

Quick Start

MATFOR in Fortran

ANCAD INCORPORATED

TEL: +886(2) 8923-5411

FAX: +886(2) 2928-9364

support@ancad.com

www.ancad.com

Information in this instruction manual is subject to change without notice.

While AnCAD Incorporated makes every endeavor to ensure the accuracy of this document, it does not accept liability for any errors or omissions or for any consequences arising from the use of the program or documentation.

Quick Start: MATFOR Version 4.1

© Copyright AnCAD Incorporated 2007

All rights reserved.

All trademarks where used are acknowledged.

Contents

CONTENTS.....	3
CHAPTER 1 INTRODUCTION	4
I. PRODUCT DESCRIPTION.....	4
II. COMPILER REQUIREMENTS.....	5
III. INSTALLATION	5
IV. REGISTRATION.....	7
V. FLOATING LICENSE & CLASSROOM LICENSE	7
CHAPTER 2 PROJECT SETTINGS.....	10
<i>IN WINDOWS</i>	
I. COMPAQ VISUAL FORTRAN	10
II. INTEL VISUAL FORTRAN	12
III. LAHEY FORTRAN	14
IV. ABSOFT FORTRAN	16
<i>IN LINUX</i>	
V. ALL COMPILERS	16
TO RUN A DEMO CASE	16
CHAPTER 3 FUNDAMENTALS.....	17
I. MFARRAY	17
II. NUMERICAL PROCEDURES.....	18
III. VISUALIZATION PROCEDURES	20
CHAPTER 4 A FIRST PROGRAM	22
I. HELLO SURFACE	22
II. TO ANIMATE HELLO SURFACE	25
III. TO RECORD HELLO SURFACE.....	28
CHAPTER 5 THE ADVANCED FEATURES	30
I. MFPLAYER.....	30
II. GRAPHICS VIEWER.....	31
III. MATFOR GUI BUILDER	32

Chapter

1

INTRODUCTION

This Quick Start gives a brief overview of using the different components in MATFOR version 4. Language considerations, the structure of programs, and the use of advanced tools are all covered.

I. PRODUCT DESCRIPTION

MATFOR is a set of numerical and visualization libraries developed to enhance programming in C++ and Fortran environments. Especially designed for scientists and engineers, MATFOR aims to accurately perform computation, dynamically visualize data, and efficiently decrease process time. Its features include:

mfArray, a MATFOR defined data type that simplifies and facilitates object-oriented programming in C++ and Fortran environments.

Numerical Library contains useful linear algebraic functions subject to assist users with computational problem solving.

Visualization Library collects well-designed graphical procedures and controls to support a variety of 2D and 3D visual functions.

Data Viewer, organized in spreadsheet format, is one convenient platform for data management, filter, and analysis.

Graphics Viewer, besides its highly customized user interface, overthrows the convention of post-processing as it instantly visualizes scientific and engineering data.

mfPlayer animates simulation results by reading and displaying numerical data, and further allows additional graphical manipulations.

Graphics Export converts dynamic presentation into standalone movie or image files to enhance accessibility of simulation results.

GUI System facilitates application-building by allowing users to create an interface of their preference.

II. COMPILER REQUIREMENTS

MATFOR supports these compiler choices:

Platform and System Requirements		
Platform	Operating System	Fortran Compilers
Intel Based 32-bit systems	Windows 2000/ XP	<ul style="list-style-type: none"> ◆ Compaq Visual Fortran 6.6 C ◆ Intel Visual Fortran 8.1/9.0/9.1 ◆ Lahey Fortran 5.7 / 7.1 ◆ Absoft Fortran 9.0/10.0
Intel Based 32-bit systems	Fedora Core 1 / 2 /3 / 4 Red Hat Enterprise Linux 3.0 / 4.0 White-Box Enterprise 3 SuSE 9.1 Enterprise Mandrake 10	<ul style="list-style-type: none"> ◆ Intel Fortran 8.1/9.0/9.1 ◆ Lahey Fortran 6.2c ◆ Absoft Fortran 9.0/10.0
EM64T/AMD64 64-bit systems	Fedora Core 2 / 3 Red Hat Enterprise Linux 4.0	<ul style="list-style-type: none"> ◆ Intel Fortran 8.1/9.0/9.1 ◆ Absoft Fortran 10.0 ◆ PathScale 2.4

Due to constant updates of the compilers, these requirements are subjected to change without further notice. Please refer to the latest version of compiler requirements at <http://www.ancad.com/requirements.html>.

III. INSTALLATION

This section provides step-by-step instructions for installing MATFOR. If you encounter any problems during the installation, please contact support@ancad.com.

WINDOWS INSTALLATION

• Pre-Installation

1. Exit any of MATFOR programs executing;
2. Remove all previous versions of MATFOR components if this is an upgrade;
3. Ensure the Compiler Requirements are satisfied;
4. Obtain administrator rights under Windows 98/2000/NT/XP.

• Begin Installation

1. Insert the MATFOR CD into the CD-ROM drive.
The standard MATFOR Installation Procedure shall start automatically. If the Procedure fails to start, you may manually start it by double-clicking on the MATFOR.exe file under the <CDROM>\Content\ path.
2. Follow the self-explanatory instructions in the Procedure to set up all MATFOR components.
3. Specify a destination folder to create MATFOR through the Procedure.
By default, the Package creates a program group in your C drive under the

path **C:\Program Files\AnCAD\MATFOR4**.

- **Set Environment Path**

The Procedure automatically sets the \$PATH and \$MATFOR4DIR environment variables.

To manually set the environment variables:

1. Go to **Control Panel\System**.
2. In the **System Dialog Box**, select the **Advance** label and click on the **Environment Variables** tab.
3. In the **System Variables Box**, add **<MATFOR4>\bin** and **<MATFOR4>\tools** to the [path] variable.

NOTE

<MATFOR4> shall specify the directory under which MATFOR is installed.

i.e. C:\Program Files\AnCAD\MATFOR4

LINUX INSTALLATION

- **Pre-Installation**

1. Exit any of MATFOR programs executing.
2. Remove all previous versions of MATFOR components if this is an upgrade.
3. Ensure the OS Requirements are satisfied.
4. Ensure the Compiler Requirements are satisfied.
5. Obtain root rights under Linux OS.
6. Run the '**rpm -ivh compat-libstdc++-33-3.2.3-47.3.i386.rpm**' command to install the compat-libstdc++-33 package. (Optional)

- **Begin Installation**

1. Open a command shell.
2. Change the current directory to the one containing the **matfor_f-4.xx-i686.rpm** Installer.
3. Run the '**rpm -ivh matfor_f-4.xx -i686.rpm**' command.
4. Follow the self-explanatory instructions in the Procedure to set up all MATFOR components. By default, the Installer creates a program group under the path **/usr/lib/matfor4**.

- **Environment Path Settings**

The Procedure automatically sets the \$PATH and \$MATFOR4DIR environment variables.

To manually set the environment variables:

1. Add the following lines to your `~/.bashrc` file.
2.

```
export MATFOR4DIR=/usr/lib/matfor4
export LD_LIBRARY_PATH=$MATFOR4DIR/lib:$LD_LIBRARY_PATH
export PATH=$MATFOR4DIR/tools:$PATH
```
3. Run the '`source ~/.bashrc`' command or re-login your system.

IV. REGISTRATION

At the end of installation, you are required to enter the License Password for registration, which can only be obtained by submitting the Host ID of your local machine to AnCAD, Inc.

1. Go to <http://www.ancad.com/activation.php> to activate MATFOR product with the **Serial Number** and **Host ID** information.
2. MATFOR license key should be sent to you through the email provided within 24 hours.

(Note: trial users may receive trial keys via email upon the receipt of download request. Please go to <http://webphp/download.php> to submit your request.)

UNDER WINDOWS

The registration program can be launched from **Start ► Program Files ► MATFOR4 ► Utilities ► Register MATFOR 4 in Fortran (or C++)**

UNDER LINUX

- Go to `/usr/lib/matfor4/tools`.
- Type `./reg_xxx` to launch the registration program.

V. FLOATING LICENSE & CLASSROOM LICENSE

To enable a group of developers to share a pool of licenses more efficiently, MATFOR supports floating licenses and classroom licenses for cross-system/cross-platform environments. The floating license is designed to be used in any shared network environment, and the classroom license is designed for the purpose of teaching in an academic environment. Any project developed under MATFOR classroom license may not be redistributed to a third party with profit interest or being a commercial institution.

These license models consist of one or more license servers; the license server runs the license management process and monitors number of concurrent users using MATFOR licenses in the network. The installation steps are:

WINDOWS INSTALLATION

- **Installation of License Servers**

1. Run **license_server.exe** (download from <http://www.ancad.com/activation.php>) on machines designated as license servers.
2. Retrieve required information from the machine.
Under command prompt,
 - i. Change directory to **C:\Program Files\AnCAD\License Server**.
 - ii. Execute **Imtools** to get Ethernet Address and Host Name.
3. Collect the following product information:
 - i. Compiler version
 - ii. Operating system
 - iii. Number of license of each version
4. Obtain license file(s) for license server(s).
 - i. Go to <http://www.ancad.com/activation.php> to activate MATFOR with the **Serial Number** and **Host ID** information.
 - ii. The license file(s) should be sent to you through the email provided within 24 hours.
5. Place the license file **xxx.lic** into **<AnCAD>License Server** directory.
6. To start the license server.
Under command prompt, type:
Imgrd -c xxx.lic or
Imgrd -c . for multiple license files.

