Setting up 64-bit ADMB for Windows with the Microsoft Visual C++ 2008 compilers

Allan Hicks
Allan.Hicks@noaa.gov
June 15, 2010

This is an installation guide for setting up the 64-bit version of ADMB for Windows on a single computer using the compiler from Microsoft Visual Studio 2008. This setup differs from the setup described on the admb-project website. I have chosen to eliminate the batch scripts that are designed to set up environment variables (such as the PATH) and to manually enter these variables such that they are persistent and automatically available when any command window is opened. This greatly increases my ability to work with ADMB, especially when opening command prompts in specific windows to quickly compile ADMB code, or when compiling ADMB code from within text editors such as textpad.

This guide also recommends that the 32-bit version of ADMB is installed. The 32-bit installation is easier and it allows you to compile programs that will run on 32-bit Windows operating systems. Programs compiled in 64-bit can allocate more memory and work with larger integers, but does not guarantee that the program will run faster, although they often do. See my 32-bit installation guide for instructions on setting up 32-bit ADMB.

NOTE: *These instructions indicate the way that I set up ADMB on my computers and I do not guarantee that they will work on your computer. In addition, because of different versions of SDK and Microsoft.NET, these instructions may need to be modified slightly depending on your versions. This guide will likely be updated in the future.*

ADMB installation

AD Model Builder (ADMB) can be downloaded from the ADMB project website (`http://admb-project.org`) and is available for 32-bit and 64-bit Windows Visual Studio compilers. This guide does not necessarily assume that you will want to install both 32-bit and 64-bit compilers, and can be used to install only the 64-bit compiler. Remember, to use the 64-bit compiler and executables compiled with the 64-bit compiler, you must have a 64-bit Windows operating system.

1. Install the 64-bit ADMB (admb-9.1-windows-vc9-64bit)
    - Available from `http://admb-project.org/downloads`
    - Install it into a folder that is accessible by all users (i.e., do not install in the Program Files directory if using it with non-administrative accounts). To keep 32-bit and 64-bit installations separate, install them into their own directories. I assume that you will install 64-bit ADMB into the directory `C:\ADMB\Microsoft\VC9\64bit`.

There is no need to run `set-admb-vc9.bat`. We'll set these environment variables up such that they are persistent for all users.

Install Visual Studio 2008 compilers

There are two ways to install the Visual Studio 2008 64-bit compilers onto your computer.

1. Install the paid for version of Visual Studio 2008
    - This will by default install 32-bit compilers and you need to check the option to install 64-bit compilers.
2. Install Microsoft SDK for Windows 7 with .NET Framework 3.5 SP 1.
    - Download it from this link SDKdownload or search for `Windows SDK 7 .NET 3.5 SP 1`
    - You can download the web installer (recommended) or the ISO to burn to a CD for later use.
    - Is free and includes both 32-bit and 64-bit Visual Studio 2008 compilers.
    - Does not include an IDE, but compiles ADMB code via the command line.
    - Works with the Visual Studio Express IDE if you wish to use that environment.
    - To greatly reduce download and installation time, do not install documentation and examples (uncheck these options)

Visual C++ Express includes only the 32-bit compilers. However, it can be installed and used as an IDE if you wish. However, I do not explain how to set it up for 64-bit compiling, although it is possible to do by calling the correct batch scripts. Microsoft Visual C++ Express is available from `http://www.microsoft.com/express/vc/`. See the 32-bit ADMB installation guide for more information.

I assume that you will use option 2 above and install the SDK (software development kit). If you wish to use the Visual Studio Express IDE, install Visual Studio Express first then install the SDK for Windows 7. If you use the paid for Visual Studio 2008, you do not need to install the SDK, but do need to make sure that you select the 64-bit compilers when installing Visual Studio 2008.

You do not need to install Visual Studio 2008 or Visual Studio 2008 Express to compile ADMB code. The compilers are installed with the SDK and a Microsoft Visual Studio 2008 folder is created in the Program Files directory.

Setting up the environment variables for ADMB and Visual Studio C++ compilers

When compiling ADMB code, the compiler needs to know where to look for header and library files. Environment variables are used to indicate where these files are. To set these environment variables for the entire system (all users) you will need administrator access. Below, I describe how to set these environment variables for only the logged in user. This does not require an administrator account.

