Octave on Mac OS X

A survival guide

Per Persson

October 14, 2007

Contents

1 Introduction

This document assumes that you have a working knowledge of Mac OS X, the command line interpreter (CLI) i.e. "Terminal.app" and a general understanding of computers. It is not a "How-To" in the ordinary sense, since I'm not knowlegeble enough to author such a document, but hopefully it could be the start of a good "How-To". While I'm at it I might as well add the standard disclaimer: I'm not a UNIX programmer, just a Ph. D. student, and therefore I take no responsibility whatsoever for any damage (mental or physical, direct or indirect) that might be the consequnce of following any instructions in this document.

That said, let's rock!

1.1 Prerequisits

Check that the following criteria is met:

- Mac OS X v10.1 or later. Parts of this document specific to v10.1 through 10.1.5 will be marked **[10.1.x]**
  while parts specific to 10.2 (Jaguar) will be marked **[10.2.x]**
- Apples developer tools,
  `http://developer.apple.com/macosx/gettingstarted/`
  corresponding to your system version. They are free of charge but you are required to register.
- Fink, a package manager for Mac OS X,
  `http://fink.sourceforge.net/`
  Not strictly necessary but will save you a lot of time.
- dlcompat, a package providing dlopen family functionality on Mac OS X
  Install via fink: `fink install dlcompat`
  (For an overview of Mac OS X native dynamic loading, see [1])
- f2c or g77 FORTRAN compiler
  Install via fink: `fink install [g77 | f2c]`
- tetex or texinfo for generating documentation
  Install via fink: `fink install [tetex | texinfo]`

N.B. **[10.1.x]**If you are using 10.1.x you must use the gcc 2.95.2 compiler when following the instructions, verify by typing gcc -v at the prompt.

With Mac OS X being unix-based, one would expect that octave would compile out-of-the-box. It doesn't. instead, it has exposed a lot of problems with the modified gcc that Apple inherited from NeXT[2] and especially C++ templates have caused problems. This situation should improve as Apple merges more code with the FSF[3] gcc codebase.
The explicit purpose of this chapter is of course to provide information such that this chapter can be eliminated in future versions of this document!

2 Building octave on Mac OS X 10.1

The following instructions refer to octave version 2.1.35 *exclusively.*

2.1 Problems and fixes

2.1.1 config.{guess, sub}

Configure script doesn't recognize Mac OS X, a.k.a. Darwin-5.x.

Replace config.guess and config.sub with the ones in /usr/libexec/ by issuing the command:
`cp /usr/libexec/config.* .`
`cp /usr/libexec/config.* kpathsea/.`
N.B. This could probably be fixed upstream.

2.1.2 Dynamic loading

Mac OS X has different kind of dynamic linking compared to e.g. Linux, octave requires dlopen() etc.
Some clever people has written a wrapper, called dlcompat, around the .dylib dynamic loading. It is available under GPL from the Fink project.

2.1.3 Preprocessor

CPP complains about code that is correct.

Apple supplies a standard CPP as well as CPP-precomp[4] (which is default) but to compile octave (and other C++ code) the standard CPP must be used. Set the environment variable CPPFLAGS:
`setenv CPPFLAGS -no-cpp-precomp`

2.1.4 Character-escapes

In Makeconf, the line:
`UGLY_DEFS = ... -DSEPCHAR_STR=\\\":\\\" ...`
must be changed to
`UGLY_DEFS = ... -DSEPCHAR_STR=\\\\\":\\\\\" ...`
N.B. This could probably be fixed upstream.

2.1.5 Compiler flags

The following are the settings I've used for the compiler et al.:

CXXFLAGS `= -O2 -Wall`

XTRA_CXXFLAGS `= -fno-coalesce-templates -fno-implicit-templates`
this deals with the C++ template issues

LDFLAGS `= -Xlinker -m`
this forces the linker to link with the first of any multiply-defined symbols. Multiply-defined symbols are **not allowed** *in Mac OS X, this is a scary work-around...* [1]

SH_LDFLAGS `= -bundle -bundle_loader /usr/local/bin/octave`
a bundle is an aggregation of executables, headers, resources, images etc. that can be dynamically loaded[1]

