https://github.com/leonard-psu/PIHMgis

How to Compile PIHMgis in both Windows and Linux environments.

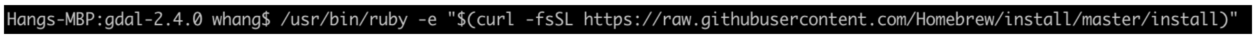
1. Build and install QT5.12 or higher. You will need to use QTCreator.
2. Build and install GDAL from https://github.com/OSGeo/gdal
3. Build and install Sundials-2.7.0 from https://computation.llnl.gov/projects/sundials/download/sundials-2.7.0.tar.gz
4. Build and install Triangle from https://www.cs.cmu.edu/~quake/triangle.html
5. Download PIHMgis from github.
6. Open/Import PIHMgis.pro using QTCreator.
7. Clean all.
8. Add Library and Include Windows or Linux paths for gdal (libgdal_i.a or dll), sundials (libsundials_nvecserial.a or dll, libsundials_cvodes.a or dll), to the PIHMgis.pro using QTCreator "Add Library".
9. Add Include folder for sundials (under *the install folder*/include) to the PIHMgis.pro using QTCreator "Add Include". Sometimes, the QTCreator would automatically add the Include which may be not right path. Need to pay attention to that.

****

Download *GDAL* package from the website:
https://trac.osgeo.org/gdal/wiki/BuildHints

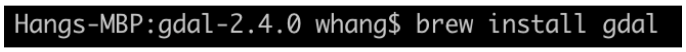
Following the instruction: https://trac.osgeo.org/gdal/wiki/BuildingOnMac

1. Follow the instructions to install Homebrew: https://brew.sh/



```
Hangs-MBP:gdal-2.4.0 whang$ /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

2. Compile and install the current release version of gdal



```
Hangs-MBP:gdal-2.4.0 whang$ brew install gdal
```

3. After a while, GDAL is installed in the mac under the hidden folder (Cmd + Shift + . (dot) to show hidden files):

/usr/local/Cellar/gdal/2.4.0/

4. The *library and Include* for gdal ((libgdal.a or dll) is under: /usr/local/Cellar/gdal/2.4.0/lib

****

***Sundials*, need to be compiled using the source package**
https://cmake.org/download/

Follow the install guide in the package:

1. create a new folder that will contain the installment of SUNDIALS and follower the steps below:

```
% mkdir (...)sundials/instdir
% mkdir (...)sundials/builddir
% cd (...)sundials/builddir
```

(…) is the path where the folder instdir and builddir is created.

2. To build the default configuration using the GUI, under the folder *builddir*, enter the *ccmake* command and point to the *srcdir*:

```
% ccmake ../srcdir
```

.. is the path where the downloaded and unzipped Sundials folder with the name of scrdir

3. Using CMake with the GUI follows this general process:

• *Select and modify values, run configure (c key)*

• *New values are denoted with an asterisk*

• *To set a variable, move the cursor to the variable and press enter*

– If it is a boolean (ON/OFF) it will toggle the value – If it is string or file, it will allow editing of the string

– For file and directories, the <tab> key can be used to complete

• *Repeat until all values are set as desired and the generate option is available (g key)*

• *Some variables (advanced variables) are not visible right away*

• *To see advanced variables, toggle to advanced mode (t key)*

• *To search for a variable press / key, and to repeat the search, press the n key*

4. Pressing the (g key) will generate makefiles including all dependencies and all rules to build sundials on this system. Back at the command prompt, you can now run:

```
% make
```

To install SUNDIALS in the installation directory specified in the configuration, simply run:

```
% make install
```

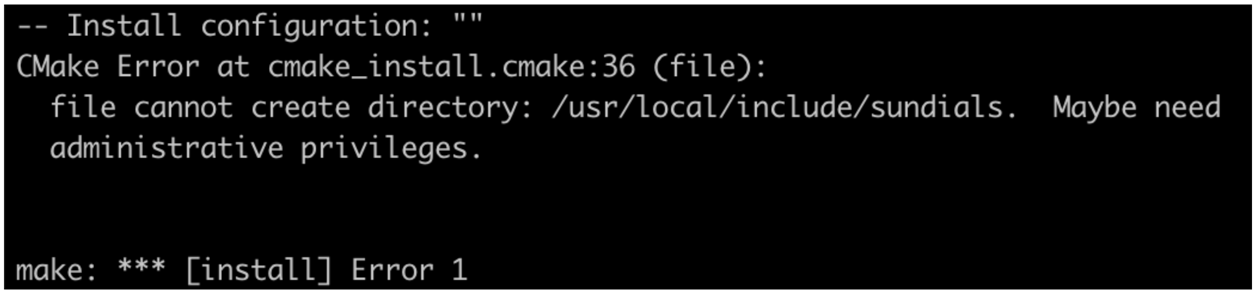
5.

Building from the command line

Using CMake from the command line is simply a matter of specifying CMake variable settings the cmake command. The following will build the default configuration:

```
% cmake -DCMAKE_INSTALL_PREFIX=/home/myname/sundials/instdir \
> -DEXAMPLES_INSTALL_PATH=/home/myname/sundials/instdir/examples \
> ../srcdir
% make
% make install
```

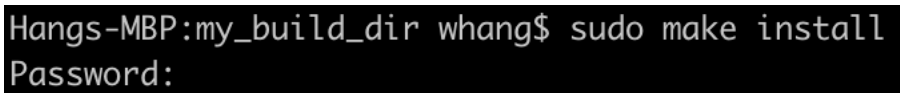
If encountering the problem like this:



```
-- Install configuration: ""
CMake Error at cmake_install.cmake:36 (file):
  file cannot create directory: /usr/local/include/sundials.  Maybe need
  administrative privileges.


make: *** [install] Error 1
```

Please use the command and enter your mac password:



```
Hangs-MBP:my_build_dir whang$ sudo make install
Password:
```

After installing Sundials, you can find sundials (libsundials_nvecserial.a or dll, libsundials_cvodes.a or dll) under the folder

```
% cd (...)sundials/builddir
```

The subfolders for adding Library (*mine: /documents/my_build_dir*):
../src/nvec_ser: libsundials_cvodes.a
../scr/covde: libsundials_nvecserial.a
The subfolder for adding Include (*mine: /documents/my_build_dir*):

…/include

Here, … is the install folder.

****

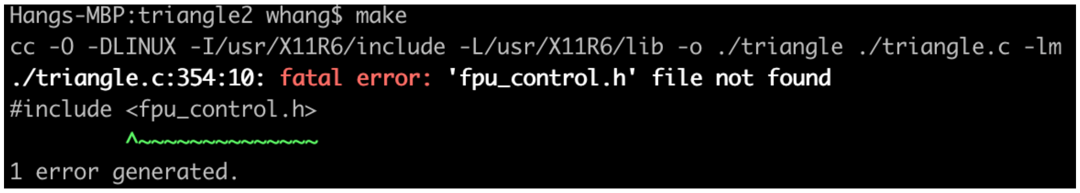
Build and install Triangle from https://www.cs.cmu.edu/~quake/triangle.html

1. Download the source code Triangle from https://www.cs.cmu.edu/~quake/triangle.html

Triangle (version 1.6, with Show Me version 1.6) is available as a .zip file (159K)

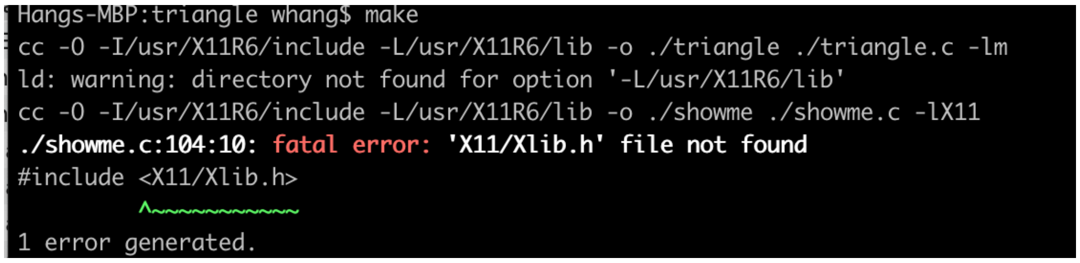
2. Compile the code using *make* under the folder with source code

If error occurs,



```
Hangs-MBP:triangle2 whang$ make
cc -O -DLINUX -I/usr/X11R6/include -L/usr/X11R6/lib -o ./triangle ./triangle.c -lm
./triangle.c:354:10: fatal error: 'fpu_control.h' file not found
#include <fpu_control.h>
         ^~~~~~~~~~~~~~~
1 error generated.
```

Or



```
Hangs-MBP:triangle whang$ make
cc -O -I/usr/X11R6/include -L/usr/X11R6/lib -o ./triangle ./triangle.c -lm
ld: warning: directory not found for option '-L/usr/X11R6/lib'
cc -O -I/usr/X11R6/include -L/usr/X11R6/lib -o ./showme ./showme.c -lX11
./showme.c:104:10: fatal error: 'X11/Xlib.h' file not found
#include <X11/Xlib.h>
         ^~~~~~~~~~~~
1 error generated.
```

Need to edit the Makefile from:

```
# An example CSWITCHES line is:
#
#   CSWITCHES = -O -DNO_TIMER -DLINUX -I/usr/X11R6/include -L/usr/X11R6/lib

CSWITCHES = -O -DLINUX -I/usr/X11R6/include -L/usr/X11R6/lib
```

To

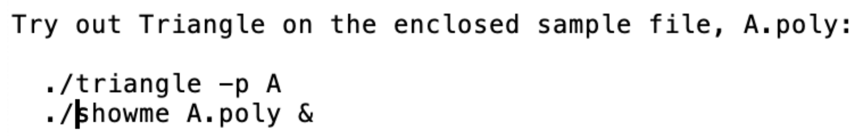
```
# An example CSWITCHES line is:
#
#   CSWITCHES = -O -DNO_TIMER -DLINUX -I/usr/X11R6/include -L/usr/X11R6/lib

CSWITCHES = -O -I/usr/X11/include -L/usr/X11/lib
```

Note that the /usr/X11/include and /usr/X11/lib is the folder where includes X11 in your mac. X11 is from the installment of XQuartz.

After fixing this, Triangle should be able to compiled by ***make***

3. Done. You can test if the installment is successful or not:



```
Try out Triangle on the enclosed sample file, A.poly:

  ./triangle -p A
  ./showme A.poly &
```