Assignment #1: Compiling and Running WRF-ARW on *comet*

Due: 19 September 2019

This assignment is designed to give you experience with compiling and running the WRF-ARW numerical model on *comet*, the national supercomputer which we will be using for selected course assignments and for the course project. It also serves as a crash course in Linux-based computing. The assignment that you turn in is primarily a curated set of screen captures. Consequently, you can e-mail me your assignment (before the start of class on 19 September) rather than print a copy and turn it in during class.

Prerequisites

1. (5 pts) Register for an XSEDE user account. To do so, go to https://portal.xsede.org and click on the "Create Account" button. Once you have completed registration, please e-mail me with your user name so that I can add you to our resource allocation. *You will not be able to complete any model-related tasks later in this assignment until this portion of the assignment is completed, and so I encourage you to complete this task as soon as possible!*

   After creating your account, on the main XSEDE portal website, click on the "Profile" link in the blue upper menu bar. In the upper right-hand side of this page, click the link to enroll in Duo two-factor authentication. Accept the terms and enroll. *This step is also necessary to be able to complete later tasks in this assignment!*

   For later reference, the XSEDE "Getting Started" guide is available at:

   https://portal.xsede.org/documentation-overview

   Likewise, the *comet* User's Guide is available at:

   https://portal.xsede.org/sdsc-comet

   comet is a supercomputer located at the San Diego Supercomputing Center. All access to *comet* is remote – i.e., you use your computer to securely yet remotely log in to *comet*.

2. (10 pts) Successfully log on to *comet*. To connect to the XSEDE Single Sign-On Hub and then to *comet*, you need a Secure Shell (SSH) client, preferably one that can display remote windows.

   If you are using a Mac, you already have an SSH client available to you via the Terminal. The Macs in the Atmo Lab (W434) and your offices already have the needed software to

display remote windows; however, if you are using a personal Mac, you may need to install the Xquartz package to add this capability to your machine.

If you are using a Windows machine, you must install an SSH client. I recommend using MobaXTerm Home Edition (https://mobaxterm.mobatek.net/download.html), including an SSH client and the ability to display remote windows.

To connect to *comet*, you first log on to the XSEDE Single Sign-On Hub. If you are using Terminal on a Mac, issue the following command in the Terminal window:

ssh -Y <username>@login.xsede.org

where <username> is replaced by your username, without brackets. Note that the Y is case-sensitive. Enter your password when prompted, then select either a Duo push or phone call to complete the login process.

If you are using MobaXTerm, click on the "Session" button at upper-left, then on SSH. Enter login.xsede.org in the "Remote Host" entry, check the "Specify username" box, and enter your username in the box to the right. Connect, enter your password when prompted, then select either a Duo push or phone call to complete the login process. If it prompts you for your password a second time, simply cancel out of that window.

Once logged in to the Single Sign-On Hub, you connect to *comet* using the same command no matter whether you are on a Mac or Windows machine:

gsissh -Y comet

This command is entered into the terminal window at the Single Sign-On Hub prompt (e.g., [<username>@ssohub ~]$). Note that the Y is case-sensitive.

Once you have logged on to *comet*, run the date command by typing date and hitting enter. Take a screenshot of your screen at this point and attach it with your completed assignment.

3. (10 pts) Establish your computing environment for compiling WRF-ARW. All software on *comet*, and many other supercomputers like it, is made available using modules. This way, users can specify only the software they need, and system administrators can simplify the basic computing environment that all users start with.

   To compile and run WRF-ARW, we need to load several modules: the compiler, software that allows us to run the model on many separate processing nodes at once, and software

that WRF-ARW uses to read input data and write model output. Some of these are available system-wide and were configured by XSEDE supercomputer facilitators, whereas I have compiled others for our use only.

