Compiling GMAT using Visual Studio 2010

Darrel J. Conway*
Thinking Systems, Inc.
Tucson, AZ

April 16, 2013

Abstract

This document describes the steps needed to build GMAT using Visual Studio 2010 (VS2010). The instructions start with a fresh installation of VS2010, provide directions for installing wxWidgets, and finally for building GMAT.

1 Introduction

The General Mission Analysis Tool, GMAT, is a space trajectory optimization and mission analysis system developed by NASA and private industry in the spirit of the NASA Vision. GMAT contains new technology and is a testbed for future technology development. To satisfy NASA's mandate and maximize technology transfer, GMAT is an open source software system licensed under the NASA Open Source Agreement. Interested parties are encouraged to use GMAT to plan spacecraft missions, to build the system when they have requirements for features not yet included in GMAT, and to contribute new capabilities to the system either through direct contributions to the code base or through plugin libraries.

This document describes the steps a new developer takes to set up a build environment for GMAT using Microsoft's Visual Studio development system. The instructions were written using Visual Studio 2010 Express Edition, and validated using both the Express Edition and the Professional Edition. The following instructions assume that the developer is running a Windows XP, Vista, or Windows 7 based computer. Separate instructions are available for users building GMAT with the GNU Compiler Collection (gcc) on Windows, Mac, or Linux based computers.

The following sections describe installation of the compiler, folder arrangements and code used in the GMAT build files, preliminary steps necessary to collect and build the libraries GMAT needs, and the build steps for GMAT itself. The document concludes with instructions for plugin libraries that GMAT uses to interface with MATLAB and the MATLAB Optimization toolbox, to perform estimation (the estimation plugin is a preview of capabilities still in development), and the VF13ad optimizer (core code available separately).

2 Installing Visual C++ / Visual Studio 2010 Express Edition

The Visual C++ environment can be installed either from a web-based installer or from disk. Users of the paid versions of Visual C++ should skip this section and install the product following the instructions provided by Microsoft. Users of the Express edition can install from the online instructions provided by Microsoft, or by following the instructions provided here.

*Support provided by NASA GSFC FDSS Contract, Task 28.

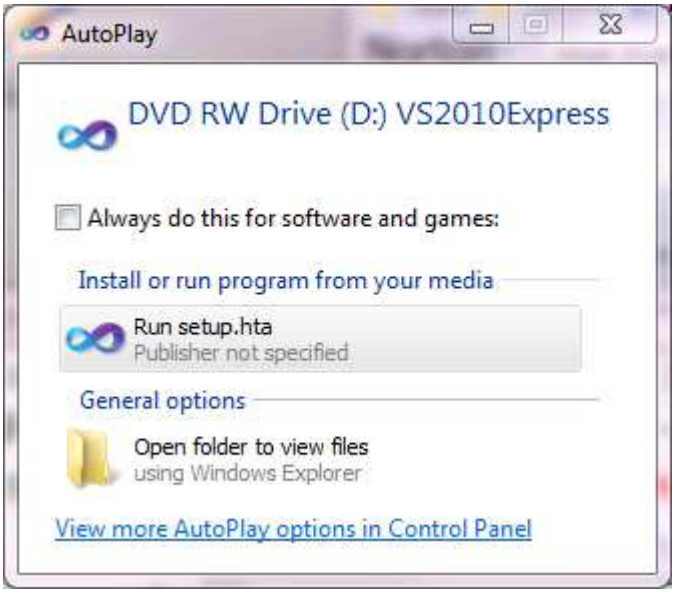

Figure 1: Screen for a New Disk

2.1 Option 1: Web installation

Visual C++ can be installed using a web based installer by following these steps:

1. Open a web browser and browse to the Visual Studio 2010 Express download page: http://www.microsoft.com/express/Downloads/#2010-Visual-CPP
2. Download the installer for Visual C++
3. Open the installer (vc_web.exe) by double clicking on it
4. Follow the installation instructions

2.2 Installing from a Disk Image

If you have a disk containing Visual Studio Express, follow these steps:

1. Put the installation disk into your DVD drive
2. If prompted, select "Run Setup.hta" from the Autoplay Menu (See Figure 1)
3. Select Visual C++ 2010 Express from the menu that opens (See Figure 2)
4. Follow the installation instructions. GMAT does not require SQLServer, so you can deselect that option for installation.

