ROMS GETTING STARTED PROCEDURE

Personal Notes freely taken from Hetland's movie, ROMS' site forum discussion and my daily work book

Introduction

In order to setup the ROMS model is necessary:

- ◇ to be registered on the web site http://www.myroms.org with username and password;
- ◇ to have installed a C precompiler (cpp);
- ◇ to have installed a FORTRAN compiler (FC);
- ◇ to have installed the NetCDF software libraries for FC and for your own Operating System (OS) (http://www.unidata.ucar.edu/software/netcdf/);
- ◇ to have installed the subversion(svn) platform on your computer.

svn platforms are often already installed on computer with UNIX/Linux OS and it is easy to verify that: open a shell on your computer and type (blue colour):

> svn (*enter and wait*)

If the prompt answer (green colour) is:

> Type 'svn help' for usage

the svn platform is already installed.
Otherwise for a Linux OS architetture in a debian-like system, follow the command sequence:

> apt-get update *(enter and wait)*

> apt-get install subversion *(enter and wait)*

in a redhat-like system

> yum -y install subversion *(enter and wait)*

in a gentoo system

> emerge subversion *(enter and wait)*

Pay attention to have enough memory space on you hard disk and answer to all questions done during this procedure accordingly with your computer and OS feature.

I tested this procedure on 3 different machine/computer architectures:

a) machine: Hp Intel IA64 - Itanium 2
OS: Linux 2.6.8-2-mckinley-smp (dannf@biglpk) - Debian 1:3.3.6-5
Cpp: gcc 3.3.6
FC: g95
NetCDF: release 3.6.1

b) machine: Intel(R) Core(TM)2 Duo CPU E6750 @2.66GHz - Intel i386
OS: Linux 2.6.22-14-generic (buildd@palmer) - Ubuntu 4.1.2-16ubuntu2
Cpp: gcc 4.1.3 20070929 (prerelease)
FC: ifort
NetCDF: release 3.6.1

c) machine: McIntosh
OS: Mc OS X – Leopard 10.5
Cpp: cpp
FC: g95
NetCDF: release 3.6.1

ROMS' source code DOWNLOAD

1)Create an empty folder on your computer, e.g. *yourpath*/ROMS-3.1.

2)Open a shell and on your root type

> svn checkout https://www.myroms.org/svn/src/trunk *yourpath*/ROMS-3.1 (*enter and wait*)

Allow the validation of site's authenticity certificate permanently (p) and insert your computer's password, ROMS' username and password, if and when requested. The ROMS' site don't request these information if computer's username/password and ROMS' username/password are the same. Now you are downloading the last revision of ROMS source code (e.g. revision number 211).
At the end of downloading you can read on the shell:

> Checked out revision 211

and you will find (https://www.myroms.org/wiki/index.php/Getting_Started) the following subfolders under *yourpath*/ROMS-3.1

```
/Atmosphere/                    Atmosphere models root directory
           /COAMPS              COAMPS root directory (empty)
           /WRF                 WRF root directory (empty)
/Compilers/                     make configuration files
/Data/                          Input data root directory
     /ROMS/                     ROMS data root directory
          /CDL                  ROMS Metadata design
          /Forcing              Input test cases forcing NetCDF files
          /Grid                 Input test cases grid NetCDF files
          /Initial              Input test cases initial conditions NetCDF
/Lib/                           External libraries
    /ARPACK                     Arpack eigenvalue problems library
    /MCT                        Modeling Coupling Tool library
    /MCT_WRF                    WRF Modeling Coupling Tool library
/Master                         Main standalone and coupling programs
/ROMS/                          ROMS root directory
     /Adjoint                   Adjoint model
     /Bin                       Executable scripts
     /Drivers                   Computational drivers
     /External                  Standard input scripts
     /Functionals               Analytical expression header files
     /Include                   Test cases configuration header files
     /Modules                   Declaration modules
     /Nonlinear                 Nonlinear model
     /Obsolete                  Discontinued files
     /Programs                  Support programs
     /SeaIce                    Sea-ice model (empty)
     /Representer               Representer model
     /Tangent                   Tangent linear model
     /Utility                   Generic utility files
     License_ROMS.text          Open source license
     Version                    SVN Version information
/User/                          ROMS User interface root directory
     /External                  User standard input scripts
     /Functionals               User analytical expresions templates
     /Include                   User application header files
/Waves/                         Waves models root directory
      /SWAN                     SWAN root directory
           /External            SWAN input data and standard input files
           /Src                 SWAN model
```

Note:

In order to keep up to date the ROMS source code at the last revision, everyday go into *yourpath*/ROMS-3.1 on your shell and type:

> svn up (*enter and wait*)

The answer will be

> At revision 211

if your revision is already up to date, or

> Updated to revision 212

if something has been modified in the code and the up to date files have been downloaded.