- **Configuration in Client Computers**

1. Install MATFOR on machines designated as clients.
2. Go to **Start ► Program Files ► MATFOR4 ► Utilities ► Register MATFOR**.
3. Under Registration Window, choose Network License.
4. Fill in the Host Name or IP Address of the license server in the blank area. (IP Address can be obtained from the license server by typing **Imhostid -internet -n** under command prompt.)

LINUX INSTALLATION

- **Installation of License Servers**

1. Run license server executable file (download from <http://www.ancad.com/activation.php>) on machines designated as license servers.
tar xzf license_server.tar.gz -c /usr/lib
2. Retrieve required information from the machine.
Under command shell,
 - i. Go to ***/usr/lib/matfor_licsvr***
 - ii. Type ***./lmhostid -n*** to get Ethernet Address.
 - iii. Type ***./lmhostid -hostname -n*** to get Host Name.
3. Collect the following product information:
 - i. Compiler version
 - ii. Operating system
 - iii. Number of license of each version
4. Obtain license file(s) for license server(s).
 - i. Go to <http://www.ancad.com/activation.php> to activate MATFOR with the **Serial Number** and **Host ID** information.
 - ii. The license file(s) should be sent to you through the email provided within 24 hours.
5. Place the license file **xxx.lic** into ***/usr/lib/matfor_licsvr*** directory.
6. Start the license server.
Under command prompt, type:
lmgrd -c xxx.lic or
lmgrd -c . for multiple license files.

- **Configuration in Client Computers**

1. Install MATFOR on machines designated as clients.
2. Go to ***/usr/lib/matfor4*** and type ***./reg_xxx***
3. Under Registration Window, choose Network License.
4. Fill in the Host Name or IP Address of the license server in the blank area. (IP Address can be obtained from the license server by typing ***lmhostid -internet -n*** under command shell.)

For other questions regarding MATFOR floating license or classroom license installation and registration, please contact support@ancad.com.

Chapter

2

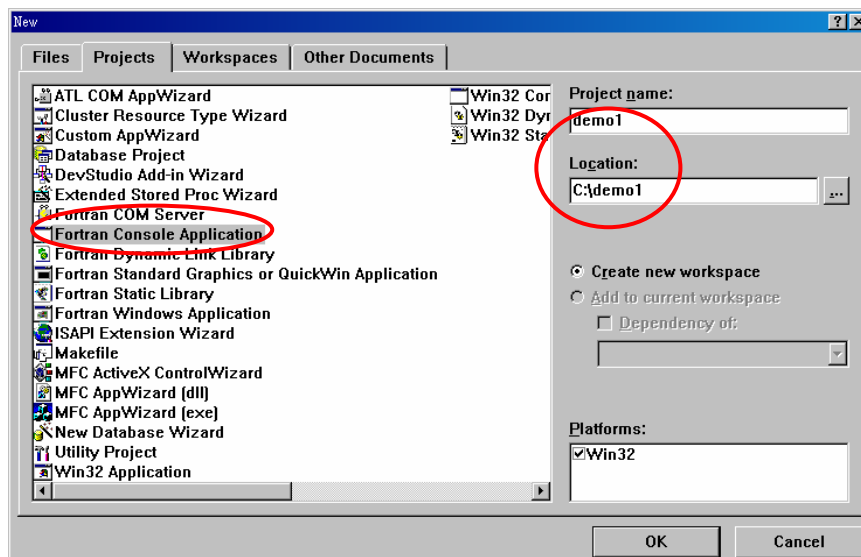
PROJECT SETTINGS

Due to the variety in MATFOR-supporting compilers, this chapter is composed to guide users to configure project settings under different compiler environments.

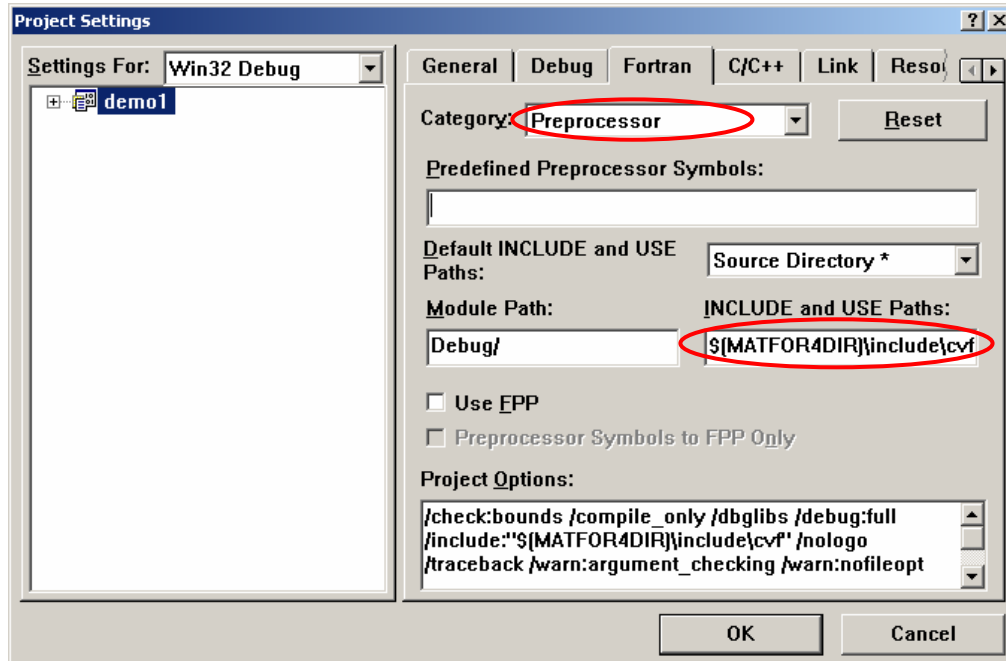
IN WINDOWS

I. COMPAQ VISUAL FORTRAN

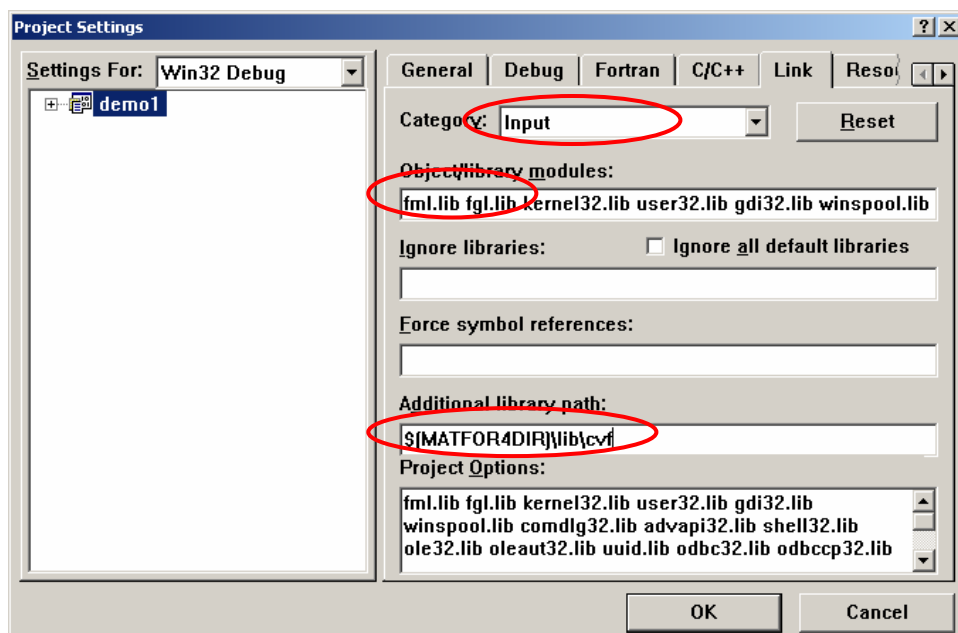
1. Open Compaq Visual Fortran Developer Studio.
2. Go to **File ► New**.
3. Select the project type **Fortran Console Application**.
4. Enter a name for the new project, specify its location, and click **OK**.
5. Select **An empty project** and click **Finish**.



