The IMSL® Fortran Numerical Library

Evaluation Guide

Rogue Wave Software
5500 Flatiron Parkway, Suite 200
Boulder, CO 80301, USA
www.roguewave.com

The IMSL® Fortran Numerical Library Evaluation Guide

by Rogue Wave Software

Trademark Information

TABLE OF CONTENTS

Thank you for your interest in evaluating the IMSL® Fortran Numerical Library offered by Rogue Wave Software. Since 1970, IMSL Library products have served as the gold standard for comprehensive and robust mathematical, statistical, and financial libraries. It provides advanced analytics to the technical, scientific, and business communities across a wide range of fields.

This Evaluation Guide is offered to assist you in quickly evaluating the IMSL Fortran Numerical Library. The guide will first discuss the comprehensive documentation supplied with the IMSL Fortran Numerical Library. Then it will build an example program from the documentation for two different development environments: Linux and Visual Studio on Windows.

As a prospective customer, you may already be working with a Technical Support Engineer to whom you can address any questions during the evaluation period. If you do not have a technical contact, please do not hesitate to contact us at EvalSupport@roguewave.com.

Installation of the IMSL® Fortran Numerical Library

Installation of this library is dependent on the development platform (Windows, Unix or Linux). For more information on installing the IMSL Fortran Numerical Library including screenshots, see the online Getting Started Guide.

NOTE >> Keep in mind that the IMSL Fortran Numerical Library requires a Fortran compiler that is listed in the Rogue Wave supported environments, and is usually documented in the ftp download email you received. Each environment is different and requires the compiler the Library was built with.

Installation steps include:

1. Windows

   Using the setup program, it is easy to install the product. Unzip the Windows download file into a temporary folder and run setup.exe to start the installation. If you are installing from CD, the Installer runs automatically once you have inserted the CD-Rom into your workstation.

   Linux/Unix

   After extracting the download tar file into a temporary directory, or mounting the CD-ROM, you will create the installation target directory (e.g., `mkdir /usr/local/vni`) and start the installation program (for example, `/tmpimsl/imsl/install/cd_install`).

2. Once you have started the installation process, as an evaluator simply enter 999999 for your license number when prompted. You may set the install directory or use the default. The example below uses `/usr/local/vni` for Linux and `C:\Program Files\VNI\` for Microsoft Windows.

3. After installing the product, please open and read the README file. You can find this file by browsing to the product installation directory in the 'notes' subdirectory.

4. As part of the IMSL Fortran Numerical Library, you will receive an evaluation license key that should be placed in the license subdirectory in a file named imsl_eval.dat. Refer to the README file for specific instructions related to setup of the evaluation license.

For more details on installation, please consult the Getting Started Guide.

Congratulations! You have successfully installed the IMSL Fortran Numerical Library.

Documentation

After successfully installing the IMSL Fortran Numerical Library, it is recommended that you briefly view the documentation. The IMSL Library documentation is renowned for ease of use and code examples for clear illustrations.

To access the IMSL Fortran Numerical Library documentation, please follow these steps:

- Browse to the product installation directory and then to the `/imsl/fnl710/help` subdirectory.

  imsl.html is the main entry point. Navigate to complete documentation for the math algorithms, special functions, or stat algorithms in either html or pdf format. The DONLP2 Users Guide contains additional documentation for the nonlinear programming functions `NNLPF` and `NNLPG`.

- To get a feel for the content of the documentation, open html IMSL Math Library User Guide, expand Chapter 8, Optimization, and the Linear Programming function `DENSE_LP`. Here you will find a synopsis of the function call with details on required and optional arguments. The Description section provides a brief summary of the problem type to be solved and the algorithm(s) used. The length of these sections is proportional to the complexity of the problem or algorithm. For longer examples, you can view the sections covering `NNLPF`, which is also located in Chapter 8, or `DSCRM` in Chapter 10, Discriminant Analysis, of the IMSL Stat Library User Guide.

- Let us return to the `DENSE_LP` section of the Math documentation. Following the Description section yields several code examples to solve typical linear programming problems.

Congratulations! We hope that you have found the documentation to be helpful, easy to use and thorough. The IMSL Fortran Numerical Library documentation comprehensively includes algorithm explanations and references, graphs and illustrations and customers around the world view it as a very valuable resource.

Building an Example

Now that you have successfully installed the IMSL Fortran Numerical Library and have become familiar with the documentation, you can now put the IMSL Fortran Numerical Library to some use. For that purpose, we will provide two examples. Example 1 uses a regular text editor and the command line in a linux environment. Example 2 uses the Visual Studio 2012 IDE for Windows with the Intel Fortran compiler.

NOTE >> `<env>`, used below, is the environment mnemonic and will be different for each platform.

A Straightforward Example for the Linux Environment

1. For your example application, you will use some code from the help file. Open the API documentation, and again, you will need to browse to the DENSE_LP function. Find the code for Example 1 which resides in the Examples section near the end of the documentation for that routine. Highlight the code in the HTML or PDF document and save the file, *linprogex.f90* to a working directory (let us assume `/usr/local/ex`). This example solves a straightforward linear programming problem subject to several constraints.