ROMS' FUNCTIONING TEST

To do a ROMS' FUNCTIONING TEST you may either follow the subsequent 3 to 8 steps (short method) or leap to the step 9 of the next paragraph and replace the name MyApp with UPWELLING (complete method).

3) Type

> cd *yourpath*/ROMS-3.1/Compilers

> ls

4) In the list of *.mk files choose the one, whose name correspond to your OS and FC (e.g. Linux-g95.mk), create a safety copy and edit it:

> cp Linux-g95.mk Linux-g95.mk_old

> vi Linux-g95.mk

If necessary, change the cpp's name and the flags of your cpp and FC according to your machine ad OS. Write also the correct and full path to the folders Include and Lib of the NetCDF application. Pay attention that you might indicate this path for NetCDF4 or NetCDF3; modify only one according to the release you are using. Save and close the file.

Note:

if in the *yourpath*/ROMS-3.1/Compilers directory doesn't exist a file like OS-FC.mk with OS and FC of your computer, modify and rename conveniently one of the files there. Example: in the case c) of the tested computer architectures I modified Darwin-f90.mk and renamed it Darwin-g95.mk.

5) Edit the makefile in *yourpath*/ROMS-3.1:

> cd *yourpath*/ROMS-3.1

> vi makefile

Indicate the appropriate FC at line:

FORT ?= (e.g. FORT ?=g95)

Specify if you are using debug (on/off), NetCDF4, openMP or MPI…
Save and close the file.

6)Type

> make –r (*enter and wait*)

Now the cpp and FC will produce the executable oceanS (oceanG if debug is on) of test application Upwelling and put it in *yourpath*/ROMS-3.1.

7) Type

> ./oceanS < *yourpath*/ROMS-3.1/ROMS/External/ocean_upwelling.in > upwelling.log &

(*enter and wait*)

or

> ./oceanG < *yourpath*/ROMS-3.1/ROMS/External/ocean_upwelling.in > upwelling.log &

(*enter and wait*)

You are running the application UPWELLING through the input file ocean_upwelling.in and you may see in the file upwelling.log the progress of the run, apart from the summary of the application characteristics. This test application don't need external forcing file; the bathymetry, the grid, the initial condition etc. are all analytically defined in the files *yourpath*/ROMS-3.1/ROMS/Functionals. At the end of the process, you have produced 4 Output netCDF files (ocean_*.nc) in *yourpath*/ROMS-3.1 that you can read with the NetCDF software or your post processing application.

8) The physical variables and parameters might also be plotted trough the Matlab software supplied with NetCDF interface for Matlab (http://www.unidata.ucar.edu/software/netcdf/). In the ROMS site or on John Wilkin's page http://marine.rutgers.edu/~wilkin/wip/, you may download some useful matlab scripts to do this.

If you have got this step, ROMS has been correctly installed on your computer.

CREATE YOUR OWN APPLICATION

9) Control if in the *yourpath*/ROMS-3.1/Compilers directory exists a file like as OS-FC.mk with OS and FC of your computer. Otherwise modify and rename conveniently one of those files there (see Note at point 3).
If you want, edit the makefile in *yourpath*/ROMS-3.1 to change the name of the executable: for example from oceanS to MyApp_oceanS.

10) Remove, if existing, the *yourpath*/ROMS-3.1/Build/ directory on *yourpath*/ROMS-3.1 and create there a new folder (e.g. MyApp):

> make clean

> cd *yourpath*/ROMS-3.1

> mkdir MyApp

11) Create the 5 new folders under *yourpath*/ROMS-3.1/MyApp: External, Functionals, Include, Input and Output.

12) Copy:

in *yourpath*/ROMS-3.1/MyApp the file *yourpath*/ROMS-3.1/ROMS/Bin/build.bash;

in *yourpath*/ROMS-3.1/MyApp/Include, the file *yourpath*/ROMS-3.1/ROMS/Include/cppdef.h;

in *yourpath*/ROMS-3.1/MyApp/External, the file *yourpath*/ROMS-3.1/ROMS/External/varinfo.dat;

in *yourpath*/ROMS-3.1/MyApp/External, the file *yourpath*/ROMS-3.1/User/External/ocean.in.