To make these modules available to you every time you log in, we can place their loading commands into the .bashrc file in your home directory on *comet*. The .bashrc file contains a list of commands for the supercomputer to load every time you log in. To do so, use the nano text editor:

nano ~/.bashrc

The leading ~/ in front of .bashrc is a shortcut to your home directory. The default .bashrc file contains a few lines that should not be edited, followed by a comment "# User specifc aliases and functions." Use your keyboard's arrow keys to go below this line (note: get used to using the keyboard – your mouse and associated point-and-click habits will not work in this environment!) and enter the following lines:

```
module purge
module load gnutools
module load intel/2016.3.210
module load mvapich2_ib
module load hdf5
export HDF5="/opt/hdf5/intel/mvapich2_ib"
module use /oasis/projects/nsf/wim108/evans36/modules/
module load netcdf-local
module load grib2
```

Once you have entered these lines, save the file using Ctrl-O on your keyboard, then exit nano using Ctrl-X on your keyboard. Log off of *comet* back to the XSEDE Single Sign-On Hub, then reconnect to *comet* as you did in Question 2 above – that way, the module loads will take place immediately without further action on your part.

To test that you completed these commands correctly, after logging back on to *comet*, type ncdump and hit enter. Take a screenshot of your screen at this point and attach it with your completed assignment.

4. (10 pts) Establish links to your scratch and working directories. *comet* is organized around multiple disk storage spaces:

    - Your home directory (/home/<username>). This is a small storage space that is not

intended for completing computing tasks.

- A scratch directory (/oasis/scratch/comet/<username>/temp_project). Your scratch directory has the ability to store large amounts of data (*O*(TB)), but it is a temporary storage space that is shared among all supercomputer users. You should run all of your model simulations in this directory but you should move their output after they complete to your working directory.
- A working directory (/oasis/projects/nsf/wim108/<username>). This is where you should compile all of your code and complete all of your assignments. While ample storage is available in this directory, it is shared among your classmates, with ~150 GB available to each person.

Note that your username on comet may, in rare instances, be different than your username used to log on to the Single Sign-On Hub! When logged on to comet, you can check what it is by looking at the text on the command line between the [and @. You should use this username, and not your XSEDE user name, in all instances where <username> appears in the list above and in the questions that follow.

As you might surmise from the last bullet above, your scratch and working directories do not exactly have easy-to-remember names. However, you can make them available from your home directory, with easier-to-remember names, using symbolic links.

Start by changing into your home directory by typing cd and hitting enter. cd is the Linux command for *c*hange *d*irectory; when followed by a directory name, it will change into that directory. Next, create a symbolic link using ln -s:

ln -s full name of directory to link to name of link

Repeat for your other folder. Once this is done, confirm that each were created by listing the contents of your home directory:

ls -al

Take a screenshot of your screen at this point and attach it with your completed assignment. Any time you log on to *comet*, you should make sure you first change directory into either the scratch or working directory, whichever is more applicable for your job, noting that you often will be switching between the two (particularly when running WRF-ARW).

Compiling and Running WRF

This assignment largely follows the WRF-ARW Online Tutorial to guide you in downloading,

compiling, configuring, and running the WRF-ARW model. The Online Tutorial is available at:

http://www2.mmm.ucar.edu/wrf/OnLineTutorial/

Start by going to the Introduction menu at upper left, then choosing Getting Started.

5. (10 pts) Download WRF-ARW v4.1.2 and WPS v4.1, following the directions beginning under "Get Source Code" on the "Let's Get Started" page to do so. Note that when running git clone, *you must be in the working directory*; you only need to be in the scratch directory when running the model itself. Once the model code has been downloaded, run ls -al to list the contents of the working directory. Take a screenshot of your screen and attach it with your completed assignment.

6. (15 pts) Configure and compile the WRF-ARW model code. Move through the WRF Code section of the Online Tutorial to the "Configure WRF" page. When presented with the long list of compile options, choose option 15. When asked about nesting, choose option 1. Once done, move ahead to the "Compiling for Real Data Cases" page and follow the directions to compile the model code, which will take 30-60 minutes to complete. Once done, run ls -al in your test/em_real/ directory to list its contents. Take a screenshot of your screen and attach it with your completed assignment.

7. (10 pts) Configure and compile the WPS code. Whereas WRF-ARW is the model itself, WPS is a set of preprocessing programs that establish the model domain, prepare data to initialize the model, and interpolate the initialization data to the domain. Page through the WPS Code section of the Online Tutorial to the "Configure WPS" page. (Note that you do *not* need to set the NETCDF environment variable as the WPS Code page implies – this is already done by loading the netcdf-local module upon login.)