1. Manual setup of environment variables for ADMB
   (a) Right click on "My Computer" and select Properties, or choose "System" from the Control Panel.
   (b) Click on the "Advanced" tab at the top
   (c) Choose "Environment Variables" near the bottom
      - There are two windows here: "User variables for ..." and "System variables". We will add to the "User Variables". However, the variables will not be available to other users. If multiple users may use ADMB, I suggest creating these variables under the "System Variables" section.
   (d) Under "User variables for ..." at the bottom, check to see if `ADMB64_HOME` is present.
      i. If not, choose `New`. If it is present, select `ADMB64_HOME` and choose `Edit`
      ii. "Variable name:" `ADMB64_HOME`
      iii. "Variable value:" type in the directory you chose for ADMB 32-bit during installation.
         - For example, I typed in `C:\ADMB\Microsoft\VC9\64bit`. Make sure `ADMB64_HOME` isn't set anywhere else, and if it is set under "System variables," delete it or modify it. Keep `ADMB_HOME` if it is present, as this points to the 32-it installation.
      i. Under "User variables for ...," choose `Path` and click on `Edit` if it exists, or select texttNew.
      ii. "Variable name:" `PATH`
      iii. In "Variable Value" enter `C:\ADMB\Microsoft\VC9\64bit\bin` at the very end of all the text.
      iv. If PATH exists and other directories are listed, precede the directory you entered with a semicolon (;) to separate the directories. And, **make sure to keep the current text there or serious problems could occur!**

2. Manual setup of environment variables for Visual C++ 2008 Express
   (a) Open the "Environment Variables" window as in 1a–c) above
   (b) Under "User Variables", check to see if `MSSDK` is present
      i. If it is not, choose `New` under "User Variables". If it is, select it and choose `Edit`
      ii. "Variable name:" `MSSDK`
      iii. "Variable Value:" the directory of the SDK. This should be `C:\Program Files\Microsoft SDKs\Windows\v7.0` if you installed the SDK above, but may be different if you installed Visual Studio with 64-bit compilers.
         - Verify that this directory exists and contains a folder called "Lib" and a folder called "x64" within "Lib". If not, you may have to search for the directories or install the programs to obtain the 64-bit libraries.
         - Make sure `MSSDK` is not set anywhere else, and if it is set under "System variables," delete it or modify it by deleting all directories and adding only the above directory.
   (c) Under "User Variables", check to see if there is a variable called "Include"
      i. If there is not, choose `New`. If there is, click on Include and choose `Edit`
      ii. "Variable name:" `Include`
      iii. "Variable Value:" add in the directory `C:\Program Files (x86)\Microsoft Visual Studio 9.0\VC\INCLUDE;`
         - You may want to verify that this directory exists and is where these files were installed. If not, you may have to search for the directories or install the programs.
   (d) Under "User Variables," choose `Path` and click on `Edit`.
      i. In "Variable Value" enter the following directories separated by semicolons `;C:\Program Files (x86)\Microsoft Visual Studio 9.0\Common7\IDE;` at the very end of all the text (after `C:\ADMB\Microsoft\VC9\64bit\bin`).
         - **Make sure to keep the current directories there or serious problems could occur!**

This should allow you to open a command window in any directory and immediately compile an admb tpl file without setting up any variables or navigating to various directories.

Batch files for ADMB 64-bit compiling

To call the 64-bit Visual Studio 2008 compiler, you must modify the batch scripts in ADMB that do the compiling and linking. Three batch files need to be created and place in the directory `C:\ADMB\Microsoft\VC9\64bit\bin`. These files are very similar to the batch files already present for 32-bit compiling except that they have a few lines added and/or changed. The easiest way to do this is to create a copy of the batch files mentioned and rename them to the 64-bit versions. The names and contents of the files are as follows, with changes and additions marked in red. Make sure that the directories for Microsoft Visual Studio correspond to where your version was installed. If it is different, you will have to change the batch files to reflect your installation.