1. Select **Settings** from the **Project** menu.
2. Click on the **Fortran** tab and select **Preprocessor** from the **Category** menu.
3. Under **INCLUDE and USE Paths**, type in: `$(MATFOR4DIR)\include\cvf`

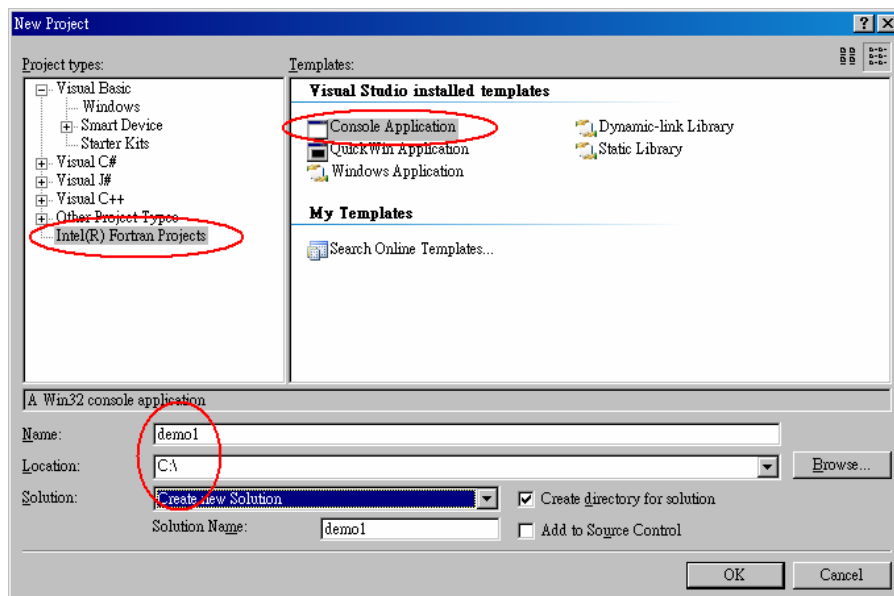


4. Click on the **Link** tab and select **Input** from the **Category** pull-down menu.
5. Under **Object/Library Modules**, add: **fml.lib fgl.lib** (and **spml.lib** if using MATFOR Sparse Array)
6. Under **Additional library directories**, type in: `(MATFOR4DIR)\lib\cvf`

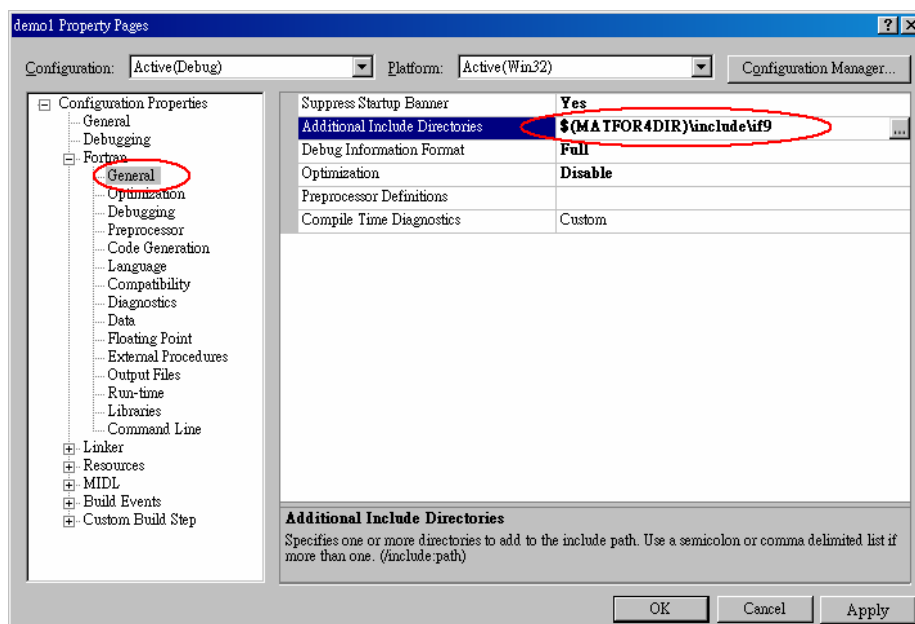


II. INTEL VISUAL FORTRAN

1. Open Microsoft Visual Studio 2005.
2. Go to **File ► New Project**.
3. Select **Intel® Fortran Projects** from the **Project Types** menu.
4. Select **Console Application** from **Templates** and type a solution name.
5. Enter a name for the new file and click **OK**.

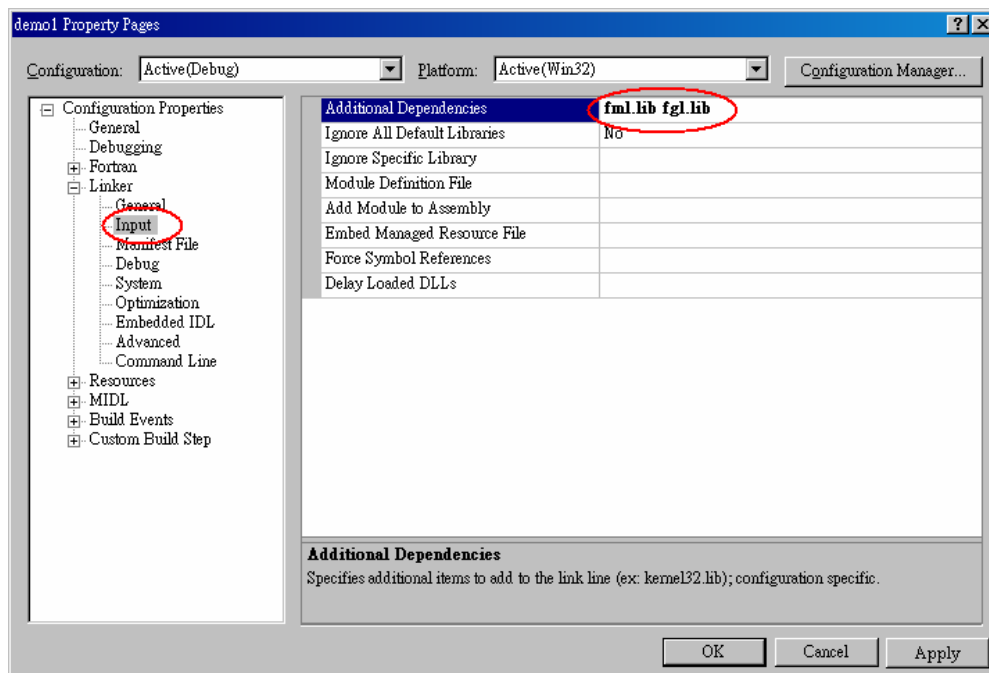
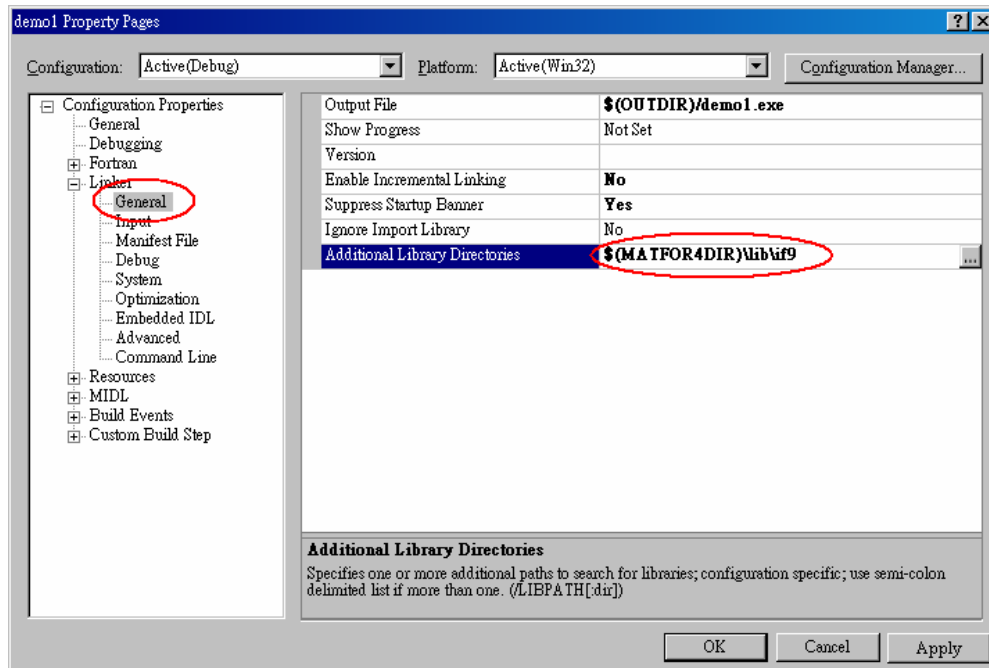


6. Click on **Finish**.
7. Select **Properties** from the **Project** menu.
8. Go to **General** under the **Fortran** folder.
9. In **Additional Include Directories**, type in: `$(MATFOR4DIR)\include\if9`



10. Go to **General** under the **Linker** folder.

11. In **Additional Library Directories**, type in: **\$(MATFOR4DIR)\lib\lib9**



12. Go to **Input** under the **Linker** folder.

13. In **Additional Dependencies**, add: **fml.lib fgl.lib** (and **spml.lib** if using MATFOR Sparse Array)

III. LAHEY FORTRAN

Lahey 5.7 Fortran users

1. Open **Lahey Fortran 95 Console Prompt**.
2. Type in the following command to compile the codes:

```
If95 -i "<MATFORDirectory>\include\lf" -libpath  
"<MATFORDirectory>\lib\lf" -lib fml.lib fgl.lib spml.lib mxui.lib mxuiF.lib  
<fileName>
```