   When presented with a list of compile options, choose option 19. After this is done, move to the "Compile WPS" page and follow the instructions to compile WPS. (You do not need to compile the utility programs that this page provides as an option.) Once the WPS code has been successfully compiled, run ls -al in the WPS directory to list its contents. Take a screenshot of your screen and attach it with your completed assignment.

At this point, you should read through but not complete any of the tasks listed in the *Run Basics*, *Basics – GEOGRID*, *Basics – UNGRIB*, *Basics – METGRID*, and *Basics – WRF* sections of the Online Tutorial. This will give you an overview of the WRF-ARW and WPS workflows, whereas completing the tutorial simulation for the remainder of this assignment will give you experience with running the WPS and WRF-ARW programs.

8. (5 pts) Create the domain for the January 2000 tutorial case. This can be accessed directly from the *Basics – WRF* page from above or from the Case Studies – Default Case – Case menu at the top of any Online Tutorial page. Follow the direcitons on the "Set up the model domain" page of the tutorial to create the domain, noting that the needed terrestrial data are already available on *comet* at /oasis/projects/nsf/wim108/evans36/wrfgeog/; this is the path you will enter for geog_data_path in namelist.wps.

   To complete the ncl command listed, you must first load the ncl module:

   module load ncl_ncarg

   Note also that you should use the plotgrids_new.ncl, rather than plotgrids.ncl, script when running ncl. Use this to obtain a graphical display of the model domain, take a screenshot of the resulting window, and attach it with your completed assignment.

   Our WPS compilation does not allow us to directly run geogrid.exe. Instead, geogrid.exe is run by submitting the program to a scheduler, which then runs the code when the needed resources are available. A sample submission script for geogrid.exe can be copied to your WPS directory:

   cp /oasis/projects/nsf/wim108/evans36/samplescripts/geogrid.sbatch .

   This submission script is nearly identical to that in the *comet* documentation for 1-processor jobs; in other words, we aren't doing anything special in this script. You should not need to make any changes to this script to use it for yourself, though you should first review its structure.

   When you are ready to run geogrid, you submit the batch script to the scheduler as follows:

   sbatch geogrid.sbatch

   To check on its status, use the squeue command:

   squeue -u <username>

   where you substitute your username for <username>. squeue provides information about all running and submitted jobs, while -u <username> filters the output from squeue to only include lines that have your username in them.

   Depending on how much of the supercomputer is being used at a given time, it may take a

little while before the code runs. If your code is running, the last three columns will indicate how long it has been running, how many computer nodes it is running on, and the computer nodes on which it is running. If squeue comes back empty, then your submitted job is done, though you still need to check whether it completed successfully. To do so, open the new file geogrid.#.out (where # is the job ID for your submitted job) and inspect its contents; a success message will be found at the bottom of the file if geogrid completed successfully.

Upon successfully running geogrid.exe, run ls -al in your WPS directory, take a screenshot, and attach it with your completed assignment.

9. (10 pts) Complete the "Let's get data for our domain!" (ungrib) and "Now interpolate these data onto our domains" (metgrid) tutorial sections.

   Note that when prompted to download the January 2000 case study data, right-click on the link, then choose "Copy link address" or similar. Next, in your DATA directory on *comet*, type wget, add a space, paste the copied link, and then hit return. This will download the data directly to *comet* rather than your personal computer.

   Part four of the Ungrib section of the tutorial asks you to link the GFS Vtable to your WPS directory. However, the GFS Vtable provided with WPS is set up to only work with output from very recent GFS model runs. The tutorial, which uses data from 2000, does not count as very recent. If you open the ungrib/Variable_Tables/Vtable.GFS file in a text editor and scroll down to line 72, starting with "As of WPS V4.1...," you'll see a note about the sea-level pressure field in GFS output. While you can revise Vtable.GFS so that the PMSL line (line 14) matches that in the comments, I have done this for you. In your WPS directory, issue the following command:

   cp /oasis/projects/nsf/wim108/evans36/samplescripts/Vtable.GFS ungrib/Variable_Tables/

   Once you have done this, you can follow the tutorial instructions for linking the GFS Vtable to your WPS directory without any changes.

   ungrib.exe can be run directly, as in the tutorial, but metgrid.exe cannot for the same reason as geogrid.exe. Copy the sample submission script for metgrid.exe to your WPS directory:

   cp /oasis/projects/nsf/wim108/evans36/samplescripts/metgrid.sbatch .