2.3 64-bit Compilers (Optional)

If you plan to build the 64-bit version of GMAT[1], you'll need to install the Windows 7 SDK version 7.1. You'll also need to be running on a 64-bit version of Windows so you can test the resulting build.

[1]The GMAT development team does not maintain the 64-bit builds of the system. Project settings and source code files for 64-bit builds are not guaranteed to work without some user interaction to correct changes that may not have been checked in 64-bit compilations.

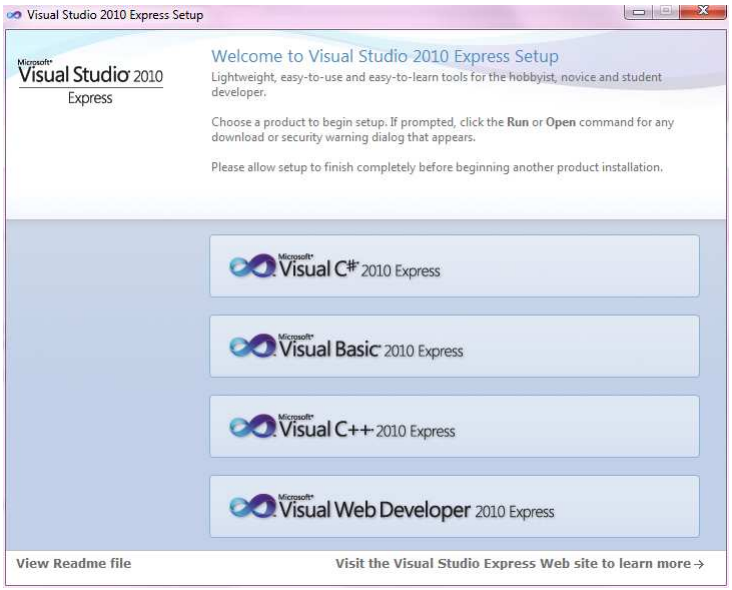

Figure 2: Select Visual C++ Here

The SDK can be downloaded from (all on one line)

```
http://www.microsoft.com/downloads/en/details.aspx?FamilyID=6b6c21d2-2006-4afa-
9702-529fa782d63b\&displaylang=en
```

Download the web based installer by clicking on the "Download" button, and then run it. This installs the software development kit, including a 64-bit compiler.

3 Folder Configuration

Create a folder for building GMAT. I'll use C:\GmatVS as the root folder in this document. This folder will contain the subfolders for third party components of the build – wxWidgets, cspice, pcrecpp, and similar files. Add the following folders to your root folder:

- Gmat3rdParty
- GmatDevelopment
- GmatPlugins (if you will be using the configuration managed plugin libraries)

Inside of the Gmat3rdParty folder, create the following subfolders; for the 32-bit builds:

- cspice
- f2c
- pcre
- wxWidgets

At this point, your folder structure should match the structure shown in Figure 3. If you want to also build the 64-bit release and have either a paid version of Visual Studio 2010 (that is, Professional, Premium, or Ultimate), or if you installed the 64-bit tools in the Windows SDK, add these folders for the 64-bit versions of the 3rd party tools:

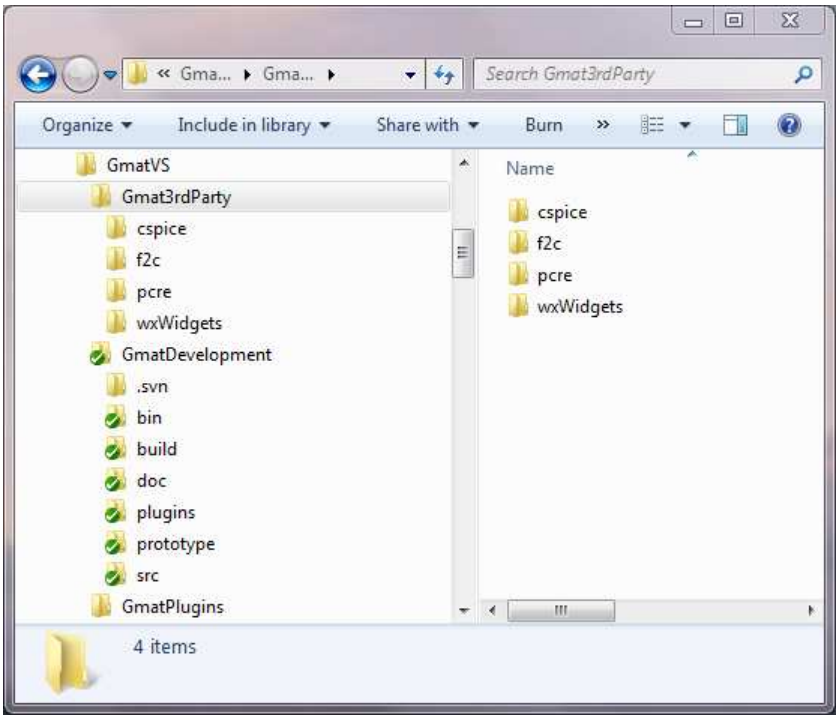

Figure 3: The File Structure used in GMAT's Configuration Managed Build Files

- cspice64
- f2c64
- pcre64
- wxWidgets64

4 Downloading GMAT

GMAT's source code is located in a Subversion repository at SourceForge. In addition to the source code, the repository at SourceForge contains the build files we'll need to proceed, including wxWidgets project files configured for Visual Studio 2010. Because of this, you need to begin by downloading the GMAT development code using a Subversion client. I am using SmartSVN on Windows, but any client should work. Set up your client to access code at the following URL:

https://svn.code.sf.net/p/gmat/code/trunk

Check out the entire source tree from that location into the GmatDevelopment folder you created earlier. When you check out the tree, you'll retrieve GMAT's source code, development files, documentation, source for several plugin libraries, and the support files needed to run GMAT.

Alternatively, if you can access Subversion from a command prompt, you can download the code directly by setting to your current diractory to the root folder containing the GmatDevelopment and Gmat3rdParty folders and then entering the command

svn checkout svn://svn.code.sf.net/p/gmat/code/trunk GmatDevelopment

5 Building wxWidgets

From this point through Section 9 we will concentrate on the 32-bit build of GMAT. Appendix C describes the settings and changes necessary to build GMAT as a 64-bit application. We begin by building wxWidgets, the GUI toolkit used for GMAT's user interface.

5.1 Install the wx Code

1. Download wxWidgets 2.8 from http://wxwidgets.org/downloads/. Use the latest stable version of the 2.8 tree (2.8.12 at this writing), and download the wxMSW zip package.
2. Unpack the wx code into the folder named wxWidgets in your Gmat3rdParty folder. You should see folders named art, build, contrib, demos, docs, include, lib, locale, samples, src, and utils in the wxWidgets folder when finished.
3. Visual Studio 2010 has trouble processing the Visual C++ project files that ship with wx. Instead, we'll use files already configured for wx at Thinking Systems.
   (a) The Visual Studio 2010 solution and project files for wxWidgets are located in an archive file in the GMAT folders you checked out earlier. Find the file wxWidgets.zip in the build\windows-VS2010 folder of that working copy of the code.
   (b) Unpack the wxWidgets.zip archive into your wxWidgets folder, overwriting existing files as necessary.
   (c) Check to see that the file wx_dll.sln is in your Gmat3rdParty\wxWidgets\build\msw folder. (This step ensures that all of the build files are properly positioned in the wxWidgets folder structure. The key files are wx_dll.sln in the Gmat3rdParty\wxWidgets\build\msw folder and wxcontrib_dll.sln in the Gmat3rdParty\wxWidgets\contrib\build folder.)