admb64.bat

Create a file called `admb64.bat` from `admb.bat` and make it look like the following.

```
@echo off
setlocal
if [%1]==[] goto HELP
if [%1]==[-help] goto HELP
if [%1]==[--help] goto HELP
REM ##############################################################################################################
REM                                                                                                              #
REM Script:   admb [-d] [-r] [-s] model                                                                          #
REM                                                                                                              #
REM Purpose:  Build ADMB executable from TPL, using tpl2cpp/tpl2rem, adcomp, and adlink                          #
REM                                                                                                              #
REM Args:     -d creates DLL                                                                                     #
REM           -r creates ADMB-RE                                                                                 #
REM           -s uses safe bounds and debugging symbols                                                          #
REM           model is the filename prefix, e.g. simple                                                          #
REM                                                                                                              #
REM Requires: ADMB header files and libraries, tpl2cpp, tpl2rem, adcomp, adlink                                  #
REM                                                                                                              #
REM Returns:  Creates executable or DLL with same prefix                                                        #
REM                                                                                                              #
REM History:  24 May 2009  Arni Magnusson created                                                                #
REM           15 June 2010 Allan Hicks modified for 64-bit compiling in Windows when 32-bit is also present      #
REM                                                                                                              #
REM ##############################################################################################################

rem Pop args until model=%1
set d=
set r=
set s=
set tpl2cpp=tpl2cpp
set bounds=
set dll=
:STARTLOOP
if [%2]==[] goto ENDLOOP
if %1==-d set d=-d& set dll=-dll& shift
if %1==-r set r=-r& set tpl2cpp=tpl2rem& shift
if %1==-s set s=-s& set bounds=-bounds& shift
goto STARTLOOP
:ENDLOOP

echo.& echo *** %tpl2cpp% %bounds% %dll% %1
                %tpl2cpp% %bounds% %dll% %1
echo.& echo *** adcomp64 %d% %r% %s% %1
           call adcomp64 %d% %r% %s% %1
echo.& echo *** adlink64 %d% %r% %s% %1
           call adlink64 %d% %r% %s% %1
goto EOF

:HELP
echo Usage: admb [-d] [-r] [-s] model
echo.
echo Build AD Model Builder executable from TPL.
echo.
echo   -d     Create DLL
echo   -r     Create ADMB-RE
echo   -s     Use safe bounds and debugging symbols
echo   model  Filename prefix, e.g. simple
echo.

:EOF
```

adcomp64.bat

Create a file called `adcomp64.bat` from `adcomp.bat` and make it look like the following.

```
@echo off
setlocal
if [%1]==[] goto HELP
if [%1]==[-help] goto HELP
if [%1]==[--help] goto HELP
REM ##################################################################################################################
REM Script:   adcomp [-d] [-r] [-s] model                                                                          #
REM                                                                                                                #
REM Purpose:  Compile ADMB C++ to object code, using the MinGW GCC 3.4.5 compiler                                  #
REM                                                                                                                #
REM Args:     -d creates object file for DLL                                                                       #
REM           -r creates object file for ADMB-RE                                                                   #
REM           -s creates object file with safe bounds and debugging symbols                                        #
REM           model is the filename prefix, e.g. simple                                                            #
REM                                                                                                                #
REM Requires: ADMB header files, g++                                                                               #
REM Returns:  Creates object file with same prefix                                                                 #
REM                                                                                                                #
REM History:  24 May 2009  Arni Magnusson created                                                                  #
REM           15 June 2010 Allan Hicks modified for 64-bit compiling in Windows when 32-bit is also present        #
REM ##################################################################################################################

REM Added by Allan Hicks to call 64 bit compiler when 32 bit also installed
@SET OLDPATH=%PATH%
@SET PATH=C:\Program Files (x86)\Microsoft Visual Studio 9.0\VC\BIN\amd64;%PATH%

rem Pop args until model=%1
set g=
set dll=
set opt=-DOPT_LIB
:STARTLOOP
if [%2]==[] goto ENDLOOP
if %1==-d set dll=-DBUILDING_DLL& shift
if %1==-r shift
REM if %1==-s set g=-g& set opt=& shift
if %1==-s set g=-g& shift
goto STARTLOOP
:ENDLOOP

@echo on
cl -c /EHsc -DUSE_LAPLACE -DWIN32 %opt% /Ox -D__MSVC32__=8 -I. -I"%MSSDK%"\include -I"ADMB64_HOME"\include %1.cpp
@echo off

REM Added by Allan Hicks to call 64 bit compiler when 32 bit also installed
@SET PATH=%OLDPATH%

goto EOF

:HELP
echo Usage: adcomp [-d] [-r] [-s] model
echo.
echo Compile AD Model Builder C++ to object code, using the MinGW GCC 3.4.5 compiler.
echo.
echo   -d     Create object file for DLL
echo   -r     Create object file for ADMB-RE
echo   -s     Create object file with safe bounds and debugging symbols
echo   model  Filename prefix, e.g. simple
echo.

:EOF
```