NOTE

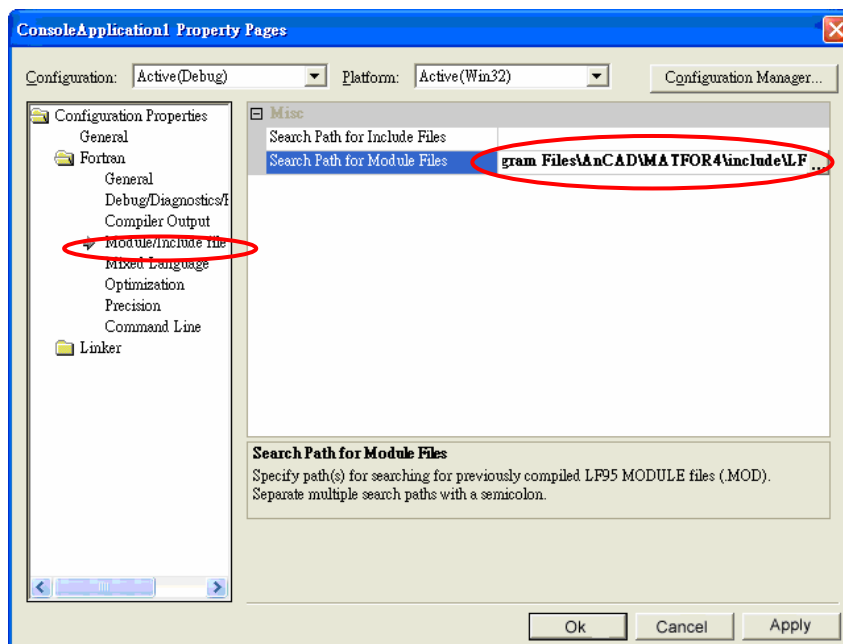
<MATFORDirectory> shall specify the directory under which MATFOR is installed. **<fileName>** shall specify the name given to the file. For example,

```
If95 -i "c:\progra~1\ancad\matfor4\include\lf" -libpath  
"c:\progra~1\ancad\matfor4\lib\lf" -lib fml.lib fgl.lib spml.lib mxui.lib  
mxuiF.lib test.f90
```

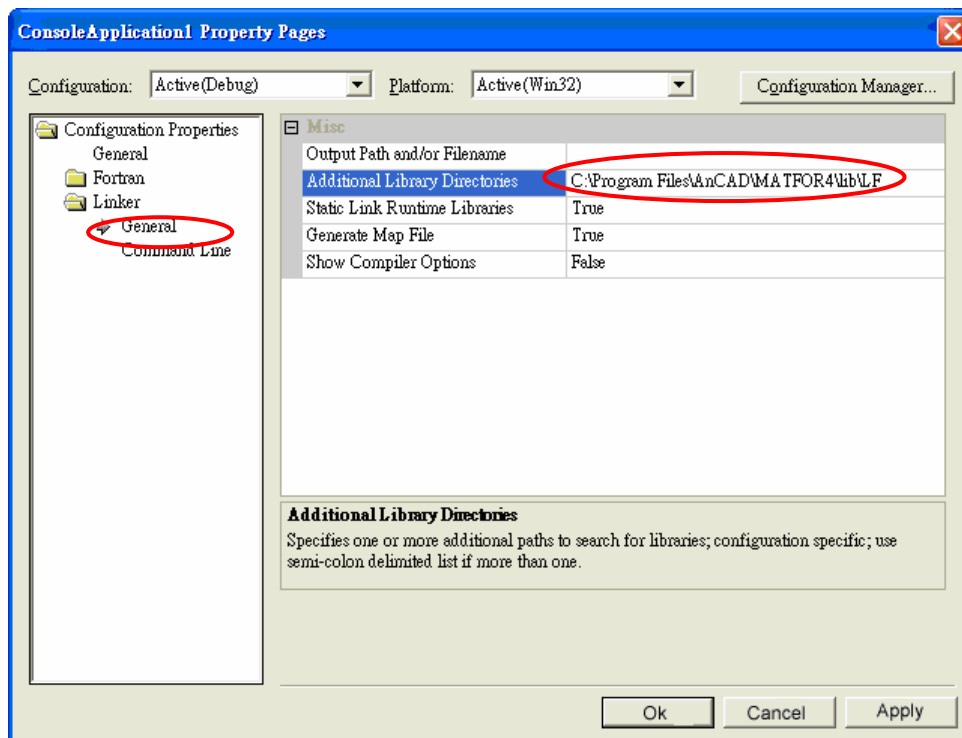
Lahey 7.1 Fortran users

1. Select **Properties** from the **Project** menu.
2. Select **Module/Include file paths** under the **Fortran** label.
3. In Search Path for Module Files, type in MATFOR Module install path, for example,

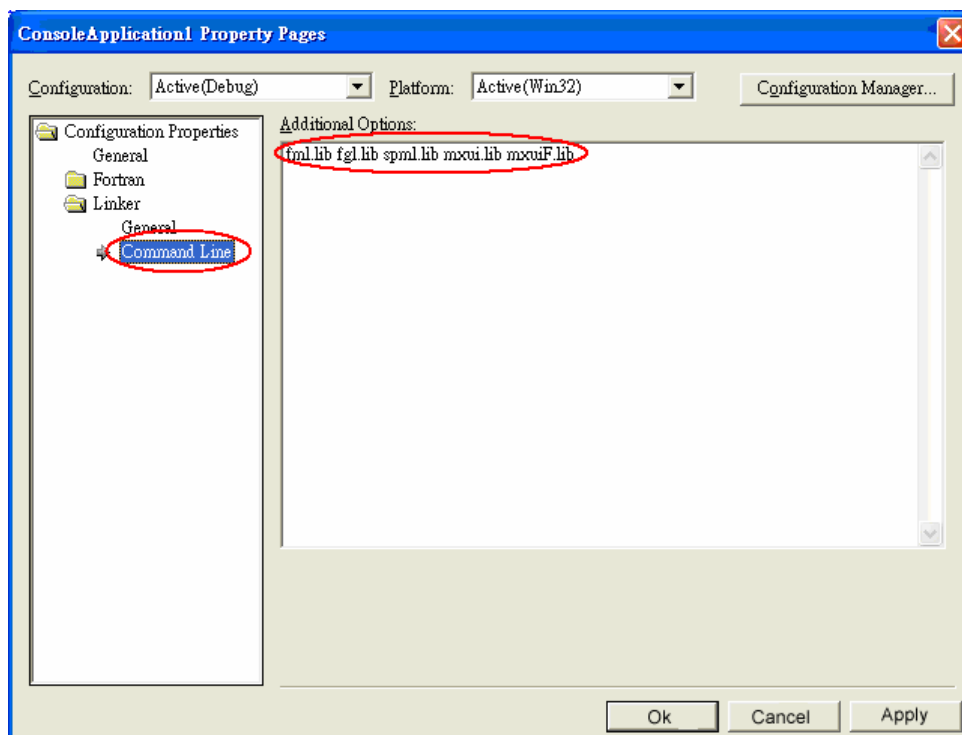
C:\Program Files\AnCAD\MATFOR4\include\LF



4. Select **General** under the **Linker** label.
5. In **Additional Library Directories**, type in: **C:\Program Files\AnCAD\MATFOR4\lib\LF**



6. Select **Command Line** under the **Linker** label.
7. In **Additional Options**, type in: **fml.lib fgl.lib spml.lib mxui.lib mxuiF.lib**



IV. ABSOFT FORTRAN

1. Open Absoft Fortran Development Command Prompt.
2. Copy `win_abf_template.mak` from `<MATFOR4>\tools\template` to the directory of the project.
(Note: users of version 10 shall use `win_abf10_templat.mak`.)
3. Rename the make file to **`win_abf.mak`**.
4. Modify **`win_abf.mak`** as required.
5. Type **`nmake -f win_abf.mak`** to build and compile the project.

IN LINUX

V. ALL COMPILERS

1. Open Command Prompt.
2. Copy **`linux_xxx_template.mak`** from `/usr/lib/matfor4/tools/template` to the directory of the project.
3. Rename the make file to **`linux_xxx.mak`**.
4. Modify **`linux_xxx.mak`** as required.
5. Type **`make -f linux_xxx.mak`** to build and compile the project.

TO RUN A DEMO CASE

UNDER WINDOWS

1. Go to the directory where MATFOR 4 is installed.
2. Open **`demo/fortran/<demo case>`**.
3. Select the **`.dsw`** file to run the demo case.

UNDER LINUX

1. To compile demo cases all at once, go to `/usr/lib/matfor4/demo/fortran`
To compile each demo case individually, go to `/usr/lib/matfor4/demo/fortran/<demo case>`.
2. Type **`make -f linux_if9.mak`**
3. Go to the `/usr/lib/matfor4/demo/fortran/<demo case>` to run each demo case. For example,
`cd $MATFOR4DIR/demo/fortran/3dplot`
`./3dPlot`

Chapter

3

FUNDAMENTALS

This chapter is an introductory summary on how to integrate MATFOR into Fortran programming. Beginning with `mfArray`, the core of MATFOR, the chapter subsequently introduces the numerical library and the visualization library.

I. MFARRAY

`mfArray` is a dynamic array endorsing the data types and dimensions listed in Table 3.1.

