

# CESM1 Tutorial: Basic Modifications

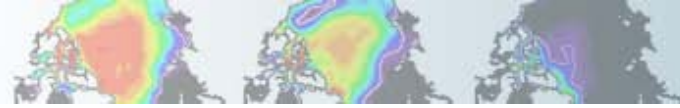
Christine A. Shields

August 2, 2011



U.S. DEPARTMENT OF  
**ENERGY**

Office of  
Science

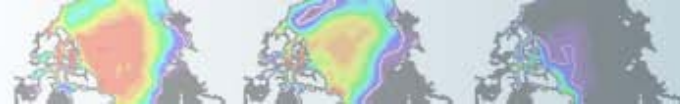


## CESM1 Tutorial: Basic Modifications: **Review**

1. We will use the CESM code located locally on bluefire, no need to checkout or download any input data.
2. We will run with resolution T31\_gx3v7.
3. Scripts will **automatically** be configured for you using the code/script base prepared uniquely for this tutorial.

*Tutorial Code and script base:*

***/glade/proj3/cseg/collections/cesm1\_0\_3\_tutorial***

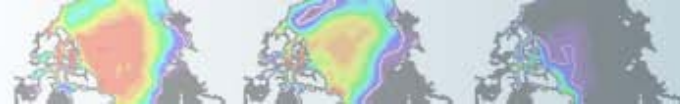


## CESM1 Tutorial: Basic Modifications: **Review:** **Creating a new case**

**EXERCISE.0:** Create and configure an out-of-the-box set of scripts called “b.day2.0” on bluefire using T31\_gx3v7 and compset B\_1850\_CN. Build the model.

### Explanation of steps

1. Change directories, (“cd”) to tutorial code base scripts directory.
2. View compset choices.
3. Create initial scripts.
4. “cd” to your casedir.
5. Congfiure scripts.
6. Build the model.
7. “cd” to your working directory and explore

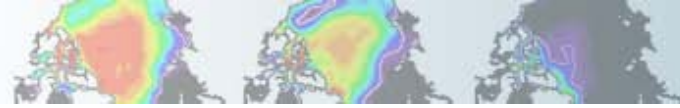


## CESM1 Tutorial: Basic Modifications: **Review:** **Creating a new case**

**EXERCISE.0:** Create and configure an out-of-the-box set of scripts called “b.day2.0” on bluefire using T31\_gx3v7 and compset B\_1850\_CN. Build the model.

1. `cd /glade/proj3/cseg/collections/cesm1_0_3_tutorial/scripts`
2. `create_newcase -list`
3. `create_newcase -case ~/b.day2.0 -res T31_gx3v7 -mach bluefire  
-compset B_1850_CN`
4. `cd ~/b.day2.0`
5. `configure -case`
6. `b.day2.0.bluefire.build`
7. `cd /ptmp/$LOGNAME/b.day2.0`

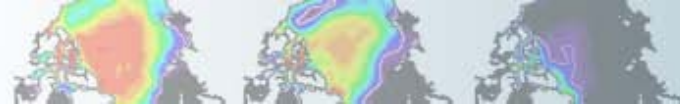
We will **not** run this case, but if you want to review short term archive space, “cd” to a run completed from yesterday, i.e. /ptmp/\$LOGNAME/archive/<mycase-from-yesterday>. Remember that the short term archive space is created **after** the initial run completion.



## CESM1 Tutorial: Basic Modifications: **Review: Queues and Jobs**

### BLUEFIRE

1. Submitting jobs:  
Type `$CASE.<mach>.submit` i.e. `b.day2.0.bluefire.submit`
2. Checking jobs:
  - a. Type `bjobs` or
  - b. Type `bjobs -u all` to see everyone's jobs, or
  - c. Type `batchview` to see everyone's jobs in a nice format
3. Killing jobs:
  - a. Find your JOBID after typing `bjobs`
  - b. Type `bkill <JOBID>`

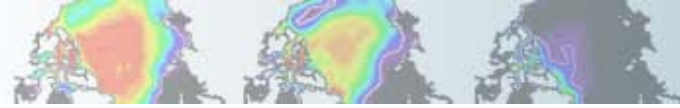


## CESM1 Tutorial: Basic Modifications: **Review:** **README**

In your case directory, in addition to your scripts, you will find a README directory and a README.case file.

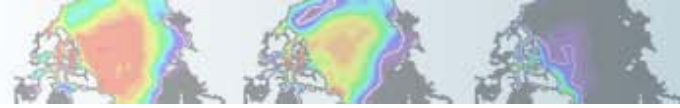
1. **README directory:** automatically created documentation files, please review.
2. **README.case file:** one line of text which echos your create\_newcase command.