adlink64.bat

Create a file called `adlink64.bat` from `adlink.bat` and make it look like the following.

```
@echo off
setlocal
if [%1]==[] goto HELP
if [%1]==[-help] goto HELP
if [%1]==[--help] goto HELP
REM ##############################################################################################################
REM Script:   adlink [-d] [-r] [-s] model                                                                        #
REM                                                                                                              #
REM Purpose:  Link ADMB object code to executable, using the MinGW GCC 3.4.5 compiler                           #
REM Args:     -d creates DLL                                                                                     #
REM           -r creates ADMB-RE                                                                                 #
REM           -s uses safe bounds and debugging symbols                                                          #
REM           model is the filename prefix, e.g. simple                                                          #
REM Requires: ADMB libraries, g++, dllwrap                                                                       #
REM Returns:  Creates executable or DLL with same prefix                                                        #
REM History:  24 May 2009  Arni Magnusson created                                                                #
REM           15 June 2010 Allan Hicks modified for 64-bit compiling in Windows when 32-bit is also present      #
REM ##############################################################################################################

REM Added by Allan Hicks to call 64 bit compiler when 32 bit also installed
@SET OLDPATH=%PATH%
@SET OLDLIB=%LIB%
@SET PATH=C:\Program Files (x86)\Microsoft Visual Studio 9.0\VC\BIN\amd64;%PATH%
@SET LIB=C:\Program Files (x86)\Microsoft Visual Studio 9.0\VC\LIB\amd64

rem Pop args until model=%1
set re=0
set s=-s
set def=
set df1b2lib=df1b2stubo.lib
set adlib=ado32.lib
:STARTLOOP
if [%2]==[] goto ENDLOOP
if %1==-r set re=1& shift
REM if %1==-s set s=& set adlib=ads32.lib& set df1b2lib=df1b2stubs.lib& shift
if %1==-s set s=& shift
goto STARTLOOP
:ENDLOOP
if %re%==1 if %adlib%==ado32.lib set df1b2lib=df1b2o.lib
if %re%==1 if %adlib%==ads32.lib set df1b2lib=df1b2s.lib

@echo on
cl  %1.obj %df1b2lib% admod32.lib %adlib% adt32.lib /link /libpath:"%ADMB64_HOME%"\lib /libpath:"%MSSDK%"\lib\x64
@echo off

REM Added by Allan Hicks to call 64 bit compiler when 32 bit also installed
@SET PATH=%OLDPATH%
@SET LIB=%OLDLIB%
goto EOF

:HELP
echo Usage: adlink [-d] [-r] [-s] model
echo.
echo Link AD Model Builder object code to executable, using the MinGW GCC 3.4.5 compiler.
echo.
echo   -d     Create DLL
echo   -r     Create ADMB-RE
echo   -s     Use safe bounds and debugging symbols
echo   model  Filename prefix, e.g. simple
echo.

:EOF
```

Extras

I have created a small registry editor file which allows you to right-click on a folder and open a command prompt set up to work in that folder. For example, I can right-click on the "simple" folder of the examples, open a command prompt and immediately compile the simple.tpl file. You can create this by adding these lines to a text file and renaming it `CmdHere.reg`

```
Windows Registry Editor Version 5.00
[HKEY_CLASSES_ROOT\Folder\shell\Cmd Here]
@="Command &Prompt Here"
[HKEY_CLASSES_ROOT\Folder\shell\Cmd Here\command]
@="cmd.exe /k pushd %L"
```

then run this file by double clicking on it. Since this modifies the registry, you must do this in an account with administrator access.

If problems occur

1. make sure that none of the variables created above (`ADMB_HOME` etc) are masked in the User variables box of the "Environment Variables"
2. make sure that the directories added to the variables are the actual directories
3. e-mail me (Allan.Hicks@noaa.gov) and I can see what I can do