RDYNAMIC_FLAG =
dynamic loading is default

2.1.6 libstdc++ bug

This is a really nasty bug in the libstdc++ that comes with Mac OS X which manifests itself when building with -fno-coalesce-templates *and* -O2. The workaround is to extract all object files from libstdc++ and the explicitly link with them instead of libstdc++.
By extracting the object files using
`ar -x /usr/lib/libstdc++.a`
and creating a variable `LIBSTDC++_OBJ` that contains a reference to all the files
```
LIBSTDC++_OBJ := PlotFile.o filebuf.o iofscanf.o iostream.o \
pfstream.o SFile.o filedoalloc.o iofsetpos.o \
```

```
iostrerror.o procbuf.o builtinbuf.o fileops.o \
iogetc.o ioungetc.o sbform.o cleanup.o floatconv.o \
iogetdelim.o iovfprintf.o sbgetline.o cmathi.o \
fstream.o iogetline.o iovfscanf.o sbscan.o cstdlibi.o \
indstream.o ioignore.o isgetline.o stdexcepti.o cstringi.o \
initio.o iomanip.o isgetsb.o stdiostream.o cstrio.o \
ioassign.o iopadn.o isscan.o stlinst.o cstrmain.o ioextend.o \
iopopen.o ldcomio.o stream.o dcomio.o iofclose.o ioprims.o \
ldcomplex.o streambuf.o dcomplex.o iofeof.o ioprintf.o \
strops.o editbuf.o ioferror.o ioputc.o outfloat.o \
strstream.o osform.o fcomio.o iofgetpos.o ioseekoff.o \
parsestream.o valarray.o fcomplex.o iofread.o ioseekpos.o \
peekc.o
```

which is added to the rules for making targets `octave` and `munge-texi` the linking will work.
N.B. This is where my unix weaknesses shows the most... It is obvious that this could be done using some kind of wildcard, e.g.:
`LIBSTDC++_OBJ :=$(wildcard tmpobj/*.o)`
but I can't get the dependencies right.
It is not likely that all of the above *.o files must be linked separately, but nobody has volonteered to pick a subset...

2.1.7 tempnam

Mac OS X has a broken tempnam() function and it is recommended that mkstemp() is used instead. Problem is that overriding `HAVE_TEMPNAM` with `-DHAVE_TEMPNAM=0` at compile time doesn't help. Seems like a new implementation of octave's tmpnam() using mkstemp() is needed.

2.2 Installation

Since fink has a working package that can be installed by simply typing:
`fink install octave`
or
`fink install octave-atlas` for octave linked with the Atlas math libraries,
I recommend that means of installation, but by applying the fixes in section 3.1 a simple configure & make should suffice.

2.2.1 DejaGNU results

Does it work? Well, according to DejaGNU we have 3 unexpected failures (residue-1.m, rename-1.m, getgrgid-1.m) which are listed below.

```
[per octave-2.1.35/test] % make check | grep FAIL
FAIL: octave.test/eval-catch/eval-catch-9.m
FAIL: octave.test/eval-catch/eval-catch-10.m
FAIL: octave.test/poly/residue-1.m
FAIL: octave.test/system/rename-1.m
FAIL: octave.test/system/getgrgid-1.m
FAIL: octave.test/try/try-9.m
FAIL: octave.test/try/try-10.m
```

2.2.2 mkoctfile

Since mkoctfile is used to compile C++ files to .oct files that can be dynamically linked to octave, mkoctfile must also work around the bug in libstdc++. One way to do this is to set
`objfiles=‘ls -1 /path/to/tmpobj/*.o‘`.

3 Building octave on Mac OS X 10.2

Currently, octave knows of Mac OS X 10.2 and a statically linked version of octave will buid from CVS HEAD by doing a ./configure + make. If you need to load .oct files see section 3.1.2.
The following instructions refer to octave version 2.1.40 *exclusively*.

3.1 Problems and fixes

3.1.1 Preprocessor

cpp complains about code that is correct.

Apple supplies a standard cpp as well as cpp-precomp[4] (which is default) but to compile octave (and other C++ code) the standard cpp must be used. Set the environment variable CPPFLAGS:
`setenv CPPFLAGS -no-cpp-precomp`

3.1.2 Dynamic loading

Mac OS X has different kind of dynamic linking compared to e.g. Linux, octave requires dlopen() etc.

