Downloading, compiling, and testing CM1 on the Scholar Cluster

**Download the Code**

Go to <https://www2.mmm.ucar.edu/people/bryan/cm1/> then click on “Download the code here.” Agree to the terms and then you will be brought to a page with different CM1 releases. We will use the latest release (cm1r21.0).

Right-click on the link for cm1r21.0.tar.gz, then click on “copy link address.”

Log in to Scholar.

Make a directory for this class, I just called mine EAPS591.

Type the command: **wget (then paste the link address we copied earlier)**

This will download the tar.gz file.

Type the command: **tar -xvf cm1r21.0.tar.gz**

This will untar the file, giving you a new directory called cm1r21.0

Familiarize yourself with the file structure in the cm1r21.0 directory.

There are some README files (it may be worth reading the README and README.compile)

There is a src directory. This is where we will compile the model.

There is a run directory. This is where we will run the model.

------------------------------------------------------------------------------------------------------------------------------------------

**Compiling CM1**

First, let’s load the netcdf libraries we will need. Type the command: **module load netcdf-fortran**

We will need to copy the Include path and the Library path. We can find these by typing the command: **nc-config --all**

First let’s copy the include path listed next to “--fflags”, it should start with -I

This is the Include path.

Go to the cm1r21.0/src directory and view the Makefile.

Uncomment the code needed for netcdf output capability (lines 11-14).

Paste the include path we copied earlier next to “OUTPUTINC”

Back on the command line, type the **nc-config --all** command again.

Now let’s copy the library path listed next to “--flibs”, it should start with -L

Only copy the first path listed (i.e., the first path ending with “/lib”). This is the Library path.

Go back to the Makefile and paste the library path next to “OUTPUTLIB”.

Next we need to tell the Makefile how to build CM1 based on Scholar’s hardware. We want multiple processors, shared memory (OpenMP), Intel compiler. Find this under the “HARDWARE SECTION” of the Makefile. Uncomment the code for this section (it should be lines 38-41).

Now the Makefile should be ready to go!

To compile the model and have it run properly with the batch submit system, we will have to build (a.k.a compile, a.k.a. make) the model within an interactive session on a scholar node, rather than the login node.

To start an interactive session, type the command: **sinteractive -A scholar -N1**

This will give you 1 node on scholar for 30 minutes.

Then load the netcdf libraries we will need again. Type the command: **module load netcdf-fortran**

Ok, let’s build CM1! Go to the cm1r21.0/src directory and type the command: **make**

This will compile the model based on the configuration we specified in the Makefile.

If there are errors, please raise your hand. If the compile fails, it’s usually best to type the command: **make clean** and then debug the issue before giving the **make** command again.

The compilation will take about 10-15 minutes, maybe longer. If successful, there will be no error messages and you will find two new files in the cm1r21.0/run directory:

cm1.exe : This is the executable to run the model.

onefile.F : This is a file that documents how the model was built. If you plan on building CM1 many different ways, you may want to save a copy of this for each new build.

Once CM1 is built, you can exit the interactive session by typing **exit**

------------------------------------------------------------------------------------------------------------------------------------------

**Setting up a test run of CM1**

Go to the cm1r21.0/run directory. View the namelist.input file. This file tells CM1 how we want to run it. There are many different variables that can be changed within this file.

Let’s set up a very simple run of CM1. We are going to only let the model advance one timestep before stopping it and checking the output. The default namelist.input is set up to initialize CM1 with a warm bubble in the center of its domain. We will see the vertical motion associated with this warm bubble after just one timestep.

View the namelist.input file.

Find and set ppnode = 4

Find and set timax = 7.500

Find and set run\_time = 7.500

Find and set tap\_freq = 7.500

Find and set output\_format = 2

Next we need to create a batch script that we will submit to the job queue on scholar. Open a new file called “batchscriptTEST” and paste the following:

#!/bin/bash

# FILENAME: cm1TEST

#SBATCH -A scholar

#SBATCH --nodes=1

#SBATCH --tasks=4

#SBATCH --time=0:30:00

#SBATCH --job-name cm1SimpleTest

# Print the hostname of the compute node on which this job is running.

/bin/hostname

# Load some modules

module load netcdf-fortran

# Runs CM1

date

module list

mpirun -np 4 ./cm1.exe > cm1.print.out

date

exit

We are now ready to submit our batchscriptTEST to the job queue and run our test of CM1!

Type the command: **sbatch batchscriptTEST**

This will submit our job of running CM1 to the Scholar queue.

Remember we set up our namelist to just advance one timestep, so the simulation only takes about 10 seconds to run.

When the simulation is complete, you should find some new files in the cm1r21.0/run directory:

cm1out.nc : This contains the model output in netcdf format.

slurm-xxxxxx.out : the x’s are the job number. This file contains messages from the batch system. If the simulation ran successfully, there should be no error messages in this file.

Let’s take a look at our output! We will use ncview to quickly visualize the output.

Load ncview by typing the command: **module load ncview**

Then type the command: **ncview cm1out.nc**

This will open ncview in a new window. Click on 4d vars, drag down to “w” then unclick. This will load the vertical velocity (w) field. Click once on the current zf index (this is the model height index). Click once on the current time index. This should now display the vertical velocity associated with the warm bubble in the model. Feel free to play around with different heights or different variables (“shs” under 3d vars shows updraft helicity).

If you see values produced by the warm bubble, CM1 has run successfully! Later in the course, we will set up more complex simulations.