5.2 Prepare and Build wxWidgets

1. Open Visual Studio
2. Select "File — Open — Project/Solution..." from the menu bar
3. Open the wx_dll.sln solution file from the build\msw folder of your wxWidgets installation. (For me, the file is found in C \GmatVS\Gmat3rdParty\wxWidgets\build\msw)
4. GMAT needs the wxWidgets OpenGL components as part of the build. This setting is configured in the wx\setup.h configuration file for the build. We need to change a setting in this file from its default setting.
   (a) Open one of the projects in the Visual Studio solution tree
   (b) Open the Setup Headers folder in this project
   (c) Open the first setup.h file in the folder by double clicking on it
   (d) If this file has the line "\\Name: wx/univ/setup.h" at the top, open the other setup file. (You want to edit the file in msw rather than in univ; the file you want identifies itself as "wx/msw/setup.h".)
   (e) Search for wxUSE_GLCANVAS in the open file
   (f) Set wxUSE_GLCANVAS to 1 rather than the default value of 0
   (g) Save and close the file.
5. On the Visual C++ toolbar select "DLL Release" (specifying the build configuration) and set your solution platform to Win32. Figure 4 shows these settings.

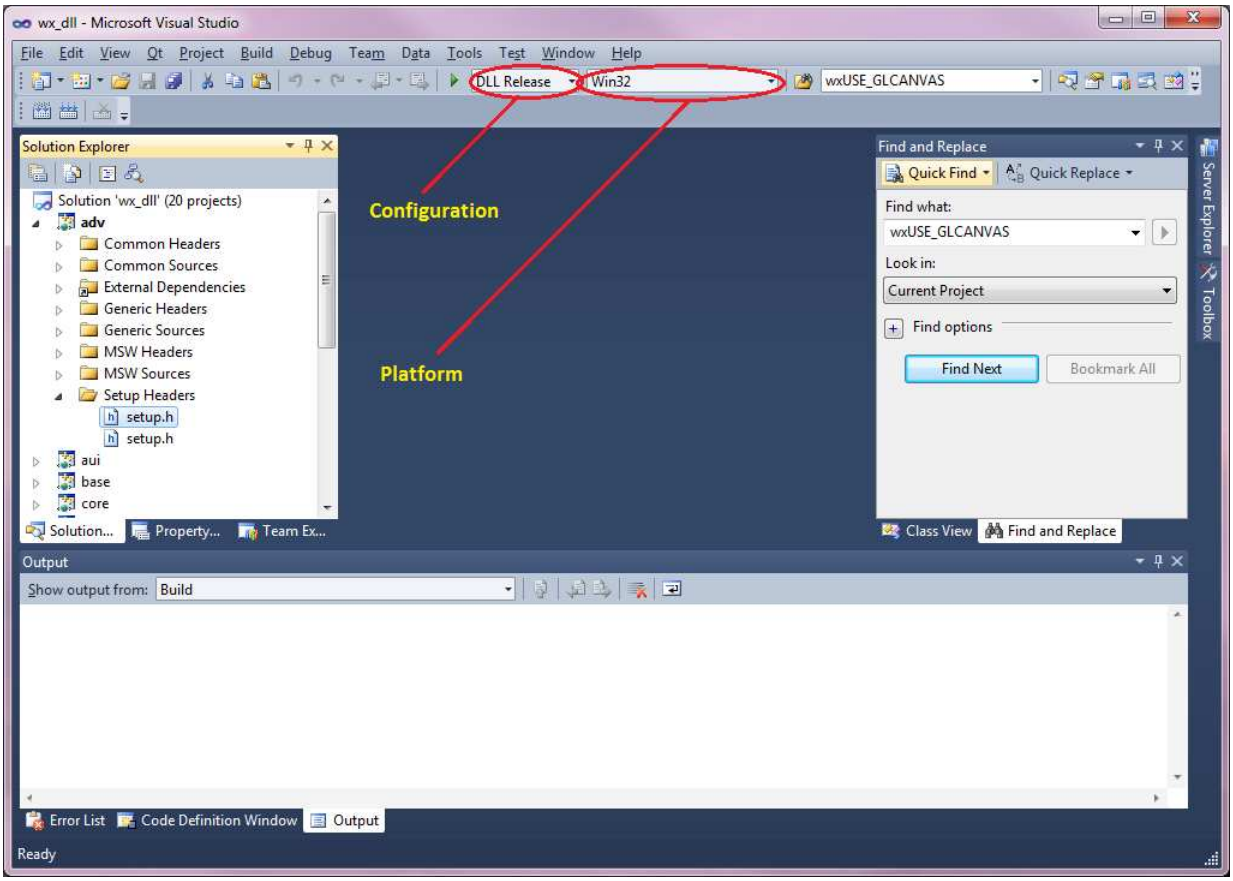

Figure 4: Configuration and Platform Settings for the 32-bit wxWidgets Build

6. Build the solution (right click on the "Solution wx_dll" node at the top of the tree and select "Build Solution".)

7. Most, but not all, of the solution will build. You may need to repeat the build process to get all of the libraries GMAT requires built. Build the solution again until all but the dbgrid projects build. (You are done when 19 of the 20 projects build.)

8. Select "File — Open — Project/Solution..." from the menu bar

9. Open the wxcontrib_dll.sln solution file from the contrib\build folder of your wxWidgets installation. (For me, the file is found in C \GmatVS\Gmat3rdParty\wxWidgets\contrib\build)