13) In *yourpath*/ROMS-3.1/MyApp/Include edit cppdefs.h: cancel the statement

#if defined ROMS_HEADER

#…

#else

#…

#endif

and create the feature of your own application following the instructions at the begin of the file and through the options named and listed there. Here you may choose the forcing to be considered, your open boundary condition (OBC), the turbulence closure, which physical quantities will be defined into an external input file or through an analytical function… Save, close and rename the cppdefs.h in myapp.h (it doesn't metter the case);

14) In *yourpath*/ROMS-3.1/MyApp edit build.bash:

Change clean from 1 to 0 and fill the right side of these assertions such that:

export ROMS_APPLICATION = MyApp

export My_ROOT_DIR =${PWD}

export My_ROMS_SRC =*yourpath*/ROMS-3.1/

export FORT =FC (e.g. g95)

Specify your CPP flags cancelling the comment (#) at the line

export MY_CPP_FLAGS='-D…'

and fill it with the appropriate options. Each flag must be preceded by –D and closed between ' or "" (It depends on the shell you are using).
Further specify if you are using Debug (on/off), NetCDF4, openMP or MPI… and the number of the nested grids!
Cancel the comment (#) at the line:

export USE_MY_LIBS =on

and write the correct path to the folders Include and Lib of the NetCDF application relative to your own FC.
Cancel the comment (#) to the 2 following lines and append the name of the directories 'Include' and 'Functionals' at the end of the path on the right side:

export MY_HEADER_DIR =MY_PROJECT_DIR/Include

export MY_ANALYTICAL_DIR =MY_PROJECT_DIR/Functionals

Save and close the file.

NOTE:

Pay attention to write immediately after the equal sing. Don't let any space!

15) Copy in *yourpath*/ROMS-3.1/MyApp/Input the external forcing input files like the grid file, the climatology file…in NetCDF format.

16) Copy from *yourpath*/ROMS-3.1/ROMS/Functionals to *yourpath*/ROMS-3.1/MyApp/Functonals all the ana_*.h files, which contain definitions that you would like to use in your application.
(For example to consider the tide forcing and to define it analytically, type in the shell under *yourpath*/ROMS-3.1/ROMS/Functionals:

> grep –nir tide .|more

And so you will find al the ana_*.h, in which you may define the tide. Equally, if you are trying to reproduce the UPWELLING test case, look for the ana_*.h files that contains the word "upwelling").

In *yourpath*/ROMS-3.1/MyApp/Functonals modify *ad hoc* the ana_*.h files:

If def MyApp
…
endif

or

if def HisApp
…
elif defined MyApp
…
endif

NOTE:

Pay attention to write the name of the application like in the build.bash file, because of the case sensitivity.

17) Edit the file *yourpath*/ROMS-3.1/MyApp/External/ocean.in. Here you will define the name of the application (e.g. MyApp – case sensitive), the number of the internal grid point in ☐ (Mm) and ☐ (Lm) directions and the number of vertical levels (N), the baroclinic and barotropic time step apart from the quantities that you don't supply through the external forcing input file and the analytical definition in *yourpath*/ROMS-3.1/MyApp/Functionals…
Specify the full path to get to varinfo.dat (or simply VARNAME = varinfo.dat) and to get to the input files. Indicate also where you will find the output files (*yourpath*/ROMS-3.1/MyApp/Output). Save and close the file. You may rename it MyApp.in for example, but it is not necessary.

18) In *yourpath*/ROMS-3.1/MyApp type:

> ./build.bash

The cpp and FC will produce the executable MyApp_oceanS.

19) Like in the point 6), type:

> ./MyApp_oceanS < ./External/MyApp.in > MyApp.log &

Now you are running your own application and at the end you will find your output files in *yourpath*/ROMS-3.1/MyApp/Output.

20) At the end you will have in *yourpath*/ROMS-3.1/MyApp the following directories and files:

```
/MyApp/                    My Application directory
    /External              MApp.in, varinfo.dat
    /Functionals           analytical expressions files (*.nc)
    /Include               header file: myapp.h
    /Input                 external forcing input files (*.nc)
    /Output                ocean_avg.nc, ocean_his.nc, ocean_rst.nc…
    MyApp_oceanS           executable
    MyApp.log              summary run's log file
```

21) Acknowledgments:

My thanks to A. Bergamasco, T. Minuzzo, S.Carniel, J. Chiggiato (CNR-ISMAR) and H. Arango for their suggestions and corrections.

Try it and have a good luck :o)
Cheers from

Arianna Trevisiol
PhD Student of Polar Science School - Department of Earth Science
University of Siena (Italy)

c/o

CNR-ISMAR
Castello 1364/a
30122 Venezia (Italy)

e-mail: trevisiol@unisi.it
tel. 0039 041 2404 764
fax. 0039 041 5204 126