**In *README.case*, we highly recommend YOU document any changes you make to the default scripts. It is YOUR paper trail and opportunity to list modifications.**



## CESM1 Tutorial: Basic Modifications: **create\_clone**

1. The **create\_clone** utility creates an exact copy of a previously created case.
2. The `create_clone` utility is very handy when the user wishes to run a slightly modified version of a previous experiment.
  - a. Invoke `create_clone` to create an exact copy of an old case by typing the following on the command line:  
***create\_clone -case <new case> -clone <case to clone>***
  - b. Implement desired modifications before building and running `.` (We will learn numerous way to modify the scripts during this presentation).
  - c. DOCUMENT changes in README.case.



## CESM1 Tutorial: Basic Modifications: **create\_clone**

**EXERCISE.1:** Clone b.day2.0 from Exercise.0, build, and run. Make no changes yet, we are simply practicing cloning a case.

1. “cd” to tutorial code base scripts directory
2. `create_clone -case ~/b.day2.1 -clone ~/b.day2.0`
3. `cd ~/b.day2.1`
4. `configure -case`
5. Informative messages from “configure –case”:

*Namelist configuration for env\_case.xml and env\_conf.xml **has already been done...skipping***

*Generating clean\_build script*

*Generating submit script*

*Generating build script*

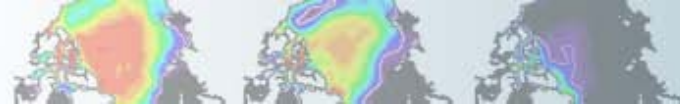
*Generating run script*

*Locking file env\_mach\_pes.xml*

*Successfully configured the case for bluefire*

*If an old build exists for this case, you might want to run the \*.clean\_build script before building*





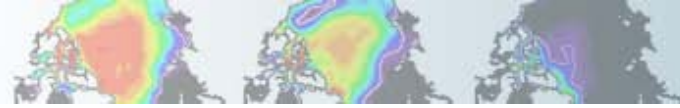
## CESM1 Tutorial: Basic Modifications: **create\_clone**

### EXERCISE.1 continued:

6. Run the build script, i.e. *b.day2.1.bluefire.build*
7. Edit *b.day2.1.bluefire.run* and add the following to *#BSUB* lines:
  - a) *#BSUB -P 37591059*
  - b) *#BSUB -U 37591059#1*

**Note this is for tutorial purposes ONLY.**

8. Submit the run, i.e. *b.day2.1.bluefire.submit*
9. Check the queue, *bjobs*
10. Go to */ptmp/\$LOGNAME/b.day2.1* and explore.
11. After the job completes, go to the short term archive space and explore.
12. After the job completes, go back to your case directory and explore the “logs” directory and the “timing” directory. How long did your job take to complete? What was the “throughput”, i.e. estimated model years per actual day? Notice that your Buildconf directory exactly matches the case you “cloned”. Explore the *\*buildnml.csh* component model namelist scripts.



## CESM1 Tutorial: Basic Modifications: **Methods and Tools**

When modifying files, the user is free to use her/his editor of choice, i.e.

**vi**

**emacs**

When modifying “xml” files, the user may also use the tool, **xmlchange**.

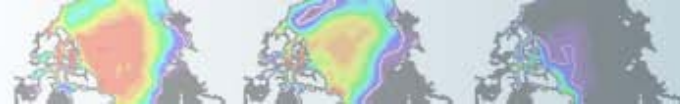
1. For help, type ***xmlchange -help***

2. Example:

You want to manually resubmitting an initial case that previously had a RESUBMIT value of 0, (i.e. you did not initially resubmit the run).

Edit `env_run.xml` via the `xmlchange` tool, type

```
xmlchange -file env_run.xml -id CONTINUE_RUN -val TRUE
```



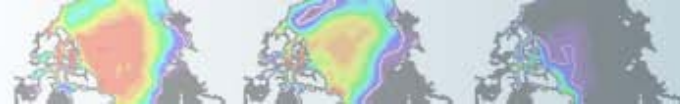
## CESM1 Tutorial: Basic Modifications: **env\_run.xml**

These variables MAY BE CHANGED ANYTIME during a run.

Common variables to change include

1. **RESUBMIT** → sets the number of times to resubmit the run
2. **STOP\_OPTION** → sets the run length time interval type, i.e. nmonths, ndays, nyears or a specific date
3. **STOP\_N** → sets the number of intervals (set by STOP\_OPTION) to run the model during the specified wallclock time. Wallclock time is set in your \*.run file and is a measure of the actual time.

**STOP\_OPTION** and **STOP\_N** control the length of the run per computer job submission. A typical simulation is comprised of many job submissions. (You can only stay in the computer queue for a specified time. This queue time limit is often shorter than the desired simulation length.)