10. On the Visual C++ toolbar select "DLL Release" (specifying the build configuration) and set your solution platform to Win32.

11. Build the solution (right click on the "Solution wxcontrib_dll" node at the top of the tree and select "Build Solution".)

Check to see that you have 15 dll files in the wxWidgets\lib\vc_dll folder:

- wxbase28_net_vc_custom.dll
- wxbase28_odbc_vc_custom.dll

- wxbase28_vc_custom.dll
- wxbase28_xml_vc_custom.dll
- wxmsw28_adv_vc_custom.dll
- wxmsw28_aui_vc_custom.dll
- wxmsw28_core_vc_custom.dll
- wxmsw28_gl_vc_custom.dll
- wxmsw28_html_vc_custom.dll
- wxmsw28_media_vc_custom.dll
- wxmsw28_netutils_vc_custom.dll
- wxmsw28_qa_vc_custom.dll
- wxmsw28_richtext_vc_custom.dll
- wxmsw28_stc_vc_custom.dll
- wxmsw28_xrc_vc_custom.dll

Copy these into your GmatDevelopment\application\bin folder.

5.3 (Optional) Preparing for Debug Builds

The GMAT solution file can be used to build a debug version of GMAT. In order to use that version, you will need to build a set of wxWidgets debug libraries. The procedure is the same as for the release build as described above, with the following changes:

- On the Visual C++ toolbar select "DLL Debug" rather than "DLL Release" as the solution configuration.
- After building the libraries, the debug versions will all have the letter "d" added to the library name: wxbase28d_net_vc_custom.dll, wxbase28d_odbc_vc_custom.dll, etc. Copy these files into your GmatDevelopment\application\debug folder.

6 Preparing SPICE

GMAT includes readers and writers for data in the SPICE kernel format supplied by the Jet Propulsion Laboratory (JPL). In order to build GMAT, you'll need to add the SPICE libraries to your system. To do this, follow these steps:

1. Download the SPICE toolkit for Visual C. The 32-bit edition is available from

   http://naif.jpl.nasa.gov/naif/toolkit_C_PC_Windows_VisualC_32bit.html

   and the 64-bit version from

   http://naif.jpl.nasa.gov/naif/toolkit_C_PC_Windows_VisualC_64bit.html

   You want the file named cspice.zip.

