Galaxy Advanced Engineering, Inc.
P.O. BOX 614
Burlingame, California 94011
Tel: (650)-740-3244
Fax: (650)-347-4234
E-mail BahmanZ@AOL.COM

UGL - GRAPHICS LIBRARY

A Scientific Graphics Subroutine Library for Micro and Mini - Computers

UGL - GRAPHICS™ is developed as scientific graphic subroutine library code for Micro and Mini - Computers at **Galaxy Advanced Engineering, Inc**. **(GAE)**. The code provides a powerful scientific graphics capability on your Windows/PC, Linux, Alpha/Open VMS, HP/UX, SUN/OS, SGI as well as VAX/VMS computers.

For many years, engineering and scientific applications have been executed on mainframe or mid-size computers. Very large investments have been made in this type of software and they're supporting utilities. The changing computer environments are moving many of these applications to the desktop, either with UNIX workstations or more powerful personal computers (PC). UGL has been developed to be a part of this process and provide an orderly migration path by providing similar graphics interfaces in both environments. UGL also provides expanded capabilities, which allow extensions of existing graphics without expensive up front, development.

The **Universal Graphics Library (UGL)** is a high-level FORTRAN scientific graphics library designed to support applications in engineering, scientific, and business environments. UGL's integrated routines allow programmers to generate graphics, charts, surfaces, contours and 2- and 3-dimensional designs, including logarithmic axes, polar coordinates, 3-D mesh-and-line plots, and error-bar charts. Full color features are available as well.

The **UGL - GRAPHICS™** is a high level **FORTRAN** graphics subroutine library for interactive or batch data representation application in engineering, scientific, and business environments. The code is a fully functional stand-alone graphics package and is a Device Independent Graphics Library for any existing computer and their related operating system and can interface and supports the most well-known graphics devices in the market. The package comes with all necessary device drivers. The **PC** version of the code is compatible with **LAHEY FORTRAN** compiler **(F77L-EM32)** as well as **LF90**, **Microsoft FORTRAN PowerStation** Version 1.x and 4.x. and **Compaq Visual FORTRAN** compiler version 5.x and 6.x. The code can run under **Windows 95/98/2000/XP** and **NT** with its version of **LF90**, **PowerStation** and **Compaq Visual FORTRAN**. This version provides **OLE** capability for porting your graphics output to another windows application such as your word processing by copying your output to clip board and past it to your applications. The code can be compiled with other **FORTRAN** compiler that has virtual memory capability such as **Salford**. There also exist **VAX/VMS**, **Alpha/AXP**, **HP/UX**, **SUN/OS**, and **LINUX**. We are presently working on version of the code to make it compatible with **Absoft** Compiler for Windows and Linux.

UGL simulates **CA-DISSPLA**, **PLOTl0**, **DIGLIB**, **NASDIG**, **PLOT88**, **GKS 0A** level (limited base), **NCAR**, and **CalComp** graphics routines for portability purposes. The most common subset of DISSPLA routines is supported directly. In the **UGL/DISSPLA** compatibility mode, most FORTRAN77/90 and 95 programs can be run with only minor modifications related to different output devices. Approximately **90% DISSPLA** calls are currently pressed and supported by UGL graphics library. UGL is capable of simulating any other calls of **DISSPLA** routines if the users do not find them among our existing routines. (Please call us with your request on your desired routines).

The PC version of the code called **UGL/PC** brings mainframe capacity and result to your desktop microcomputer. It includes all of the popular mainframe extension. As a device independent graphics library for microprocessor, it provides a powerful graphics capability in scientific, engineering and business application.

The PC version of UGL operating on the window operating system is compatible with **Compaq Visual FORTRAN** compiler, a 32-bit compiler, and versions 5.x and6.x. This compiler with the associated memory manager, debuggers, and related utilities provide virtually similar capabilities as are found on the main frames and mid-size computers. MS-DOS memory limitations are bypassed so that any calculation that could be done on the main frame can be now done on PC at greatly reduced costs.