2. After saving the file, open a Command Shell and navigate to the directory where the file is saved: "`cd /usr/local/ex`". Next, the environment for the IMSL Fortran Numerical Library must be configured:

   C Shell: `source /usr/local/vni/imsl/fnl710/<env>/bin/fnlsetup.csh`

   bash, K Shell: `. /user/local/vni/imsl/fnl710/<env>/bin/fnlsetup.sh`

   To compile and link the example, the environment variables configured in the previous step can be used:

   ```
   $FC $F90FLAGS linprogex.f90 –o linprogex $LINK_FNL
   ```

   This will use the shared library; to use the static library, replace `$LINK_FNL` with `$LINK_FNL_STATIC`.

3. The compile step creates a *linprogex* executable file that you can now run:

   ```
   ./linprogex
   ```

4. The output of this command should be the following text:

   ```
   Objective  -3.5000

                     Solution
         1        2        3        4        5        6
     0.500    1.000    0.000    1.000    0.500    0.000
   ```

A Straightforward Example Using Visual Studio

In this example the main installation directory is referred to as `<VNI_DIR>`. By default, this is `C:\Program Files\VNI`, but it could have been installed in a different location. If you are unsure of where the IMSL Fortran Numerical Library is installed refer to the environment variable `%VNI_DIR%`.

NOTE >> `<env>`, used below, is the environment mnemonic and will be different for each platform.

In this example, we use Visual Studio 2012 with the Intel Fortran compiler.

1. To begin, you will create a new Visual Studio Solution. Close all open Solutions and choose File ==> New ==> Project. Under Project Types, select Intel(R) Visual Fortran, Console Application as the project type. Under Templates, choose "Empty Project", fill in the name of the project, select the desired location, and click OK.

2. The next step is to add a source code file to your project. You can choose any of the hundreds of coding examples in the Fortran Numerical Library Help Files, but for this example, use the small example described in the Linux section above, saving it in the validate directory listed below. Then, in the main menu, select Project ==> Add Existing Item... and browse to the file:

   ```
   <VNI_DIR>\imsl\fnl710\<ENV>\examples\validate\linprogex.f90
   ```

   and click Open to add it to the project.

3. Next, you must specify the project properties required for IMSL Fortran Numerical Library.

   Select Project ==> Properties

   At the top of Property Pages, click Configuration Manager.

   Click the Active Solution Platform list, and then select the <New> option to open the New Solution Platform Dialog Box.

   Click the Type or select the new platform drop-down arrow, and select x64.

   Click OK. The platform you selected in the preceding step appears under Active Solution Platform in the Configuration Manager dialog box.

   Click Close in the Configuration Manager dialog box.

   Staying on Property Pages, go to Configuration Properties on the left side.

   Under Fortran -> General, add `<VNI_DIR>\imsl\fnl710\<ENV>\include\dll` to the Additional Include Directories.

Under Fortran -> Language, set Process openMP Directives to Generate Parallel Code (/Qopenmp).

Under Fortran -> Floating Point, check that the default setting for Floating Point Exception Handling is selected. It should be set to "Produce NaN, signed infinity, and denormal results"

Under Linker -> General, add `<VNI_DIR>\imsl\fnl710\<ENV>\lib` to the "Additional Library Directories" list and then click OK.

NOTE >> If you did not allow the IMSL Fortran Numerical Library Setup program to update the System Environment Variables, this lib directory must be added to the environment variable PATH from Control Panel, prior to starting Visual Studio.

4. Add an include statement at the top of your source code that will direct the compiler to pick up the necessary libraries:

```
include 'link_fnl_shared.h'
```

5. To compile the file and link in the IMSL Fortran Numerical Library, select Build ==> Build Solution from the menu bar. You should see "Build: 1 succeeded, 0 failed, 0 skipped" in the Output window of the IDE.

6. The build step creates an executable file that you can run by selecting Debug ==> Start Without Debugging from the menu bar.

7. The output should be the same as described in the Linux example above.

Congratulations! In only a few short minutes, you have used the IMSL Fortran Numerical Library to solve a Linear Programming optimization problem. This technique is widely applicable for financial engineering, operations research, and R&D product and process optimization, to name a few. Conservative estimates show the IMSL Fortran Numerical Library can save close to 95% of the development costs organizations would incur if they were to build algorithms from scratch. Numerical algorithms are very expensive to research, build, and maintain. Organizations can realize tremendous productivity gains by relying on fully tested and documented routines from IMSL.

Closing

Thank you for your interest in Rogue Wave's IMSL Fortran Numerical Library. We hope that this brief evaluation guide has broadened your understanding and demonstrated a few key benefits about the IMSL Fortran Numerical Library. If you would like a technical expert to help

you through a more complex code example or if you have any questions or need any technical assistance, please contact us at evalsupport@roguewave.com.