given as the position for the X and Y value, the message is placed where the previous message left off.
This is an example of the instruction alphabet MATHEMATICS

\[
\lim_{\delta \xi \to 0} \sum_{i=1}^{n} \left[ \int_{\Lambda_{i}(Y_{i})}^{\Lambda_{i}(Y_{i})} F\left(X, Y_{i}\right) \delta X \right] \nabla_{i}Y
\]
Initial Distribution

Copies:

12 Director
Defense Technical Information Center (DTIC)
Cameron Station
Alexandria, Virginia 23314

Center Distribution

Copies:

1 18/1809 Schuman, Dr. C. M.
1 1805 Cuthill, E. H.
2 1809S
1 182 Camara, A. W.
1 184 Schot, J. W.
1 185 Schaffran, R.
1 187 Zubkoff, M. J.
1 189 Gray, G. R.
1 189.2
1 189.3 Morris, J.
1 1892.1 Strickland, J. D.
40 1892.1 Brady, K. G.
10 1892.2 Sommer, D. V.
1 1892.3 Minor, L. R.
1 1894
1 1896 Glover, A.
1 1896.2 Dennis, L.
1 522 TIC (C)
1 522.2 TIC (A)
1 93 Patent Counsel
DTNSRDC issues three types of reports:

1. **DTNSRDC reports, a formal series**, contain information of permanent technical value. They carry a consecutive numerical identification regardless of their classification or the originating department.

2. **Departmental reports, a semiformal series**, contain information of a preliminary, temporary, or proprietary nature or of limited interest or significance. They carry a departmental alphanumerical identification.

3. **Technical memoranda, an informal series**, contain technical documentation of limited use and interest. They are primarily working papers intended for internal use. They carry an identifying number which indicates their type and the numerical code of the originating department. Any distribution outside DTNSRDC must be approved by the head of the originating department on a case-by-case basis.
END
12-86
DTIC