Data Type	Statement
character	<code>a = 'A'</code>
character(*)	<code>a = 'string'</code>
Logical	<code>a = .true.</code>
Integer	<code>a = 1</code>
real(4)	<code>a = 1.0</code>
real(8)	<code>a = 3.1418d0</code>
complex(4)	<code>a = (2.0, 3.0)</code>
complex(8)	<code>a = (2.0d0, 3.0d0)</code>

Table 3.1 | `mfArray` Data Types

`mfArray` Declaration

1. Use MATFOR module to appropriately contain `mfArray`.
2. Declare `mfArray` using the format **`type(mfArray) ::<variable>`**

Example Code (ch3-1)**Program Main**

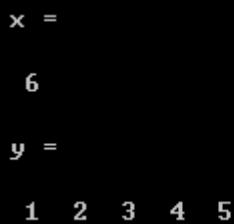
```
use fml                ! Use module fml
implicit none

type(mfArray) :: x, y    ! Declare mfArray variables x and y

x = 6.0d0                ! x is a scalar
y = (/1.0, 2.0, 3.0, 4.0, 5.0/) ! y is a 1-by-five vector containing real values

call msDisplay(x, 'x', y, 'y') ! Call msDisplay subroutine to display x and
y data

end Program Main
```

Result

```
x =
6
y =
1 2 3 4 5
```

II. NUMERICAL PROCEDURES

MATFOR embraces a comprehensive set of numerical procedures furnished with easy-to-call syntax to complement mfArray. Various categories of numerical procedures are contained in the set and are hereto listed in Table 3.2.

For a complete list of the numerical routines and descriptions on their usage and functionality, please refer to the Reference Guide.

Procedure Group	Example	Description
Data Manipulations	mfSort	Sort data in ascending order
Arithmetic Operators	mfRDiv	Right-divide the matrix
Trigonometry	mfCos	Request cosine function
Exponential	mfLog	Request natural logarithm
Complex	mfConj	Generate conjugate of complex
Rounding and Remainder	mfCeil	Round towards positive infinity
Matrices	mfMagic	Construct magic square
Matrix Manipulations	mfFind	Find indices of nonzero elements
Matrix Analysis	mfNorm	Generate matrix/vector norm
Linear Equations	mfChol	Request Cholesky factorization
Eigenvalues and Singular Values	mfSchur	Perform Schur decomposition
Factorization Utilities	mfBalance	Perform diagonal scaling

Table 3.2 | Numerical Procedure Brief

Numerical Procedure Call

1. Use fml to appropriately contain the numerical routine(s).
2. To retrieve the procedure:
 - If the procedure is a function, use the assignment operator "=";
 - Else, if the procedure is a subroutine, retrieve the subroutine by using **call**.

Example Code (ch3-2)

Program Main

```

use fml                                ! Use module fml
implicit none

type(mfArray) :: m, i, j               ! Declare mfArray variables m, i, and j

m = mfMagic(3)                         ! Construct a 3-by-3 magic matrix and assign it
to m
call msFind(mfOut(i, j), m)           ! Retrieve the row-column indices of the
nonzero(s) in m
                                     ! i and j describe the row and column,
respectively

call msDisplay(m)                     ! Display matrix m
call msDisplay( i, 'i', j, 'j')       ! Display the row-column indices of the

```

```
nonzero(s)
```

```
end Program Main
```

Result

```
Ans =
```

```
8 1 6
3 5 7
4 9 2
```

```
i =
```

```
1 2 3 1 2 3 1 2 3
```

```
j =
```

```
1 1 1 2 2 2 3 3 3
```

III. VISUALIZATION PROCEDURES

In addition to the numerical procedures, MATFOR endorses a collection of visualization routines to enhance data comprehension. Table 3.3 summarizes the procedures and introduces an example routine with description for each procedure group.

Procedure Group	Example	Description
Window Management	mfWindowSize	Set the frame size of Graphics Viewer window
Visualization Controls	mfSubplot	Create subplot in active figure
Graph Options	mfSurf	Construct 3D surface plot
Advanced Graph Options	mfGetDelaunay3	Construct 3D Delaunay triangulation
Object Manipulations	mfObjOrigin	Set the origin of the drawn object
Appearance Settings	mfColormap	Specify the colormap type of the drawn object
Annotations	mfTitle	Annotate the graph with title
3D Objects	mfSphere	Draw a sphere

Table 3.3 | Visualization Procedure Brief

For a complete list of the visualization routines and descriptions on their usage and functionality, please refer to the Reference Guide.

Visualization Procedure Call

1. Use fgl to appropriately contain the visualization routine(s).

2. To retrieve the procedure:

- If the procedure is a function, use the assignment operator "=";
- Else, if the procedure is a subroutine, retrieve the subroutine by using **call**.

Example Code (ch3-3)

Program Main

```

use fml                                ! Use numerical routines
use fgl                                ! Use visualization routines
implicit none

type(mfArray) :: nx, ny, nz
type(mfArray) :: x, y, z, c, tet       ! Declare mfArray variables

nx = mfLinspace(-2, 2.2d0, 21)         ! Construct linearly spaced vector nx
ny = mfLinspace(-2, 2.25d0, 17)        ! Construct linearly spaced vector ny
nz = mfLinspace(-1.5d0, 1.6d0, 31)     ! Construct linearly spaced vector nz

call msMeshgrid(mfOut(y, x, z), ny, nx, nz) ! Construct grid matrices
for 3D plotting

c = 2 * mfCos(x**2) * mfExp( - (y**2) - (z**2)) ! Mathematically define c

tet = mfGetDelaunay3(x, y, z)          ! Construct 3D Delaunay triangulation

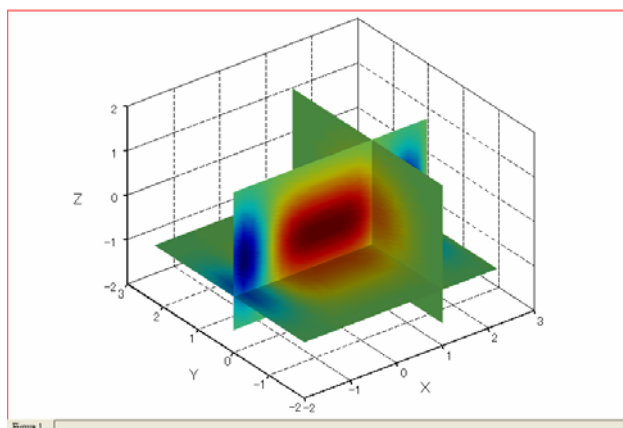
call msTetSliceXYZ( tet, x, y, z, c, &
  mf((/1.0d0, 1.0d0/)), mf(0), mf(-0.75d0) ) ! Display orthogonal sliced
planes

call msViewPause()                     ! Pause to display the graph
call msFreeArgs(nx, ny, nz, x, y, z, c) ! Deallocate mfArray

end Program Main

```

Result



Chapter

4

A FIRST PROGRAM

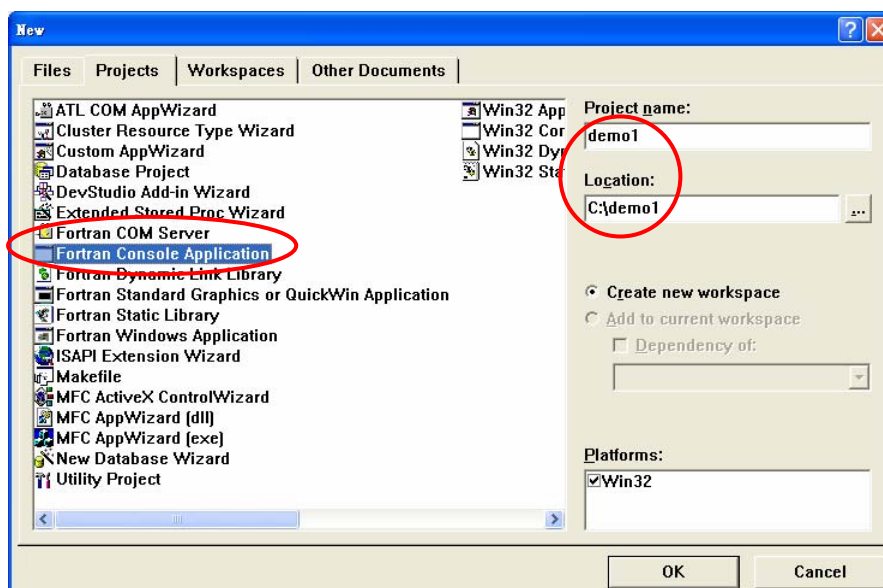
This chapter presents one simple visualization program for users to manipulate and extends as animation and recording procedures are introduced.