2. Unpack the archive into your Gmat3rdParty\cspice (or cspice64 for 64-bit) folder. You should unpack the files so that you have, for example, a Gmat3rdParty\cspice\include folder that contains the cspice header (.h) files.

Note that cspice includes Fortran to C (f2c) library code, so you do not need to manage f2c separately when working with the GMAT base library code.

7 Building GMAT

At this point, GMAT should build without further configuration. Follow these steps:

1. Open Visual C++
2. Select "File — Open — Project/Solution..." from the menu bar
3. Browse to the GmatDevelopment\build\windows-VS2010 folder
4. Select the GmatVS2010.sln solution file and click the Open button. The GMAT solution will open and show projects to build GMAT and several plugin libraries.
5. The solution has two configurations, depending on how you want to proceed: Debug and Release. Select the configuration that you would like to build.
6. Right click on the top node in the Solution Explorer, and select Rebuild to completely build GMAT including all of the plugins, but excluding the console application. This process takes a few minutes.
7. Open Windows Explorer and browse to your GmatDevelopment\application\bin or your GmatDevelopment\application\debug folder based on the configuration you selected. You should see GMAT.exe there, along with libGmatBase.dll and your wxWidgets libraries.
8. Double click on the GMAT.exe file to run GMAT. Press the Run button to run the default mission. Rejoice.

8 Building the Plugin Libraries for MATLAB, fmincon, Estimation, and the C-Interface (Optional)

At this writing there are four plugin libraries included in GMAT's trunk code on SourceForge. Two of these, libMatlabInterface and libFminconOptimizer, require a recent version of MATLAB to build. The other two, libGmatEstimation and libCInterface, build in a straightforward manner. We'll look at the latter two first, and then at the MATLAB dependent plugins.

8.1 Building libGmatEstimation and libCInterface

GMAT's source tree includes a project file and code for a GMAT compatible estimation plugin. This plugin includes both a Kalman filter and a batch estimator, along with a measurement simulator and several measurement models. At this writing, these components are in an early alpha form. As the component evolves, the build process should remain the same. GMAT's estimation plugin includes tropospheric and ionospheric models. The ionosphere model was originally coded in Fortran, and then imported into the GMAT plugin using the f2c utility program. For that reason, you must have either f2c or SPICE installed to build the estimation plugin.

The source tree also includes code for a C interface into GMAT's object-oriented engine in the shared library libGmatBase. The C-interface is being used on other projects to provide access to select components of GMAT's engine for external use. This access is contained in the libCInterface plugin.

Follow these instructions to build libGmatEstimation and libCInterface:

1. Building libGmatEstimation
   (a) Open Visual C++
   (b) Select "File — Open — Project/Solution..." from the menu bar
   (c) Browse to the GmatDevelopment\build\GmatVS2010 folder

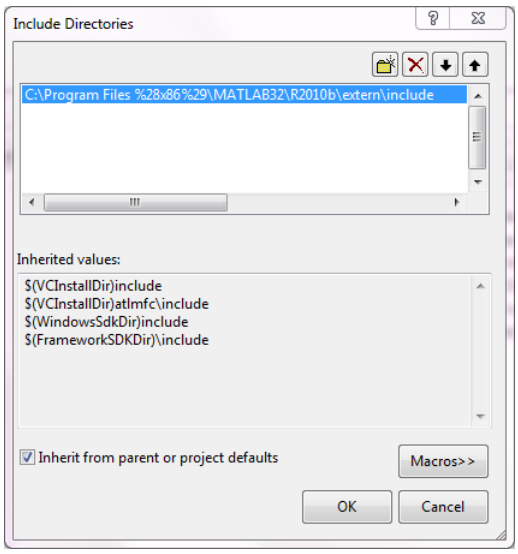

Figure 5: MATLAB Include Settings

(d) Select the GmatVS2010.sln solution file and click the Open button. The GMAT solution will open and show projects to build GMAT and several plugin libraries.