The same version of UGL exists for **DIGITAL/UNIX** workstations and **Alpha/NT** and **Open VMS**. For other platforms and operating system call Galaxy Applied Engineering, Inc. The main body of **UGL** is identical on all systems. There are some variations in output devices that are routinely supported. For example, on UNIX systems, an X-Window driver is normally the primary interface to the user (with a terminal interface for multi-user systems). On a PC, a VGA/SVGA driver replaces this interface.

UGL - Graphic's™ integrated routines allow programmers to generate graphics, chart, surfaces, contours and 2- and 3-dimensional design, including logarithmic axes, polar coordinate, 3-D mesh-and-line plots, and error-bar charts. The graphics library may be ordered to work without the math-coprocessor on request, but it will perform distinguishably slower.

UGL - GRAPHICS™ includes a set **FORTRAN** callable routine to aid you in plotting characters and character strings. The high-level applications include in **UGL - GRAPHICS™** are: X-Y Graphics, X-Y-Z graphics, Contours, Histograms, Scatter Diagrams, Streamlines, Vector Graphics 3-D solids, 3-D Surfaces, Cartographic Maps, Map Data Overlays and World Map plots as well blanking capability plots.

UGL - GRAPHICS™ was developed to provide a migration path with a **CA-DISSPLA, GKS, PLOT10, CalComp** and **DIGLIB** (from Lawrence Livermore Lab.) graphics interface in the **PC** and **VAX** as well as **UNIX** environment and provides the same high graphics standards found on the main frames using the above graphics libraries. The most common subset of **CA-DISSPLA** routine (more than 900) and rest of the mentioned graphics libraries are supported directly and any particular ones may be provided upon request. If the users have an existing code using any of these graphics library routines within their code do not have to change their calls. The bridge that are built with **UGL - GRAPHICS™** library will distinguish these routine s and maps its own routine against the direct porting of the user code to its new environment supported by **UGL - GRAPHICS™**.

UGL includes a set of FORTRAN callable routines to aid you in plotting characters and character strings. It implements an **ANSI/ISO** standard Graphical Kernel System (**GKS**) platform; an ANSI/ISO standard Computer Graphics Metafile (**CGM**) output format; and applications that overlay any ANSI/ISO standard GKS, which provides a device language and resolution independent platform for graphics generation. The high-level applications included in **UGL** are:

- **Multiple plots type:**

  - X-Y Graphics.
  - -X-Y-Z Graphics.
  - Contours.
  - Histograms.
  - Scatter Diagrams.
  - Streamlines.
  - Vector Graphs.
  - 2-D and 3-D graphs and plots.
  - Calendar charts.
  - 3-D Solids.
  - 3-D Surfaces.
  - Cartographic Maps.
  - Map Data Overlays.

- Extensive parameter support and control

  - Area blanking control.
  - Area fills with fills pattern control.
  - Curve and frame thickness control.
  - Color support.
  - Multiple line style, including user-defined style.
  - Spline and polynomial line interpolation.

- Both high and low-resolution world mapping including political boundaries.
- Legend blocks and text blocks.
- Multiple text fonts.
- Generation of plots for word processing packages such as Microsoft word and WordPerfect.

The **UGL** graphics library is a general-purpose graphics package that can be used to generate plots for most digital computer based applications. To generate a plot using UGL, the user's application program calls the appropriate **UGL** subprograms. These subprograms then cause the plot to be printed, plotted, or stored as a file image.

UGL is supplied as a set of re-locatable library files. To generate a plot using **UGL**, the user's **FORTRAN77/90** and **95** application program calls the appropriate **UGL** programs. At link time, the requested routines are collected from the library files and loaded with the user's code. These subprograms then cause the plot to be generated and displayed, printed, or stored as a file image.

UGL can generate graphics metafiles, which can be saved for later viewing without rerunning the user program. The appropriate POST processors for these metafiles are supplied. There is

normally one POST for each format. For example, **POSTX11** is for **X-Windows** release 5, version 11 on **UNIX**. **POSTVGA** is for Super VGA (VESA) display on a PC.

