cGENIE (Cupcake v0.3)
Mac OS X Instructions

Gregory J. L. Tourte

18th June 2015

These instructions have been tested on Mac OS X Yosemite (10.10) but should work on Mavericks (10.9) as well. However we recommend that you use the latest version of the OS available as we may not be able to reproduce errors and bugs you may report.

The following instructions will require you to use the Mac OS X Terminal application. This can be opened by using the ⌘ + Space key combination and searching for '*Terminal*', alternatively it can be found in the `Macintosh HD ▸ Applications ▸ Utilities` directory. We suggest you pin the application to your Dock for easy access if you haven't done so already.

Note: *Throughout this document, shell commands are shown in* `monospace` *font and prefixed with* `$` *(the dolar sign should not be reproduced when typing the command in the terminal). Commands that extend over several lines are marked with an* ↵ *on the following line.*

1 System Requirements

To install and run cGENIE on Mac OS X you will need the following packages installed:

- Apple Xcode
- Xcode Command line Tools
- the MacPorts environment
- the HomeBrew packaging system
- Python and extensions
- gfortran compiler
- NetCDF libraries

1.1 Apple Xcode

Apple Xcode can be downloaded for free from the Apple App Store. More details can be found at `https://developer.apple.com/xcode/downloads/`. XCode contains the GNU C Compiler (`gcc`) and most of the other libraries and tools to allow the compilation of the model.

In addition to Xcode, you will need to install the Xcode Command Line Tools. These used to be installed by default on older versions of Xcode but are now distributed separately. Once Xcode is installed, you will need to run the following command in the terminal:

```
$ xcode-select --install
```

You can check if Xcode is properly installed using the command:

```
$ xcode-select -p
```

which should contain the following line in the response:

```
/Applications/Xcode.app/Contents/Developer
```

While we are checking the environment, you can check that `gcc` is installed properly:

```
$ gcc --version
Configured with: --prefix=/Applications/Xcode.app/Contents/Developer/usr
 ↪ --with-gxx-include-dir=/usr/include/c++/4.2.1
Apple LLVM version 6.1.0 (clang-602.0.53) (based on LLVM 3.6.0svn)
Target: x86_64-apple-darwin14.3.0
Thread model: posix
```

1.2 Homebrew & MacPorts

Homebrew is a package manager for OS X which allows users to download and install packages found in other UNIX style environment such as Linux and keep them up to date in a managed way in the sense that one can update packages and manage dependencies. MacPort has a similar aim but with a slightly different philosophy. We will not compare these two here as it goes beyond the scope of this document but we have used both to install software required to run cGENIE on the Mac.

Homebrew and information about it can be found at `http://brew.sh/`. To install the environment, simply type the following line in the Terminal:

```
$ ruby -e "$(curl -fsSL
 ↪ https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

MacPort and information about it can be found at `https://www.macports.org/` and instructions on how to install it are found at `https://www.macports.org/install.php`. Following the installation, you should make sure your installation is fully up to date by running:

```
$ sudo port -v selfupdate
```

1.3 Python

The install scripts, running script as well as the gui for cGENIE are written in `python` and require version 2.x but preferably v2.7.x. Mac OS X comes with `python` installed by default since version 10.8 (Mountain Lion) and appears to satisfies the minimum version requirements (Yosemite comes with version 2.7.6). However, the version on Yosemite doesn't seems to be quite adequate for some of the scripts within cGENIE, especially the gui. You will therefore need to install another version of `python`. The version provided by homebrew is known to work (whereas the one from macport is known not to) so you will need to install the homebrew version even if you already have the macport package (they are no installed in the same location so should not clash. If you are happy with the homebrew version, remove the macport one to simplify things).

```
$ brew install python
```

Once installed, you will need to specify the location of your python environment via the `$CGENIE_PYTHON` environment variable for cGENIE to use it. Using the homebrew version of python you need to add the following line to your `.bash_profile` file:

```
export CGENIE_PYTHON="/usr/local/bin/python2.7"
```

You should also run that same line from the command line if you do not wish to have to open a new terminal.