As of version 2.1.40 octave has a native implementation of dynamic linking using the dyld API.
Dynamic loading of .oct files work, but is presently not fully supported.
Do *not* add `--enable-dl` to the configure options load .oct files. Instead run configure and then open `config.h` and change
`/* #undef HAVE_DYLD_API */` to `#define HAVE_DYLD_API 1`
to activate the native loader.

3.1.3 Known bugs

A bug in libstdc++ affects reading binary files.

3.2 Installation

Get the octave-2.1.40.tar.gz file from
http://www.octave.org/download.html and unpack it by issuing `gnutar xzf octave-2.1.40.tar.gz`

3.2.1 Building

Run configure. Do *not* add `--enable-dl` to the configure options load .oct files, see section 3.1.2.
Now is the time to open `config.h` and edit HAVE_DYLD_API according to section 3.1.2.
Then issue the command
`make` followed by
`make install`

3.2.2 DejaGNU results

Does it work? Well, according to DejaGNU we have 1 unexpected failure (binary-io-1.m) which are listed below.

```
[per octave-2.1.40/test] % make check | grep FAIL
FAIL: octave.test/eval-catch/eval-catch-9.m
FAIL: octave.test/eval-catch/eval-catch-10.m
FAIL: octave.test/io/binary-io-1.m
FAIL: octave.test/try/try-9.m
FAIL: octave.test/try/try-10.m
```

3.2.3 mkoctfile

4 Octave-forge

Octave-forge[5] is a set of extensions to octave that provides lots of useful functions and a great deal of MatLab^{TM}compatibility.

4.1 Considerations

Some of the files in octave-forge assumes that you have an X11 installation, something not all Mac OS X systems have (but could easily install[7, 6]). More to come...

4.2 Installation

Conflict with 'strip' and mkoctfile '-s' options.
This is resolved by using 'strip -r -u' instead of just 'strip'.
X11 installed.
Did not install:

```
./extra/mex/NOINSTALL
./extra/NaN/NOINSTALL
./extra/patches/NOINSTALL
./extra/perl/NOINSTALL
./extra/tk_octave/NOINSTALL
./extra/ver20/NOINSTALL
./extra/Windows/NOINSTALL
./main/audio/NOINSTALL
./main/sparse/NOINSTALL
./nonfree/gpc/NOINSTALL
./nonfree/splines/NOINSTALL
./configure --with-path=$HOME/Library/Octave/octave-forge
--prefix=$HOME/Library/Octave
```

N.B. see section 5 for a discussion on choice of `--path` and `--prefix`

```
make
```

`make check`, failed for:

```
assert(c \(c'\[1;1]),[0;1]);
assert(typeinfo(c),"tri");
```

5 Mac OS X customization

5.1 General guidelines

The preferred location in OS X for things like octave-forge is $HOME/Library/ rather than /usr/local/. If you dig around in /usr/local all day long, then by all means, install octave-forge in /usr/local. For those of you who still feel a bit uncomfortable with /usr/... I recommend to create a folder named Octave in your Library folder and then add the following path to the file .octaverc (create if necessary) in your home folder: `PATH=...`

5.2 Plotting - X11 & AquaTerm

Regardless of whether you are using X11 or not, you might want to install AquaTerm, and a version of gnuplot that supports it (as well as X11). See http://aquaterm.sf.net or fink.
If you use aquaterm, an augmented `figure.m` exists in octave-forge, directory extra/MacOSX/plot.

5.3 Images

By installing `convert` you can augment `image.m` to have Preview.app, or any other app of your choice, display bitmaps.

5.4 Editors

Plugin for BBEdit, carbon emacs, Cocoa emacs, GNU Emacs

6 Information sources

Links, links, links

References

[1] http://developer.apple.com/macosx/pdf/macosx_overview.pdf

[2] http://www.macdevcenter.com/pub/a/mac/2002/05/03/cocoa_history_one.html

[3] http://gcc.gnu.org

[4] http://developer.apple.com/techpubs/macosx/Darwin/PortingUNIX/PortingUNIXToOSX

[5] http://octave.sourceforge.net/

[6] http://fink.sourceforge.net/

[7] http://xonx/

[8] http://aquaterm. sourceforge.net/