UGL routines are callable within your C program on any given computing platform as well.

There are two versions of link procedures. The first is for a pure **UGL** code, which uses only calls, found in this manual. The second is a **DISSPLA** compatibility mode linker. In this mode many DISSPLA calls are recognized and translated internally to the associated **UGL** routine. The additional routines that are processed in this manner are not documented in detail in this manual.

A sample printer queuing batch file is included. When the user tells **UGL** to send a file to a printer, UGL makes an operating system call to tell the system to run a special batch procedure. It is the function of this procedure to know the correct commands to complete this task

UGL permits almost complete host and device independence for user application software. UGL achieves host independent via strict compatibility with the America National Standard Institute (ANSI) standard X3.0-1978, often referred to as FORTRAN 77. This greatly reduces application code modifications when porting programs. Internally, UGL uses "Z" and extension to avoid name conflicts with user code. The "Z" is used with routine names and that is used with common blocks. UGL, with the DISSPLA compatibility option, contains nearly 1000 routines; so name conflicts with user code can be a real problem. The extensions reduce the number by a factor of two.

The following examples are provided with their FORTRAN source codes to show the simplicity of Universal Graphics Language.

Example 1

```
C ****************************************************************
C
      PROGRAM SHUTLE
C
C     This program demonstrates a complex linear X-Y plot
C     and shaded fonts.
C     Original Source: NASA
C     Modified: Galaxy Advanced Engineering, Inc.   (GAE)
C
      DIMENSION X(7),Y(7)
      COMMON /R2D2/ KNT
C     set data for x, y
      DATA X/23.,16.,12.,10.,8.,4.,0./
      DATA Y/240000.,230000.,170000.,125000.,70000.,20000.,5000./
C
C ****************************************************************
C
C     Step 1 - initialize output device
C     Allow user to choose a device interactively
      KTYPE=0
      CALL DEVICE (KTYPE,XPAGE,YPAGE)
C
C ****************************************************************
C
C     Step 2 - define page size
C     none
C
C ****************************************************************
C
C     Step 3 - define plot axes and sub-plot area
C     Set length of X and Y axis in inches for a square plot
C     In UGL, xpage is always = 11 inches
      XPG = 8.5
      YPG = 5.
C     Compute location of physical origin in inches
C     so that plot will be centered on the page
      XRL = 0.5 * (XPAGE-XPG)
      YRL = 1.5
C     Define location of physical origin relative
C     to lower left corner of page
      CALL ORIGIN (XRL,YRL)
C     Disable border plotting by SETSUB
      CALL NOBORD
C     Define sub-plot area in terms of length of X and Y axis in
C     inches (border plotting disabled by previous call to NOBORD)
C     Add some color
```

```
      CALL SETSUB (XPG,YPG)
C
C    add background color
C
      CALL SETCLR ('DBLUE')
      CALL FILBKG
      CALL SETCLR ('YELLOW')
C
C **********************************************************
C
C    Step 4 - define plot heading and labels