```
$ echo $CGENIE_PYTHON
/usr/local/bin/python2.7
$ (exec $CGENIE_PYTHON --version)
Python 2.7.10
```

The newly developed gui requires additional python modules such as `matplotlib`. These modules can be installed as normal python modules, i.e., you can either use the source tarballs or use a packaging system such as `pip` or `easy_install`. Using pip, you should just need to do:

```
$ pip install matplotlib
$ pip show matplotlib
---
Metadata-Version: 2.0
```

```
Name: matplotlib
Version: 1.4.3
Summary: Python plotting package
Home-page: http://matplotlib.org
Author: John D. Hunter, Michael Droettboom
Author-email: mdroe@stsci.edu
License: BSD
Location: /usr/local/lib/python2.7/site-packages
Requires: numpy, pytz, pyparsing, python-dateutil, nose, six, mock
```

If `matplotlib` is already installed, you may want to make sure that it and all its dependencies are up to date by using the command:

```
$ pip install --upgrade matplotlib
```

Since `python`, and therefore pip, are now install from homebrew, you should be able to run all the pip commands as a normal user and the modules will be installed in `/usr/local/lib/python2.7/site-packages`. However, if you have used the distribution's version of python in the past, some old version of modules may already be installed in `/Library/Python/2.7/site-packages/` instead, at which point some user-run `pip` commands may fail. There is no pretty fix for this, you will need to run the `pip` command as root (via `sudo`). You can either uninstall the offending package as root and then install or upgrade the new one as your user to keep the `/usr/local` environment as clean as possible, or simply run `pip` as root instead bearing in mind that you may have permission issues later when using homebrew that will need to be fixed as root.

1.4 Fortran Compiler (`gfortran`)

The source of the `gfortran` compiler has changed since the instructions for Cupcake v0.2. The reasons for this change is that packages have been updated on the third party repositories, both macport and homebrew, and none of them work properly. The homebrew package we were using previously has been removed in favour of a full `GCC` install (overriding the system one although not replacing it). The version of `GCC` from homebrew is now 5.1.0 and is incompatible with the version of the `netcdf` package we will be getting from macport (the homebrew package for that still doesn't work). The macport package also installs the entire `GCC` suite but with different names for the binaries making it harder to use with standardise code.

Since our requirement is the `netcdf` package from macport, and this package is built with `gfortran` 4.9, we need to install this version of `gfortran`. The `GCC` maintainers do provide such a package for Mac, for both Mavericks (10.9) and Yosemite (10.10). Information can be found at `https://gcc.gnu.org/wiki/GFortranBinaries#MacOS`. For Yosemites, the packages is currently available from `http://coudert.name/software/gfortran-4.9.2-Yosemite.dmg`.

Once the packages is installed the `gfortran` binary should be available in your `$PATH`. You can check that `gfortran` is properly installed by issuing the command:

```
$ gfortran --version
GNU Fortran (GCC) 4.9.2
Copyright (C) 2014 Free Software Foundation, Inc.

GNU Fortran comes with NO WARRANTY, to the extent permitted by law.
You may redistribute copies of GNU Fortran
under the terms of the GNU General Public License.
For more information about these matters, see the file named COPYING
```

1.5 NetCDF Libraries

For NetCDF we will use the version available from MacPorts (we have tried using Homebrew for this in order to minimise the requirements and not mix the packaging environment but unfortunately, the NetCDF brew package could not be installed on our test machine. If this changes, we will reevaluate our instructions.

Although previous versions of cGENIE required the C, C++ and fortran versions of the library, this is no longer the case and the C++ layer is not a requirement anymore. The following commands will install the necessary NetCDF libraries:

```
$ sudo port install netcdf
$ sudo port install netcdf-fortran
```

2 Installing and running cGENIE (cupcake)

With all the steps described in this document, you should be able to follow the instructions described in the `cupcake-config-build.pdf` file in this folder to install and run the model.

If you have any issues with running cGENIE (cupcake) on Mac OS X, please report them at `https://github.com/genie-model/cgenie/issues`.

3 Quirks

Because so much of the system depends on third party repositories, it may not always behave as expected. We are aware that MacPorts for example provide packages for most of the tools we get from Hombrew, however, we have had reports of failures on systems with MacPorts

packages where installing the Homebrew package was a solution. This is especially valid for python.

We realised that making users change their tool chains is very much a annoyance. We are trying to test multiple install setup and try to understand why one tool chain fails where the other doesn't, but we currently lack the resources to do so quickly.

The information in this document is valid at the time of writing with the version of packages available at that time. Since we do not control the packages distributed on either Homebrew or MacPorts, we cannot tell whether updated versions of these packages will still work. We will endeavour to follow the releases of the packages with use and test any upgrade but it is not guaranteed happen in a timely manner.