(e) Right-click on the libGmatEstimation node, and select "Build"

2. Building libCInterface

(a) Open Visual C++

(b) Select "File — Open — Project/Solution..." from the menu bar

(c) Browse to the GmatDevelopment\build\GmatVS2010 folder

(d) Select the GmatVS2010.sln solution file and click the Open button. The GMAT solution will open and show projects to build GMAT and several plugin libraries.

(e) Right-click on the libCInterface node, and select "Build"

Once each build finishes, the plugin libraries will be copied to the appropriate target folders. The release build of libGmatEstimation is copied into the application/plugins folder, and the release build of libCInterface, into the application/bin folder.

8.2 The MATLAB and fmincon Plugins

Next we will build the GMAT MATLAB interface. You'll need to have MATLAB installed to build the MATLAB interface, and you'll also need the Optimization Toolbox to build the fmincon optimization plugin.

1. Open Visual C++
2. Select "File — Open — Project/Solution..." from the menu bar
3. Browse to the GmatDevelopment\build\GmatVS2010 folder
4. Select the GmatVS2010.sln solution file and click the Open button. The GMAT solution will open and show projects to build GMAT and several plugin libraries.

5. Set your include and library paths for the libMatlabInterface and libFminconOptimizer projects. This is done by following these steps for each project:

   (a) Right click on the project node (either libMatlabInterface or libFminconOptimizer)

   (b) Select the "Properties" entry on the pop-up menu

   (c) On the dialog that appears, set the Configuration to "All Configurations"

   (d) Select the "Configuration Properties — VC++ Directories" entry on the property tree in the left panel of the dialog

   (e) Select the "Include Directories" line, and select <Edit...>from the drop-down button that sets the directory list

   (f) A new dialog appears that lists the include directory settings for at the project scope. Click on the first entry in the list, and then click on the button to its right labeled with ellipses (...)

   (g) A file browser will appear. Navigate in this browser to your MATLAB folder, and then inside of it, navigate to the "extern\include" folder. This folder contains the header files GMAT needs to build the interface. Click the "Select Folder" button to select it.

   (h) Once the file browser window closes, your include dialog should resemble the one shown in Figure 5. Press the OK button.

6. Right click on the libMatlabInterface project, and select Build. (Don't select Rebuild unless you want to also rebuild libGmatBase.dll).

7. Right click on the libFminconOptimizer project, and select Build. (Don't select Rebuild unless you want to also rebuild libGmatBase.dll and libMatlabInterface.dll).

8. Open a windows explorer and browse to your GmatDevelopment\application\bin folder. You should see libMatlabInterface.dll and libFminconOptimizer.dll in the folder now.

9. Double click on the GMAT.exe file to run GMAT. fmincon will now appear as an option in the Solvers — Optimizers entry of the resource tree.

9 Standard GMAT Plugins

There are three additional plugin libraries that are built for GMAT at this writing. If you have access to the plugin repositories, you can configure and build these optional libraries: libDataFile, libCcsdsEphemerisFile, and libVF13Optimizer. The VF13ad optimizer library, like the Estimation library described above, uses the Fortran to C library, so if you plan to build it, follow the steps above to setup either f2c or SPICE. The DataFile plugin and the CcsdsEphemerisFile plugin use the C++ extensions to the Perl Compatible Regular Expression library, pcrecpp. Configuration of that component is described next.

9.1 Configuring and Building PCRECPP (Optional)

GMAT uses the C++ version of the Perl Compatible, pcrecpp, for string manipulation when working with data files and CCSDS based ephemeris products. The following steps describe how to set up pcrecpp.

1. Download the pcre 8.12 (to guarantee project file consistency – preassembled Visual C++ 2010 solutions are packaged in the GMAT repository; other versions may require some modification to the build configuration) from ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/

2. Unpack the download into your Gmat3rdParty/pcre folder. This will give you a folder named pcre-<version>, where <version>is the version number for the library. (For me, the resulting folder is Gmat3rdParty/pcre/pcre-8.12.)

3. Copy the contents of pcre-8.12 into the pcre folder, moving them up one level. This places the header files in the location needed to build the plugins.

