1 Installation Guide for MMTK and OpenMM

This installation guide has been developed to work for Mac OS X Snow Leopard for MMTK 2.7.8 and OpenMM 4.1.1

Created by: Chris Ing and Stephen Constable
Updated by: Nabil Faruk and Kevin Bishop

1.1 Pre-installation Requirements

1. Xcode

   You will require XCode to be installed on your Mac. It can be found in the additional installs directory of the discs that come with your Mac. X11 is another requirement if you need to use GnuPlot. Note: If you do not have the install discs, you can download Xcode from apple at: https://developer.apple.com/xcode/

2. Development Directory

   Create a new directory for Python binaries. A common one is $HOME/Dev . You should not use the default python installation since files in /Library/Python/2.*/site-packages can be modified by Apple patches.

1.2 MMTK Installation

1. Python

   - Download the latest compressed source tarball from http://www.python.org/download of Python 2.*
   - Extract this tarball to a temporary directory with:
     `tar -xf Python2.*.tar`
     and cd to that directory
   - Bootstrap your installation with the following command:
     `./configure -prefix=$HOME/Dev`
   - Build and install with the command:
     `make install`

   Note: You must make your new python accessible from the command line:

   - Modify your ~/.bash_profile script by adding the following line:
     `alias pydev='"$HOME/Dev/bin/python" $*'`
   - Then reload your bash_profile by entering:
     `source ~/.bash_profile`

   Now, when you type pydev at the command line, your new python will open.

2. Cython

   - Download the latest tar file and extract by entering:
     `tar -xf Cython-0.17.2.tar.gz`

- Install Cython by entering (inside the Cython directory):
  ```
  pydev setup.py install
  ```
  Ensure that you use pydev as we do not want to modify the original python environment. This will be critical for the rest of the future installations as well.

3. zlib

   - Download zlib 1.2.5 from ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-4/
   - Extract by entering:
     ```
     tar -xf zlib-1.2.5.tar.gz
     ```
   - Build and install by entering the following 2 lines:
     ```
     ./configure -prefix=/Users/kpbishop/Dev
     make check install
     ```
     Note: kpbishop should be your own username

4. HDF5

   - Download HDF5 1.8.6 from ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-4/
   - Extract by entering:
     ```
     tar -xf hdf5-1.8.6.tar.gz
     ```
   - Build and install by entering the following 2 lines:
     ```
     ./configure -with-zlib=/Users/kpbishop/Dev -prefix=/Users/kpbishop/Dev
     make check install
     ```

5. netCDF

   - Download the latest C netCDF source from:
     http://www.unidata.ucar.edu/downloads/netcdf/index.jsp
     netCDF used to come for C, C++ and Fortran but we only need the C portion. The most recent release at time of writing is the netCDF C library and utilities, version 4.2.1.1
   - Extract this file by entering:
     ```
     tar -xf netcdf4.*.tar.gz
     ```
   - Build and install netcdf and specify the locations of zlib and HDF5 with the following 4 commands:
     ```
     CPPFLAGS=-I/Users/kpbishop/Dev/include
     LDFLAGS=-L/Users/kpbishop/Dev/lib
     ./configure -prefix=/Users/kpbishop/Dev
     make check install
     ```

6. NumPy

   - Download the latest version of NumPy as a tar.gz (currently 1.6.2)
   - Extract it using the following line:
     ```
     tar -xf numpy-1.6.2.tar.gz
     ```

- To build and install NumPy, enter:
  pydev setup.py install

7. Scientific Python

   - Download the latest version of Scientific Python as a tar.gz (currently 2.9.1) from http://sourcesup.cru.fr/projects/scientific-py/
   - Extract is using the following line:
     tar -xf ScientificPython-2.9.1.tar
   - To build and install SciPy, enter:
     pydev setup.py install
   - Test your installation using any of the scripts in the Examples folder, try protein.py in /Examples/MolecularDynamics/protein.py with:
     pydev protein.py

8. MMTK

   - Download the latest development release of MMTK from:
     http://sourcesup.cru.fr/projects/mmtk
   - Extract MMTK using:
     tar -xf MMTK-2.*.tar
   - Build and install MMTK with: (This setup file may need to be changed depending on your install path)
     pydev setup.py install

Notes:

- To run the Langevin Dynamics example script in version 2.7.1 of MMTK, you may need to manually add the netCDF include directory to the setup.py script by changing Line 20:
  include_dirs=['./'])
  changed to:
  include_dirs=['./','/Users/kpbishop/Dev/include'])
- then the Langevin integrator can be compiled with:
  pydev setup.py build_ext -inplace
- and the example can be run with:
  pydev example.py
- In some cases fftw may be required to be installed.
  - You can obtain a tarball from http://www.fftw.org/download.html
    and extract using:
    tar -xf fftw-3.3.3.tar.gz
  - Configure using:
    ./configure –prefix=$HOME/Dev –enable-shared
  - and build and install using:
    make install

1.3 OpenMM Installation

1. Copy over a working OpenMM directory:
   /home/nffaruk/Sugar_GPU/OpenMM4.1.1-Source/
   or try from the OpenMM website
   (https://simtk.org/project/xml/downloads.xml?group_id=161)
2. You will need CMake for the installation process. Download and install it from
   http://www.cmake.org/files/v2.8/cmake-2.8.9-Darwin64-universal.dmg
3. GCC-XML
   - Copy to your machine by entering:
     git clone git://github.com/gccxml/gccxml.git
   - Install it with CMake by cding into the gccxml source directory and using the command:
     /usr/bin/cmake -i
     Choose your Dev folder for the install path when prompted
4. Get and install CUDA toolkit, drivers, and SDK for Mac:
   http://developer.nvidia.com/cuda/cuda-downloads
   May also need CUDA driver from:
   http://www.nvidia.com/object/mac-driver-archive.html

5. Update your bash_profile so that it has the following lines at the bottom:

```
export MMTK_USE_CYTHON=1
export DYLD_LIBRARY_PATH=$HOME/Dev/openmm/lib
export DYLD_LIBRARY_PATH=
    $DYLD_LIBRARY_PATH:$HOME/Dev/openmm/lib/plugins
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:$HOME/Dev/lib
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/usr/local/cuda/lib
export OPENCL_DIR=/System/Library/Frameworks/OpenCL.framework
export OPENMM_PLUGIN_DIR=$HOME/Dev/openmm/lib/plugins
```

   Reload it using:

```
source ~/.bash_profile
```

6. OpenMM

   - You should be able to install OpenMM now by cding to the top level of its source directory and use CMake:
     ```
     /usr/bin/cmake -i
     ```
     In the CMake prompts watch out for:
     - the installation directory... should be ~/Dev/openmm
     - where gcc-xml executable is located.. should be ~/Dev/bin/gccxml
     - whether to build C and fortran wrappers... yes
     - whether to build python wrapper.. no
   - Then build/install with:
     ```
     make OpenMM
     make install
     ```
   - Then test with:
     ```
     make test
     ```
     Remember to 'make clean' if you experience any problems and want to reinstall after making modifications