C    Set current text height in inches as desired
      CALL CLASIC
      CALL FILCHR (90.,1,.002,1)
      CALL HRDSHD
      SIZE = 0.15
      CALL HEIGHT (SIZE)
C    Set X axis label for 2-D plot
      CALL XLABEL ('MINUTES TO TOUCHDOWN',100)
C    Set Y axis label for 2-D plot
      CALL YLABEL ('ALTITUDE FEET',100)
C    Define plot heading
      CALL SETCLR ('GREEN')
      CALL PTITLE ('SPACE SHUTTLE REENTRY',100,2.,1)
C
C **********************************************************
C
C    Step 5 - draw the plot axes
      CALL INTAXS
      CALL YANGLE (0.)
      CALL YMARKS (5)
      CALL XMARKS (5)
      CALL AXES2D (25.,-5.,0.,0.,50000.,250000.)
C
C **********************************************************
C
C    Step 6 - draw the plot curve
C
      CALL CUBSPL
      CALL CRVWID (0.05)
      CALL MARKER (-1)
C    plot curve and user defined symbols.
      CALL CURVE (X,Y,7,1)
      CALL SETCLR ('GREEN')
      CALL CRVWID (0.02)
      CALL FRAME
      CALL SETCLR ('GREEN')
      SIZE = 0.3
      CALL HEIGHT (SIZE)
      CALL TXTMSG ('Figure A-8: Reentry',100,-100.,-1.2)
C    CALL SETCLR ('RED')
```

```
C    CALL UGSEAL (1.)
C
C *******************************************************************
C
C    Step 7 - end the sub-plot
     CALL ENDSUB (0)
C
C *******************************************************************
C
C    Step 8 - end the plot
     CALL STOPLT (0)
C
C *******************************************************************
C
C    Step 9 - close the output device
     CALL FINPLT
C    If hardcopy device, print copy
     CALL HRDCPY(0)
     END
C
C *******************************************************************
C
     SUBROUTINE SYMDEF(I)
C
C    This routine draws the shuttle at various angles.
C    It is user defined symbol for plot curve.
C
     DIMENSION XD(23),YD(23),ANGLE(7),XP(10),YP(10)
     COMMON /R2D2/ KNT
     PARAMETER (CC=3.14159/180.)
C    digitized data for user defined symbol
     DATA ANGLE/10.3,25.7,40.1,27.4,18.7,12.0,0.0/
     DATA XD/-.85,.70,.80,.83,.73,.63,.55,-.50,-.75,-.85,-.85,-.85,
    +       -1.03,-1.03,-.85,-.50,.42,.25,-.13,-.5,.63,.53,.53/
     DATA YD/-.15,-.15,-.08,0.0,.09,.10,.20,.20,.60,.60,-.15,.12,
    +        .17,-.17,-.12,-.03,-.03,.07,.07,-.03,.10,.10,.20/
C    increment counter for angle data for picture symbol
     KNT=KNT+1
C    scale down picture symbol
     CALL MRKSIZ (0.4)
C    rotate symbol by amount in angle  array in common
     CALL SYMROT (ANGLE(KNT))
C    draw symbol
     CALL SETCLR ('YELLOW')
     CALL SYMCRV (XD   ,YD   ,11)
     CALL SYMCRV (XD(12),YD(12),4)
     CALL SYMCRV (XD(16),YD(16),5)
     CALL SYMCRV (XD(21),YD(21),3)
C    set character height
     CALL HEIGHT (0.08)
C    get location for center of symbol
```

```
      CALL MRKLOC (XSYM,YSYM)
C     transform data to blank symbol
      THETA=CC*ANGLE(KNT)
       DO 100 I=1,10
       XP(I)=0.4*(XD(I)*COS(THETA)-YD(I)*SIN(THETA))+XSYM
       YP(I)=0.4*(XD(I)*SIN(THETA)+YD(I)*COS(THETA))+YSYM
 100    CONTINUE
      CALL BLPOLY (XP,YP,10,0.0)
C     write reentry angles out (except for last one)
      IF (KNT.NE.7) THEN
       YOFF=0.0625
       IF (KNT.EQ.1) YOFF=-0.2
       CALL REALNO (ANGLE(KNT),1,XSYM+.4,YSYM+YOFF)
       CALL TXTMSR (' DEG.',100,0.,0.,3)
      END IF
      RETURN
      END
C
C ************************************************************
```