## CESM1 Tutorial: Basic Modifications: **Example: env\_run.xml**

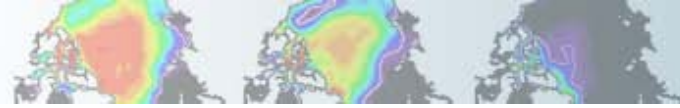
1. **RESUBMIT** → sets the number of times to resubmit the run
2. **STOP\_OPTION** → nmonths, ndays, nyears or a specific date
3. **STOP\_N** → sets the number of intervals (set by STOP\_OPTION) to run

### Question:

The special tutorial version of T31\_gx3 CESM on bluefire (one node only) simulates ~21 model years per wallclock day.

Maximum wallclock requests is 6 hours.

If you want to run 50 years, what values should be set for STOP\_OPTION, STOP\_N, and RESUBMIT?



## CESM1 Tutorial: Basic Modifications: **Example: env\_run.xml**

Question:

If you want to run 50 years, what values should be set for STOP\_OPTION, STOP\_N, and RESUBMIT?

**Answer:**

**Assume 4 jobs submissions per day, (4 6-hr jobs).**

**Model runs 21yrs/day, so  $21/4 = 5.25$  model years per job submission.**

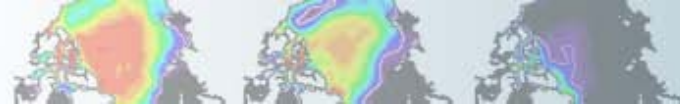
**Round down to 5 model years/submission to be safe (machines hang).**

**STOP\_OPTION = nyears, STOP\_N = 5, RESUBMIT = 9**

**Initial run of 5yrs + (9 resubmits \* 5 years per job) = 50 years**

**\*Note:**

**The released version of CESM1 T31\_gx3 can be optimized to run ~70 model year/wall-clock day.**



## CESM1 Tutorial: Basic Modifications: **env\_run.xml**

env\_run.xml continued... more common variables to change include

4. **CONTINUE\_RUN** → if TRUE, implies a CONTINUE run.

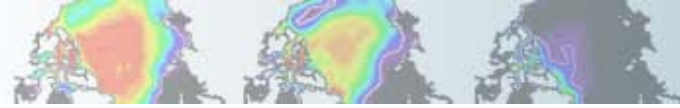
Note: if RESUBMIT is > 0 and it is an initial run (i.e. CONTINUE\_RUN=FALSE), CONTINUE\_RUN will automatically update to TRUE upon completion of initial run.

5. **INFO\_DEBUG** → sets level of stdout (standard out) print statements. If debugging, a higher value may be set.

6. **DOUT\_S** → turns on short-term archiving (short term archiving) . DOUT\_S is TRUE by default.

7. **AVGHIST\_OPTION** → coupler history file specification. Note: All other model components specify history file information within the model component namelists! (More on namelists in a few slides).

Take some time to review all other env\_run.xml settings....



## CESM1 Tutorial: Basic Modifications: **env\_conf.xml**

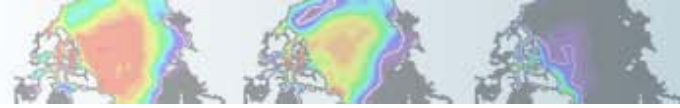
The env\_conf.xml file configures the model namelist scripts.

These variables CANNOT be modified once configure -case has been invoked without first invoking configure -cleannamelist.

For a complete list of variables that can be specified via the env\_conf.xml file, please review your env\_conf.xml file as well as on-line documentation for each component model. (example, CAM4 namelist variables are defined at: <http://www.cesm.ucar.edu/cgi-bin/eaton/namelist/nldef2html-pub>).

CAM, CLM, CICE namelist variables that differ from the default values can be specified via the env\_conf.xml file.

However, POP namelist variables that differ from default values may be modified **after** the configure command. Use your line editor of choice (i.e., vi, emacs).

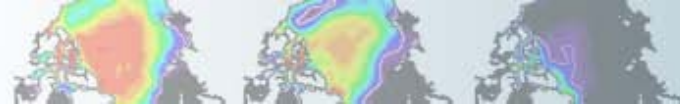


## CESM1 Tutorial: Basic Modifications: **env\_conf.xml**

Sample variables specified in this file include:

1. **RUN\_TYPE** → startup, hybrid, branch
2. **RUN\_REFCASE** → if branch/hybrid, case name you are starting from
3. **RUN\_REFDATE** → if “ “ , date stamp of reference case you are starting from
4. **CCSM\_CO2\_PPM** → CO<sub>2</sub> value to be propagated to CAM and CLM
5. **CAM\_NAMELIST\_OPTS** → **CAM** namelist options that differ from default values
6. **CLM\_NAMELIST\_OPTS** → **CLM** namelist options that differ from default values
7. **CICE\_NAMELIST\_OPTS** → **CICE** namelist options for that differ from default values