I. HELLO SURFACE

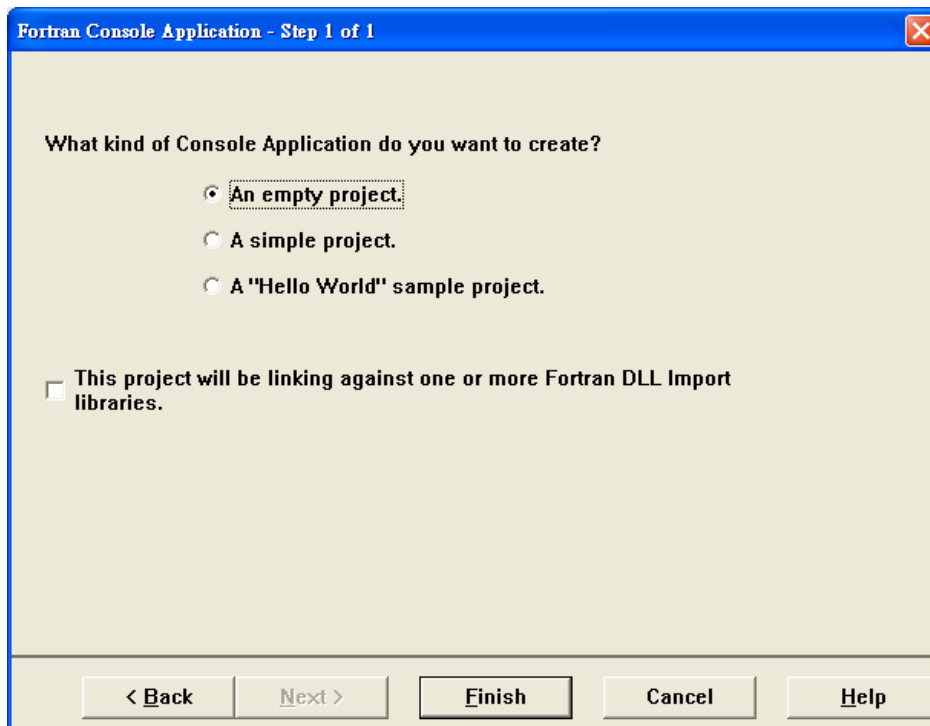
The section shall take you step by step to create a surface plot.

Create a New Project/File

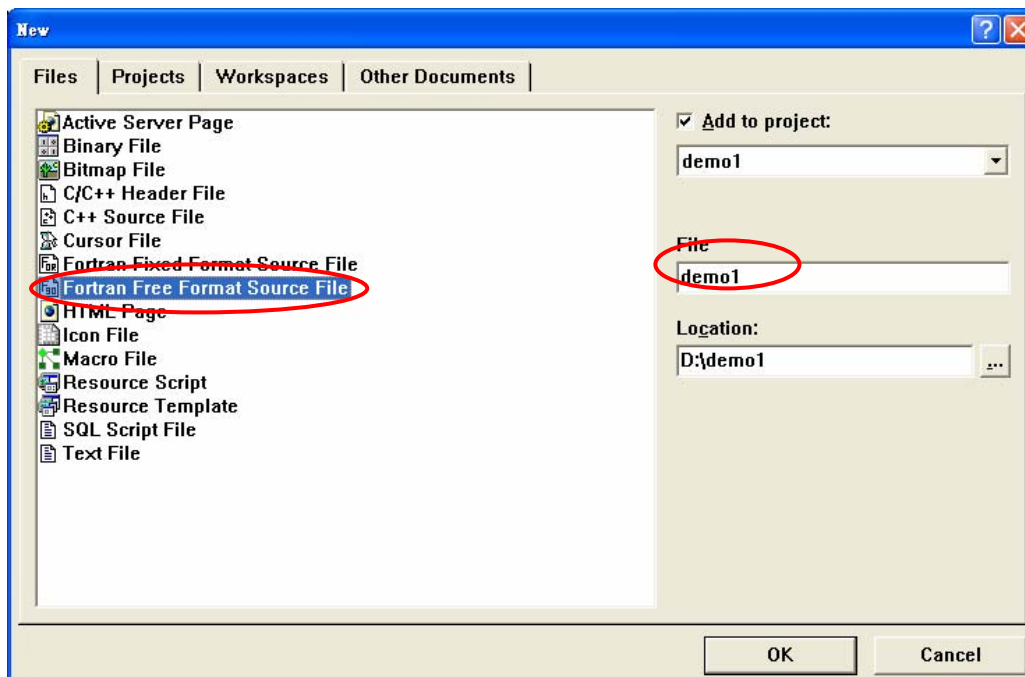
1. Select the project type **Fortran Console Application**.
2. Name the project “**demo1**,” specify its location, and click **OK**.



3. Select **An empty project** and click **Finish**.



4. Select the file type **Fortran Free Format Source File**.
5. Name the file "**demo1**," specify its location, and click **OK**.



Adding the Source Code(ch4-1)

In “demo1.f90,”type in:

Program demo1

```
use fml                                ! Use numerical routines
use fgl                                ! Use visualization routines
implicit none

type(mfArray) :: x, y, z               ! Declare mfArray variables

call msMeshgrid( mfOut(x, y), &       ! Construct grid matrices for 3D
plotting                               plotting
                                mfLinspace(-3, 3, 30), &
                                mfLinspace(-3, 3, 30) )

z = mfSin(x) * mfCos(y)               ! Mathematically define z

call msSurf(x, y, z)                 ! Plot a surf using mfArrays x, y,
and z

call msViewPause()                   ! Pause to display the graph

call msFreeArgs(x, y, z)             ! Deallocate mfArray

end Program demo1
```

Line-by-Line Walkthrough

use fml

This includes the module fml which contains the MATFOR numerical procedures.

use fgl

This includes the module fgl which contains the MATFOR visualization procedures.

type(mfArray) :: x, y, z

This declares mfArray variables x, y, and z.

call msMeshgrid(mfOut(x, y), mfLinspace(-3, 3, 30), mfLinspace(-3, 3, 30))

This generates x and y grid matrices from the domains specified by the mfLinspace-generated vectors. This procedure aims to solve functions for two variables by plotting 3D graphs.

- mfOut(x, y) specifies x and y as mfArray outputs.
- mfLinspace(-3, 3, 30) constructs a vector with 30 linearly spaced points between -3 and 3.

z = mfSin(x) * mfCos(y)

This defines z mathematically.

call msSurf(x, y, z)

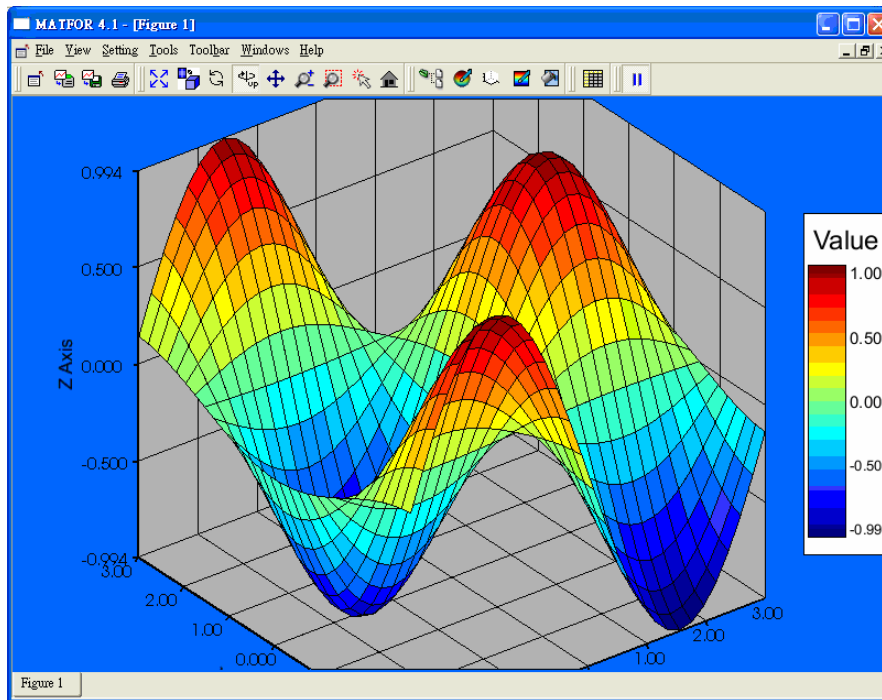
This creates a three-dimensional surface plot of the x-, y-, and z-coordinates.

call msViewPause();

This halts program execution for graphical display. `msViewPause()` should be added after each set of graphical creation routines.

call msFreeArgs(x, y, z)

This deallocates the previously declared `mfArray` variables.

Compile and Run**II. TO ANIMATE HELLO SURFACE**

In this section, the Hello Surface demo code is modified to generate animation.

Animation

Animations effects are produced by continuously updating data displayed in the Graphics Viewer.

To Animate Data

1. Use `fgl` to appropriately contain the visualization routine(s).
2. Construct and initialize the `mfArrays` for plotting.
3. Create a static plot of the graph to be animated.
4. Set up an iteration loop for the range of data to be observed through animation.
5. Within the loop, use procedure `call msGSet(handle, 'axis-data', data)` to update the targeted data of the current draw.
6. Update the drawn figure accordingly by using procedure `msDrawNow`.

7. Use procedure *call msViewPause* after the end of animation to observe the static graph.

Example Code(ch4-2)

Modify the previously created “**demo1.f90**” file as follows and rename it “**demo2.f90**”:

Program demo2

```

use fml                                ! Use numerical routines
use fgl                                ! Use visualization routines
implicit none

type(mfArray) :: x, y, z, h            ! Declare mfArray variables
integer(4) :: i                        ! Declare an integer variable

call msMeshgrid( mfOut(x, y), &       ! Construct grid matrices for 3D
plotting
                        mfLinspace(-3, 3, 30), &
                        mfLinspace(-3, 3, 30) )

do i=1, 10                             ! Create loop

    z = mfSin(x+i/8.0d0) * mfCos(y)    ! Mathematically define z

    if (i == 1) then                    ! Plot a surf using mfArrays x, y, and z
        h = mfSurf(x, y, z)
        call msDrawNow
    else                                ! Update the figure as the
z-coordinate varies
        call msGSet(h, 'zdata', z)
        call msDrawNow
    end if                             ! End if

end do                                  ! End loop

call msViewPause()                     ! Pause to display the graph
call msFreeArgs(x, y, z, h)           ! Deallocate mfArray

end Program demo2

```

Line-by-Line Walkthrough

use fml

This includes the module fml which contains the MATFOR numerical procedures.

use fgl

This includes the module fgl which contains the MATFOR visualization procedures.

type(mfArray) :: x, y, z, h

This declares mfArray variables x, y, z, and h.

integer(4) :: i

This declares an integer variable i.

call msMeshgrid(mfOut(x, y), mfLinspace(-3, 3, 30), mfLinspace(-3, 3, 30))

This generates x and y grid matrices from the domains specified by the mfLinspace-generated vectors. This procedure aims to solve functions for two variables by plotting 3D graphs.