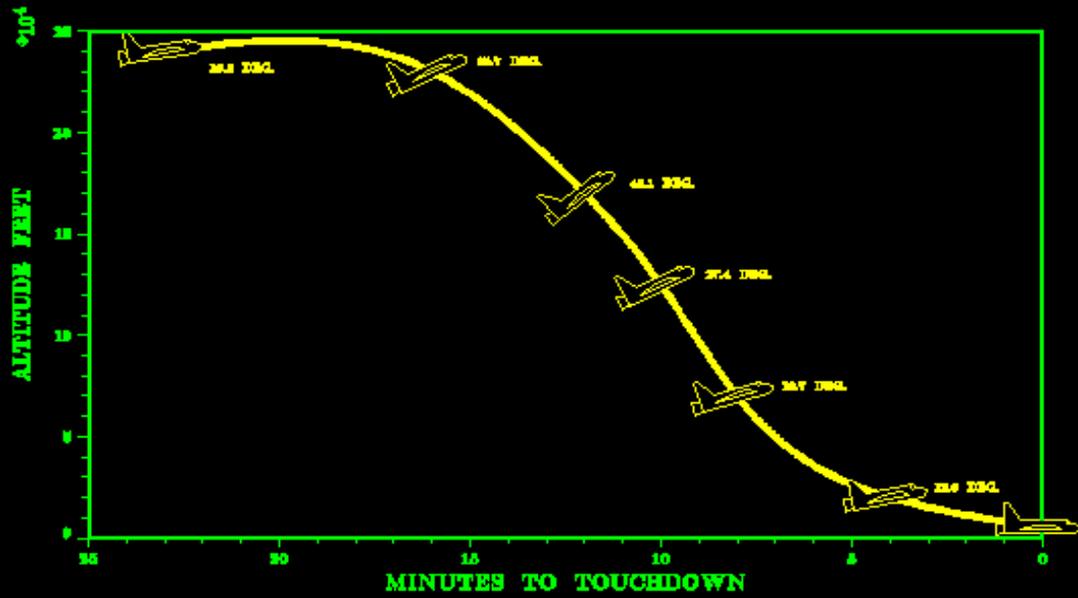# SPACE SHUTTLE REENTRY

Figure A-8: Reentry
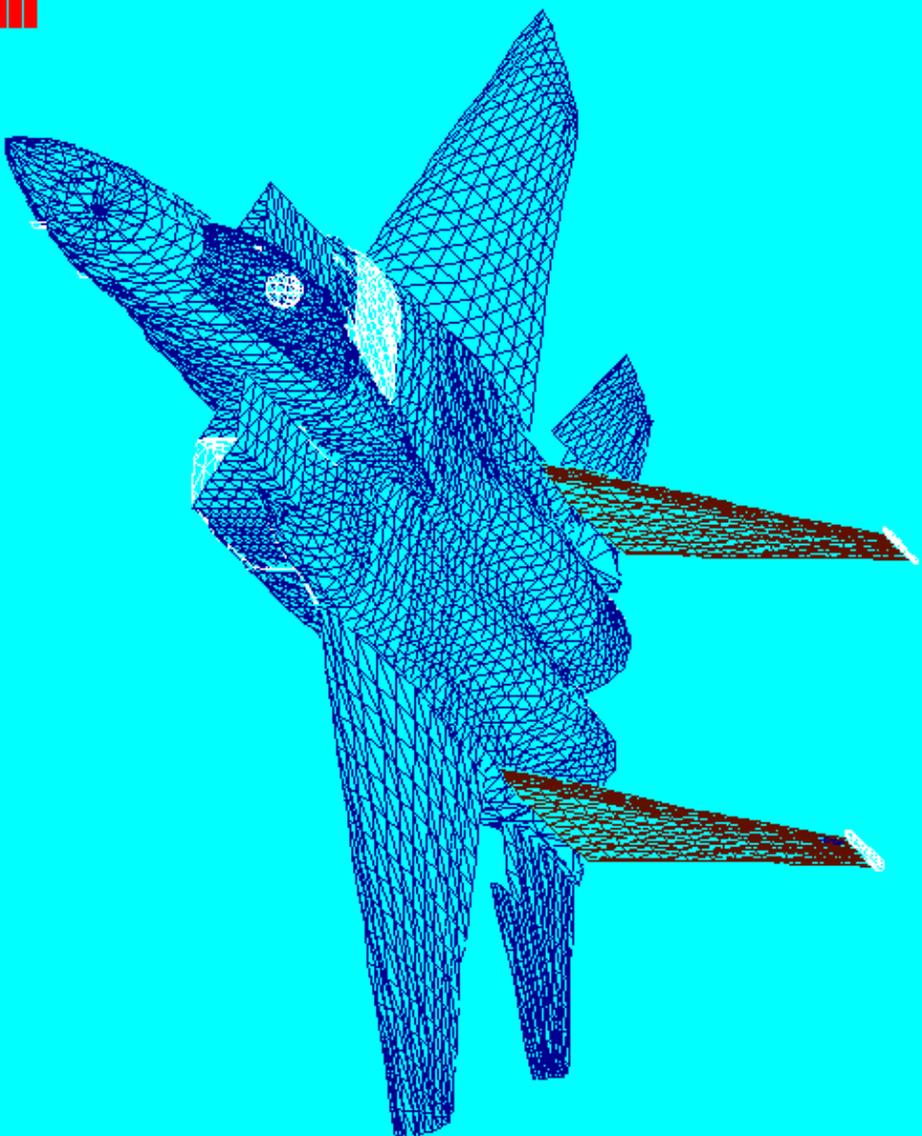
Example 2

```
PROGRAM F15
C
    OPEN(10,FILE='F15.DAT',ERR=900,STATUS='OLD')
    READ(10,*) XMN,YMN,XMX,YMX
C
    KTYPE=0
    CALL DEVICE(KTYPE,XPAGE,YPAGE)
    CALL PAGE(XPAGE,YPAGE)
    CALL NOBORD
    CALL ORIGIN(0.,0.)
    CALL SETSUB(11.,8.5)
C             Draw background
    CALL HRDSHD
    CALL SETCLR('DBLUE')
    CALL SETCLR('CYAN')
C
C    What colors are best????
C
    CALL FILBKG
    CALL CLASIC
    CALL FILCHR(90.,1,.002,1)
    CALL SETCLR('BLACK')
    CALL HEIGHT(0.4)
    CALL TXTMSG('F-15 Eagle',100,6.61,7.41)
    CALL SETCLR('RED')
    CALL TXTMSG('F-15 Eagle',100,6.7,7.5)
C
    CALL SETCLR('GRAY')
    CALL SETCLR('MAGENTA')
    CALL SETCLR('DBLUE')
    CALL LNDWID(0.05)
      DO 444 I=1,16346
      READ(10,*,ERR=445) X1,Y1,X2,Y2
      IF(I.EQ.4722) THEN
        CALL SETCLR('CYAN')
        CALL SETCLR('DGREEN')
        CALL SETCLR('BROWN')
      END IF
      IF(I.EQ.15333) THEN
        CALL SETCLR('WHITE')
      END IF
       IF(I.EQ.6885) THEN
        CALL SETCLR('GRAY')
        CALL SETCLR('DBLUE')
      END IF
      X1=11.*X1/6000.-0.8
      X2=11.*X2/6000.-0.8
      Y1=8.5*Y1/4400.-0.8
      Y2=8.5*Y2/4400.-0.8
```

```fortran
      CALL MOVETO(X1,Y1)
      CALL DRAWTO(X2,Y2)
 444    CONTINUE
 445 CONTINUE
      CLOSE(10)
C
      CALL SETCLR('RED')
      CALL GAESEAL(1.)
      CALL ENDSUB(0)
      CALL STOPLT(0)
      CALL FINPLT
      GO TO 999
C
 900 STOP 'INPUT FILE F15.DAT NOT FOUND'
C
 999 CONTINUE
      END
```