   No changes to this script should be necessary. You can submit it and check on its status in the same fashion as you did with geogrid.sbatch in the previous question.

   After metgrid successfully completes, use ncview to take a look at the met_em.d01.2000-

01-24_12:00:00.nc file. ncview is a simple utility that allows you to quickly, yet crudely, view the contents of georeferenced netCDF files such as those produced by WPS and WRF.

To make ncview available, you must first load the ncview module:

```
module load ncview
```

You can tell ncview to load a particular file by appending the filename to the end of ncview on the terminal line. Once the ncview window opens, display the HGT_M 2D variable. After the display appears, take a screenshot and attach it with your completed assignment. You can quit ncview after this is done.

10. (15 pts) Move to the "Run WRF" tutorial section. First, change into your scratch directory. Create a new directory here called WRF_run:

```
mkdir WRF_run
```

Next, change into your WRF/test/em_real/ directory. Copy the contents of this directory to your scratch directory with cp:

```
cp * ~/scratch/WRF_run/
```

The above command copies the contents of the em_real directory to the scratch/WRF_run/ directory accessible from your home (the shorthand for which is ~/) directory. If you named your scratch directory something other than scratch in #4, make sure that you substitute that name for scratch in your cp command. After you have successfully copied the contents of your em_real directory, change into the WRF_run directory – this satisfies the first step of this part of the tutorial.

Next, complete step two of this part of the tutorial – linking the met_em.* files. These files are in your WPS directory, which likely is not in the location given by ../../../WPS/. Instead, you will need to replace this the full path to your WPS directory, i.e.,

```
ln -sf /oasis/projects/nsf/wim108/<username>/WPS/met_em.d01.2000-01* .
```

Note the space after the * and before the period.

After this is completed, proceed to step three, editing namelist.input. Note that the comment on the tutorial that you won't need to make any changes is not correct – you will need to change num_metgrid_soil_levels from 4 to 2 and num_metgrid_levels from 32 to 27. Make

these changes, then save the file and exit the text editor.

Unlike indicated in steps four and five of this part of the tutorial, real.exe and wrf.exe must be run in the same way as we did for geogrid.exe and metgrid.exe. Sample submission scripts for these programs can be copied to your WRF_run directory:

```
cp /oasis/projects/nsf/wim108/evans36/samplescripts/real.sbatch .
cp /oasis/projects/nsf/wim108/evans36/samplescripts/wrf.sbatch .
```

The submission scripts will not need to be edited to run the Tutorial case, though they will need to be edited in later assignments and for your project simulations. More information on the job scheduler is available through the *comet* documentation linked in question #1.

When you are ready to run real.exe, submit the real.sbatch script to the job scheduler in the same way as you did for geogrid.sbatch and metgrid.sbatch. You can check on its status using the squeue command as before. If squeue comes back empty, follow the "verify that the program runs successfully" tutorial instructions. (If you didn't change the namelist as noted earlier, you'll find your error messages in rsl.error.0000. This is a good place to look for most WRF errors you might encounter. Some of them will be easy to diagnose, such as the two errors you'd get in this specific instance, whereas others will not be.)

When you are ready to run wrf.exe, submit the wrf.sbatch script to the job scheduler. After wrf.exe completes, use ncview to open the wrfout_d01_2000-01-24_12:00:00 file. Have it load the 4D variable T. By default, it will load T at the lowest model level at the first output time. Change to the last output time by clicking in the box under "Current" for the Time entry in the bottom of the main ncview window until the number in that box matches that in the Max column immediately to its right. Once on the correct time, take a screenshot and attach it with your completed assignment.