- mfOut(x, y) specifies x and y as mfArray outputs.
- mfLinspace(-3, 3, 30) constructs a vector with 30 linearly spaced points between -3 and 3.

do i=1, 10

This creates a loop running from i=1 to i=10.

z = mfSin(x + i/8.0d0) * mfCos(y)

This defines z mathematically. Note that the equation involves the loop index i.

if (i == 1) then

At the beginning of the loop,

h = mfSurf(x, y, z)

This creates a three-dimensional surface plot of the x-, y-, and z-coordinates.

call msDrawNow

This displays the initial figure.

else

At all other times during loop execution,

call msGSet(h, 'zdata', z)

This sets the z-coordinate for update.

call msDrawNow

This displays and updates the figure during loop execution.

end if

This ends the if-else statement.

end do

This ends the loop.

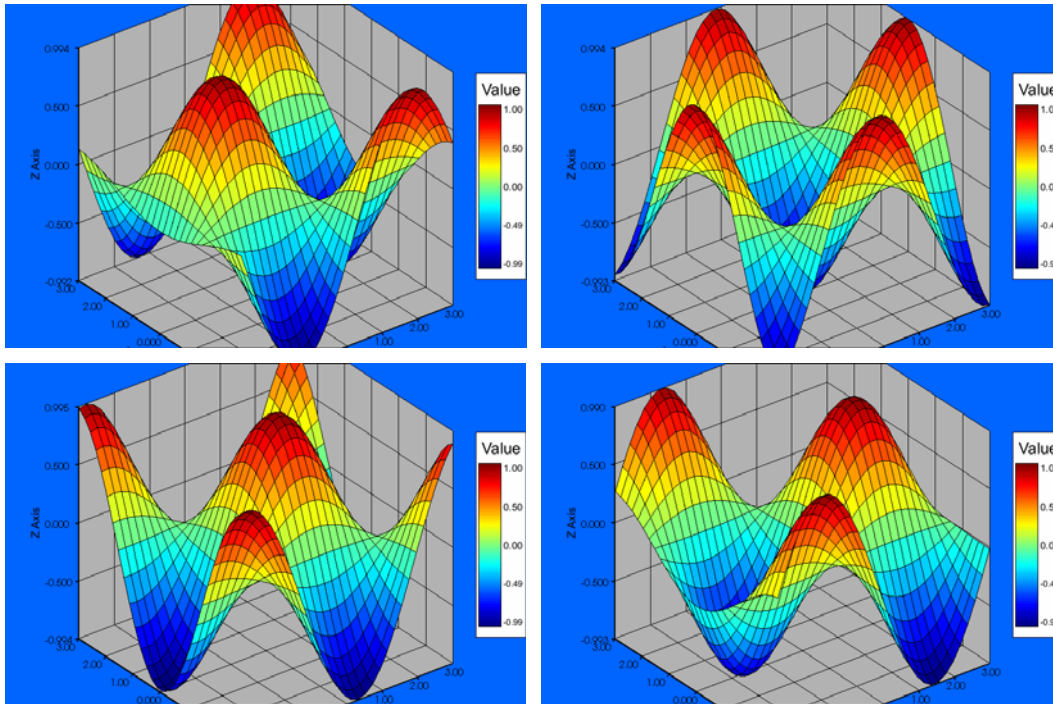
call msViewPause();

This halts program execution for graphical display.

call msFreeArgs(x, y, z)

This deallocates the previously declared mfArray variables.

Compile and Run



III. TO RECORD HELLO SURFACE

This section revises the previous sample code to perform animation recording.

Recording

MATFOR allows recording of animated simulation to facilitate date presentation with comprehensiveness and readability.

To Record Animation

1. Use `fgl` to appropriately contain the visualization routine(s).
2. Use procedures **call msRecordStart('animation.bmp')** and **call msRecordEnd()** before and after the animation codes to record the animation.

Example Code(ch4-3)

Add the recording routines to the previously created “**demo2.f90**” file and rename it “**demo3.f90**” (Note: only the modified and added lines are shown in colors):

Program demo3

```

use fml                                ! Use numerical routines
use fgl                                ! Use visualization routines
implicit none

type(mfArray) :: x, y, z, h            ! Declare mfArray variables
integer(4) :: i                        ! Declare an integer variable
call msMeshgrid( mfOut(x, y), &       ! Construct grid matrices for 3D
plotting
                                mfLinspace(-3, 3, 30), &
                                mfLinspace(-3, 3, 30) )
call msRecordStart( 'demo3.avi' )    ! Begin recording

do i=1, 10                            ! Create loop

    z = mfSin(x+i/8.0d0) * mfCos(y) ! Mathematically define z

    if (i == 1) then                  ! Plot a surf using mfArrays x, y, and z
        h = mfSurf(x, y, z)
        call msDrawNow
    else
        ! Update the figure as the
z-coordinate varies
        call msGSet(h, 'zdata', z)
        call msDrawNow
    end if                            ! End if

end do                                ! End loop

call msRecordEnd                      ! End recording

call msViewPause()                   ! Pause to display the graph

call msFreeArgs(x, y, z, h)          ! Deallocate mfArray

end Program demo3

```

Chapter

5

THE ADVANCED FEATURES

This chapter presents the innovative MATFOR 4 features that are functionally powerful in advanced programming. The first part concerns mfPlayer, which aims to enhance data presentation and accessibility. The second part focuses on MATFOR Graphics Viewer which embraces a full range of manipulation functions. The last part introduces the MATFOR GUI Builder, discussing the application-building facility and integration capability.

I. MFPLAYER

mfPlayer is an exclusive visual tool by which the previously saved numerical data is read and displayed. As MATFOR saves the simulated data into a MATFOR-defined MFA file, mfPlayer is one approach to present the file as a recorded animation.

To record simulated result(s) into an MFA file, simply use procedures **msRecordStart()** and **msRecordEnd()** before and after the animation codes to record the animation:

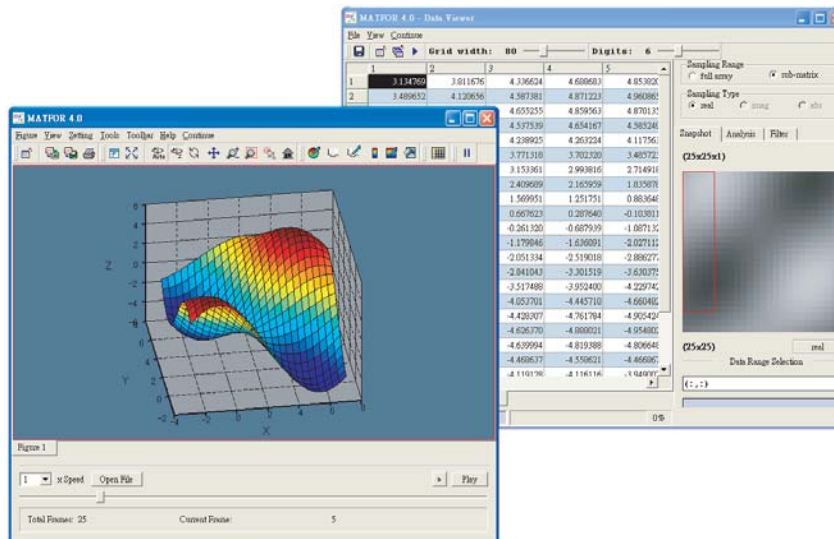
```
call msRecordStart( 'filename.mfa' )
```

```
// animation codes
```

```
call msRecordEnd
```

To play an MFA file with mfPlayer.

1. Go to **Start ► Program Files ► MATFOR4 ► Utilities ► mfPlayer**
2. Click **Open File** to find MFA file and click **Play**.



II. GRAPHICS VIEWER

Embracing a friendly user interface, MATFOR Graphics Viewer supports a variety of graphical manipulation functions.