F-15 Eagle

Example 3

```
C ******************************************************************
C
      PROGRAM VIEW3D
C
C    This program demonstrates a 3d surface plot.
C    The surface is the same as shown in 2D contour plot example.
C
C    Original Source: Universal Graphics Inc.   (UGI)
C
      PARAMETER (NWY=40)
      PARAMETER (NWX=50)
      DIMENSION W(NWX,NWY)
C
C    define data for plot
C
      M=50
      N=40
      XM=8.
      YN=9.
       DO 1 J=1,N
       Y=8.*FLOAT(J-1)/FLOAT(N-1)+1.
        DO 1 I=1,M
        X=8.*FLOAT(I-1)/FLOAT(M-1)+1.
        W(I,J)=X+YN-Y-2.*SQRT((X-XM*.5)**2+(Y-YN*.55)**2) -
     1      6./SQRT((X-XM*.5)**2+(Y-YN*.3)**2)+3./X+3./Y
     2      -5.*EXP(-(X-XM*1.01)**2-(Y-YN*.5)**2)
     3      +5.*EXP(-(X-XM*.9)**2-(Y-YN*.9)**2)
        W(I,J)=MAX(W(I,J),-7.)
    1    CONTINUE
C
C ******************************************************************
C
C    Step 1 - initialize output device
C    Allow user to choose a device interactively
      KTYPE=0
      CALL DEVICE (KTYPE,XPAGE,YPAGE)
C
C ******************************************************************
C
C    Step 2 - define page size
C    none
C
C ******************************************************************
C
C    Step 3 - define plot axes and sub-plot area
C    Disable border plotting by SETSUB
      CALL NOBORD
C    Define location of physical origin relative
C    to lower left corner of page
```

```
      CALL ORIGIN (0.,0.)
      CALL SETSUB (XPAGE,YPAGE)
      CALL DUPLEX
C
C    add background color
C
      CALL SETCLR ('DBLUE')
      CALL FILBKG
      CALL SETCLR ('YELLOW')
C
C    generate figure number
C
      CALL HEIGHT (0.22)
      CALL TXTMSG ('Figure A-12: 3D Surface Plot',100,0.5,0.25)
C
C ****************************************************************
C
C    Step 4 - define plot heading and labels
C
C    define 3-d work area and axis
      CALL HEIGHT (0.2)
      CALL SETCLR('RED')
      CALL ZLAB3D('Z AXIS ',6)
      CALL XLAB3D('X AXIS ',6)
      CALL YLAB3D('Y AXIS ',6)
      XL=14.
      YL=10.
      ZL=8.
      CALL BOXD3D(XL,YL,ZL)
      CALL SETCLR('GREEN')
