University of Energy and Natural Resources, Sunyani

Name:

UBA, Felix

How to get ROMS running

@ Summer school

August, 2016

PRESENTATION

- Introduction
- How to download the code,
- Configure it for an Application,
- Run the Model.
- Error messages that arise during the configuration process

Download ROMS

To download the code to your own machine, these are the steps you would follow:

- You must have already registered on the ROMS portal and obtained your ROMS username/password as indicated in the Register
- Create a roms folder where you will keep the ROMS source code. i.e.

```
cd ~
mkdir roms
```

Note: ~ is /home/Uba

- Check out the ROMS source code replacing *ubafelix* with the ROMS user name you registered with

```
svn checkout --username ubafelix https://www.myroms.org/svn/src/trunk roms
```

Customize the Build Script

- The ROMS source code comes with a build script in the ROMS/Bin directory. Examples written with bash (build.bash) and csh (build.sh) are provided
- A full description of the build script can be found here.

Configure for an Application

- In your home directory (you can use some other directory to organize your ROMS projects if you wish) create a new folder named Projects and change into it. i.e.

```
cd ~
mkdir Projects
cd Projects
```

- Create a folder named upwelling and change into it. ROMS is distributed with several Test Cases and the Upwelling example is the default which we will compile and run here. i.e.

```
mkdir upwelling
cd upwelling
```

- Copy the build.bash file distributed with ROMS to your Projects/upwelling directory.

Configure for an Application

- Next we need to configure a few options inside build.bash so that it finds the directories where the source code and your Project are located
- Open the build.bash script you just copied into your upwelling directory using your preferred text editor, e.g. notepad ++
- Scroll down until you find ROMS_APPLICATION. You will notice it is set as follows:

```
export ROMS_APPLICATION=UPWELLING
```

- Scroll down until you find MY_PROJECT_DIR and MY_ROOT_DIR and set it as follows:

```
export MY_ROOT_DIR=/home/Uba/roms
export MY_PROJECT_DIR=${MY_ROOT_DIR}/Projects/upwelling
```

Configure for an Application

- Set MY_ROMS_SRC to the location of the source code:

```
export MY_ROMS_SRC=${MY_ROOT_DIR}/trunk
```

- Make sure that MY_CPP_FLAGS is **not** set.

```
#export MY_CPP_FLAGS="-DAVERAGES“
```

- The UG Computer Lab machines are single core, so we need to tell build.bash not to assume MPI parallel compilation. Comment out the options for USE_MPI and USE_MPIF90, i.e.

```
#export USE_MPI=on
#export USE_MPIF90=on
```

If you were compiling in parallel you would leave the default entries in build.bash. i.e.

```
export USE_MPI=
export USE_MPIF90=
```

Configure for an Application

- We leave the compiler option to

```
export FORT=gfortran
```

- In the interests of speed for this tutorial, we turn off compiler optimization by activating the debug option

```
export USE_DEBUG=on
```

- Uncomment the line: #export USE_MY_LIBS=on

```
export USE_MY_LIBS=on
```

- Find the gfortran section inside the **if [-n "${USE_MY_LIBS:+1}"]** block and change it to

```
export      NC_CONFIG=/usr/local/bin/nc-config
export   NETCDF_INCDIR=/usr/local/include
```

- Save and close notepad++

Configure for an Application

- Copy files ocean_upwelling.in, varinfo.dat and upwelling.h into the Projects/upwelling directory you just created.

```
cp ../../trunk/ROMS/Include/upwelling.h
cp ../../trunk/ROMS/External/ocean_upwelling.in
```

- We need to make one change to our ocean_upwelling.in file so open it with your favorite editor and find the following line:

```
VARNAME = ROMS/External/varinfo.dat
```

and change it to (again replacing ‘Uba' with your ROMS user name):

```
VARNAME = /home/Uba/roms/trunk/ROMS/External/varinfo.dat
```

Compile ROMS

- Go to your upwelling project directory:

```
cd ~/Projects/upwelling
```

- Then type:

```
./build.bash
```

- You may give the option -j to the build command to distribute the compilation to multiple processors if your host supports this, e.g.:

```
./build.bash -j 8
```

to compile on 8 processor at once

- If your build was successful it will not have reported any errors, and there will be an executable file in your Projects/upwelling directory called oceanS.

Run ROMS

- You run ROMS by executing the oceanG (or oceanS) binary, giving it the ocean_upwelling.in file as UNIX standard input

```
./oceanS < ocean_upwelling.in
```

- ROMS standard output will be typed to the screen. To save it a file instead, enter, e.g.:

```
./oceanS < ocean_upwelling.in > my_upwelling.log
```

- If you have compiled a parallel (MPI) executable, the syntax for running the mode is slightly **but critically** different

```
mpirun -np 8 ./oceanM ocean_upwelling.in > my_upwelling.log
```

where the "-np 8" indicates use 8 processors and this number of tiles must have been set by

Run ROMS

- If lots of numbers are displayed on the screen ROMS is running! Poor one last cup of coffee and enjoy the show (~15 min). Out of the box, ROMS comes programmed to run the Upwelling test case, which is what you are running now. When it finishes, the following output files are created:

  ocean_avg.nc
  ocean_dia.nc
  ocean_his.nc
  ocean_rst.nc

BINGO

YOU HAVE ROMS RUNNING

END

QUESTIONS