Example Code(ch5-1)

Program demo1

```
use fml
use fgl
implicit none
```

! Use numerical routines
! Use visualization routines

```
type(mfArray) :: a, b, c, x, y, z
type(mfArray) :: indxi, indxj
```

! Declare mfArray variables

```
a = mfLinspace(-3, 7, 51)
b = mfLinspace(-2, 8, 51)
c = mfColon(1, 51)
```

! Construct linearly spaced vector a
! Construct linearly spaced vector b
! Construct a vector mfArray c

```
call msMeshgrid(mfOut(x, y), a, b)
call msMeshgrid(mfOut(indxi, indxj), c)
3D plotting
```

! Construct grid matrices for

```
z = 3 * mfSin((indxi+1)/10) &
    * mfCos((indxj+1)/10) &
    + 2 * mfSin((indxi+indxj)/10)
```

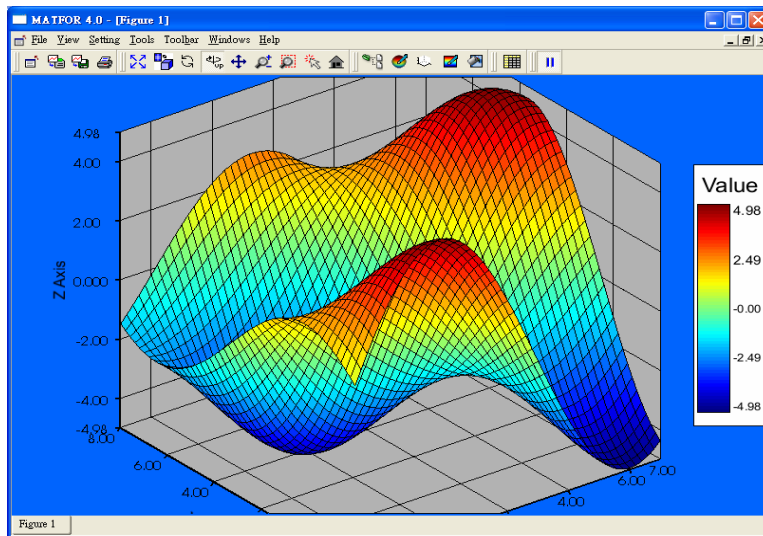
! Mathematically define z

```
call msSurf(x, y, z)
call msViewPause()
call msFreeArgs(a, b, c, x, y, z, indxi, indxj)
```

! Plot a surf using mfArray x, y, and z
! Pause to display the graph
! Deallocate mfArray

```
end Program demo1
```

Compiler and Run

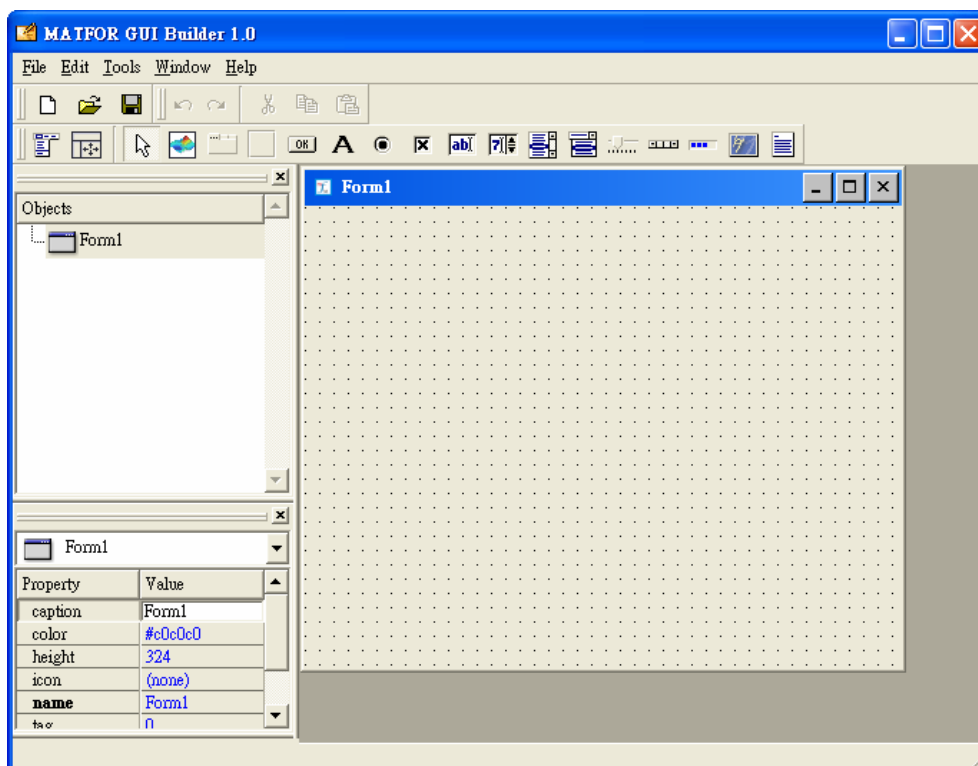


III. MATFOR GUI BUILDER

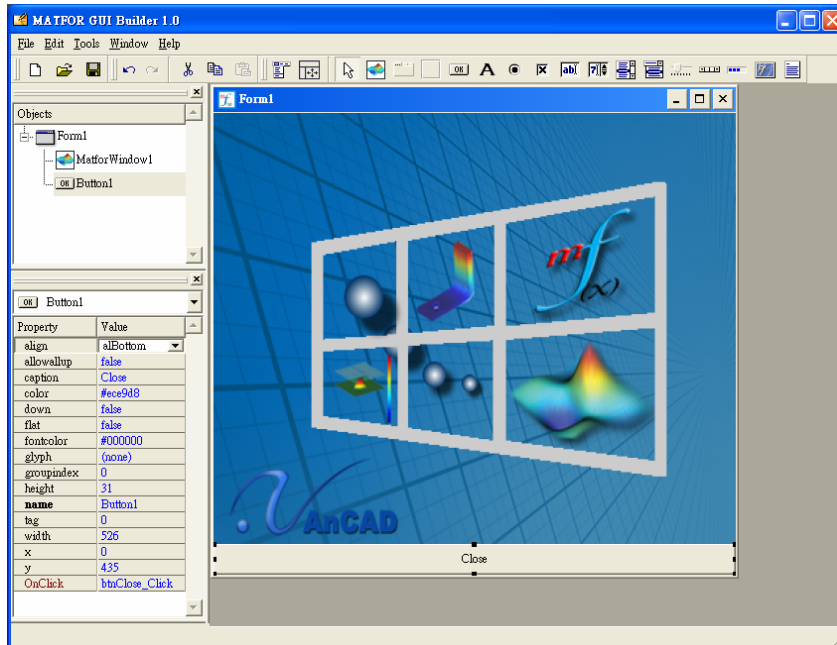
MATFOR GUI Builder allows users to create an interface of their preference, facilitating application-building by packaging the complex codes.

To Build an Application

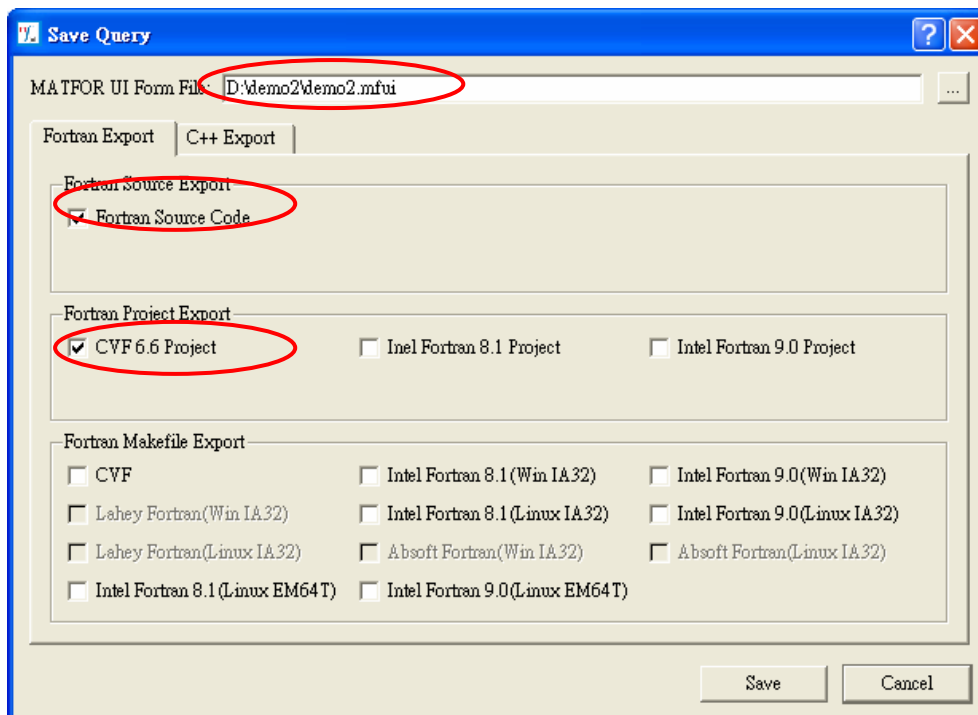
1. Go to Start ► Program Files ► MATFOR4 ► Utilities and run MATFOR GUI Builder.



2. Add **MATFOR Window** and **Button**.
3. Set **Align** to **alBottom**; enter **Close** for **Caption**; and type in **btnClose** for **Name**.
4. Select **OnClick** from **Event Handler** and set it to **btnClose_Click**.
5. Set **MATFOR Widget Align** to **alClient**.



6. Click **Save**.
7. Name the file “**demo2.mfui**” and specify its path.
8. Select the checkboxes **Fortran Main** and **CVF 6.6 Project**.



Open the project “**demo2_cvf.dsp**” and add the following code to “**demo2.f90**”:

```
Subroutine btnClose_Click(sender)
  CHARACTER(*) :: sender
  call exit(0)
end Subroutine btnClose_Click
```

Compile and Run.