4. The usual way to build pcre is to download and install CMake for your platform, use it to create build files (Makefiles on unix like systems, project files and solution files on Windows), and then build from those files. The Windows pcre library files have already been created for pcre-8.12, and can be found in the GMAT plugins build folder build\windows-VS2010 in the archive pcre-VS2010-out.zip.

5. Locate pcre-VS2010-out.zip.

6. Unpack its contents into your pcre folder. This will create a lib and a dll folder in your pcre folder.

7. Copy the contents of the pcre\dll folder into your GMAT executable folder.

This completes the pcre setup. If you'd prefer to compile pcre yourself, the instructions for doing this using CMake can be found in the pcre\NON-UNIX-USE file. Just be sure that you place the resulting libraries in place in pcre\lib and pcre\dll.

9.2 Building the DataFile, CCSDS Ephemeris File, and VF13ad Plugins

Instructions to be written as needed

References

[1] Visual Studio Express can be downloaded from http://www.microsoft.com/express/Downloads/#2010-Visual-CPP

[2] http://wxwidgets.org/downloads/

[3] http://naif.jpl.nasa.gov/naif/index.html

A GMAT Data Files and Support Files

GMAT requires access to data files organized in a file structure described in the GMAT startup file, gmat_startup_file.txt. That file should be in the folder from which GMAT is launched. In a typical setup, the GMAT startup file identifies a root folder, a data file folder, and then uses these two locations to identify the locations of additional data files needed by GMAT. A typical startup file contains these lines for the top level folders:

```
ROOT_PATH               = ../
DATA_PATH               = ROOT_PATH/data/
```

The remaining lines in the file describe the locations of planetary ephemerides, coefficient files for planetary gravity fields, time system data, and other data elements needed when GMAT runs. A typical file structure for the data files is shown in Figure 6. This file structure matches the structure delivered with GMAT R2011a. It also matches the data file structure found in the source code repository for GMAT at SourceForge at the time of this writing. The entries in the GMAT startup file point to specific files in this structure. For example, the lines

```
SPK_PATH                = DATA_PATH/planetary_ephem/spk/
PLANETARY_SPK_FILE      = SPK_PATH/de421.bsp
```

identify the location of the SPICE planetary ephemeris file, de421.bsp, in the file system. Using the default entries provided thus far, GMAT looks for the file

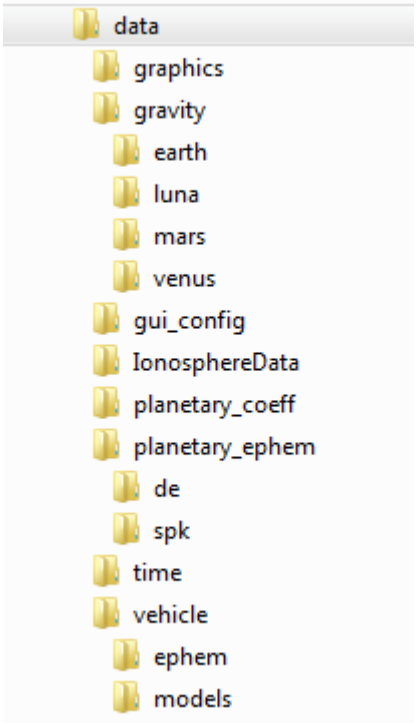

Figure 6: The Default Data File Strucure for GMAT

```
../data/planetary_ephem/spk/de421.bsp
```

relative to the starting folder for the system. The startup file provides similar path data for all of GMAT's data files; interested users should look in this file when trying to understand the data used when running a mission.

B Library Versions

At this writing, the software used to populate the Gmat3rdParty folder is as follows:

- wxWidgets, 2.8.12
- The SPICE Toolkit, N0064
- MATLAB, R2011a
- pcre, 8.12
- f2c, built from source last updated Sep 3, 2010

These versions are the target versions for GMAT R2011b.

C Building GMAT in 64-bit mode

To be written as needed