      CALL SETCLR('GOLD')
C
C ****************************************************************
C
C    Step 5 - draw the plot axes
C
C    define 3-d view point and box
C
      VPHI=120.0
      VPHI=130.0
      VTHTA=30.0
      VR=80.0
      XMIN=0.
      XSTEP=2.
      XMAX=14.
      YMIN=0.
      YSTEP=2.
      YMAX=10.
      ZMIN=-8.
      ZSTEP=4.
      ZMAX=8.
```

```
      CALL VPAN3D(VPHI,VTHTA,VR)
      CALL AXES3D(XMIN,XSTEP,XMAX,YMIN,YSTEP,YMAX,ZMIN,ZSTEP,ZMAX)
C
C ***************************************************************
C
C    Step 6 - draw the plot curve
C
C    define surface with matrix
      CALL BLK3DS
      IXSP=1
      IYSP=1
      CALL SURMAT(W ,IXSP ,M ,IYSP ,N ,0)
C
C    enter 3d project loop and define 2-d plot
      CALL SETCLR('YELLOW')
      CALL PROJ3D(0.0,0.0,0.0,1.0,0.0,0.0,0.0,1.0,0.0)
      CALL INTAXS
      CALL SETSUB(XL,YL)
      CALL XLABEL(' ',0)
      CALL YLABEL(' ',0)
      CALL AXES2D(XMIN,XSTEP,XMAX,YMIN,YSTEP,YMAX)
      CALL GRID(1,1)
      CALL ENDP3D
C
      CALL SETCLR('MAGE')
      CALL PROJ3D(XL ,0.0,0.0, XL ,YL,0.0, XL ,0.0,ZL)
      CALL INTAXS
      CALL SETSUB(YL,ZL)
      CALL XLABEL(' ',0)
      CALL YLABEL(' ',0)
      CALL AXES2D(YMIN,YSTEP,YMAX,ZMIN,ZSTEP,ZMAX)
      CALL GRID(1,1)
      CALL ENDP3D
C
      CALL SETCLR('CYAN')
      CALL PROJ3D(XL,0.,0.0, XL,0.,ZL, 0.0,0.,ZL)
      CALL INTAXS
      CALL SETSUB(ZL,XL)
      CALL XLABEL(' ',0)
      CALL YLABEL(' ',0)
      CALL AXES2D(ZMIN,ZSTEP,ZMAX,XMIN,XSTEP,XMAX)
      CALL GRID(1,1)
      CALL ENDP3D
C
C ***************************************************************
C
C    Step 7 - end the sub-plot
      CALL ENDSUB (0)
C
C ***************************************************************
C
```

```
C     Step 8 - end the plot
      CALL STOPLT (0)
C
C ***************************************************************
C
C     Step 9 - close the output device
      CALL FINPLT
C     If hardcopy device, print copy
      CALL HRDCPY(0)
      END
C
C ***************************************************************
```
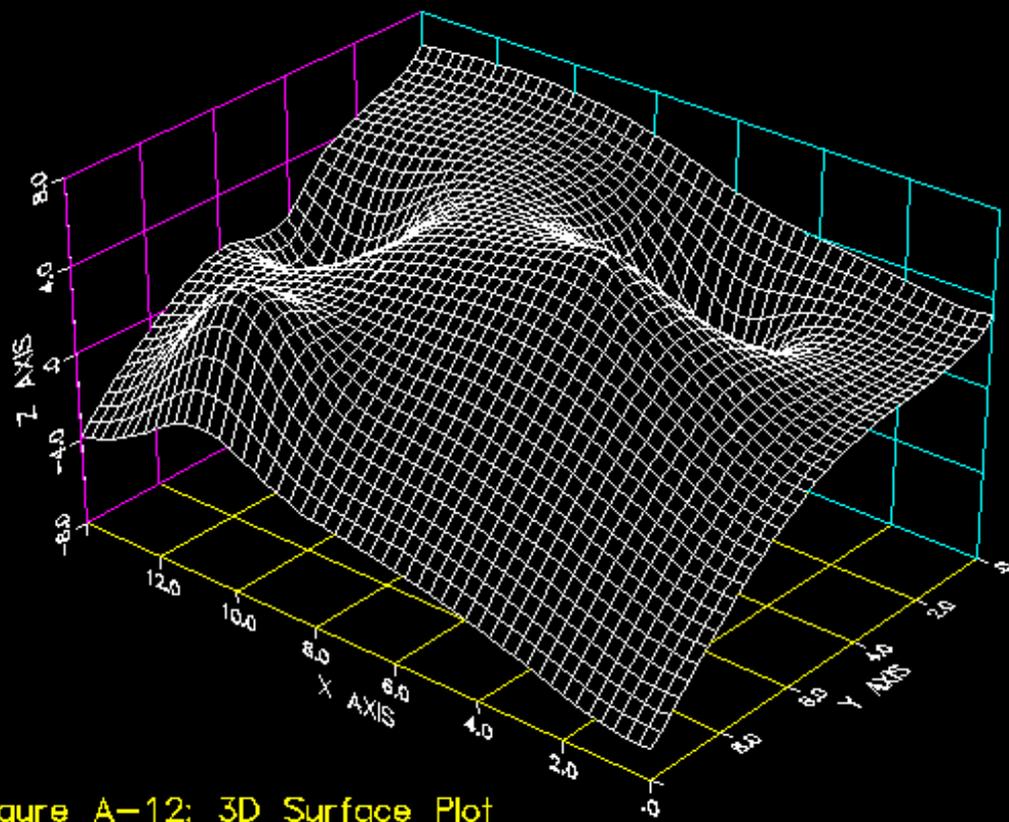
Figure A-12: 3D Surface Plot