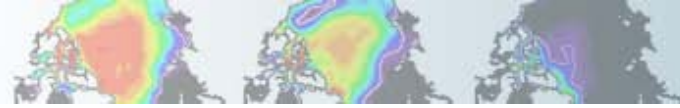


## CESM1 Tutorial: Basic Modifications: **run\_types**

**CESM has four “types” of runs:**

- STARTUP:** All model components are initialized from basic default initial conditions.
- HYBRID:**
- The atmosphere and land are initialized from initial condition files generated by a user-specified CESM simulation.
  - The ocean and ice are initialized from restart files generated by a user-specified CESM simulation.
  - Initial conditions and restart files use the same reference case and reference date.
- BRANCH:** All model components are initialized from restart files generated by a user-specified CESM simulation.
- CONTINUE:** Continuation runs for all run types.

**Default T31\_gx3 cases using compset B\_1850\_CN (INCLUDING OUR TUTORIAL) are HYBRID runs that are initialized from the CESM1 T31x3 1850 Control.**



## CESM1 Tutorial: Basic Modifications: **env\_conf.xml**

**EXERCISE.2:** Create a new fully coupled (with CN) hybrid case from 1850 conditions, but double the CO<sub>2</sub> values and turn on the ice runoff in the land model. Increase the amount of standard out produced by the model and change the default location for the log files to a directory on /ptmp. Run 1 month.

1. from scripts directory

```
create_newcase -case ~/b.day2.2 -res T31_gx3v7 -mach bluefire  
-compset B_1850_CN
```

2. from case directory

```
xmlchange -file env_run.xml -id INFO_DEBUG -val 2
```

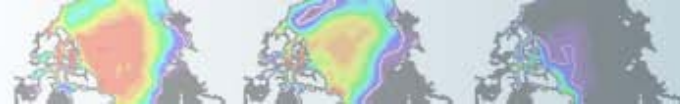
```
xmlchange -file env_run.xml -id STOP_N -val 1
```

```
xmlchange -file env_run.xml -id STOP_OPTION -val nmonths
```

3. *vi* (or *emacs*) *env\_run.xml* to change LOGDIR to */ptmp/\$CCSMUSER/logs/\$CASE* , **or**

```
xmlchange -file env_run.xml -id LOGDIR -val '/ptmp/$CCSMUSER/logs/$CASE'
```

(note that single quotes prevent evaluation of environment variables)



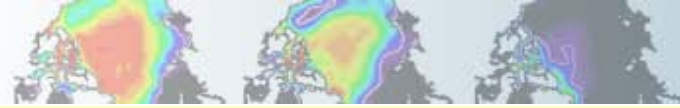
## CESM1 Tutorial: Basic Modifications: **env\_conf.xml**

### EXERCISE.2 continued:

4. `xmlchange -file env_conf.xml -id CCSM_CO2_PPMV -val 569.4`  
`xmlchange -file env_conf.xml -id CLM_NAMELIST_OPTS -val ice_runoff=.true.`
5. `configure -case`
6. Manually update your README.case file to reflect changes
7. `b.day2.2.bluefire.build`
8. Edit `b.day2.2.bluefire.run` and add the following to `#BSUB` lines:
  - a) `#BSUB -P 37591059`
  - b) `#BSUB -U 37591059#1`

**Note this is for tutorial purposes ONLY.**
9. `b.day2.2.bluefire.submit`

Review your Buildconf/\*.buildnml.csh files to verify changes. Review log files to familiarize yourself with standard out. Start looking at your history files.



## CESM1 Tutorial: Basic Modifications: **HOMEWORK**

Before you leave for the day:

**Resubmit and continue to run Exercise 2 for 26 months. This will run overnight in the Bluefire queues.**

Your data will be used for the Practical Session tomorrow on *Diagnostics and Output*. Assuming your `b.day2.2` exercise ran successfully in class....

**In `env_run.xml`:**

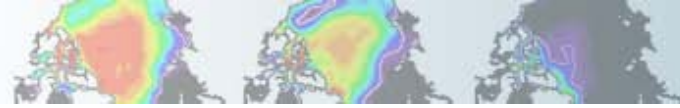
1. Set `CONTINUE_RUN` to `"TRUE"`
2. Set `STOP_OPTION` to `"nmonths"`
3. Set `STOP_N` to `"26"`
4. Set `INFO_DEBUG` to `1`

**In `b.day2.2.bluefire.run`:**

1. Set `#BSUB -W 4:00`
2. **REMOVE** : `#BSUB -U 37591059#1`

**Try to get as many exercises in this presentation completed in class *BEFORE* you go back and set up this continue run.**

**Note:** The reserved nodes are for day-time use only. Overnight jobs only require the project ID specification: `#BSUB -P 37591059`

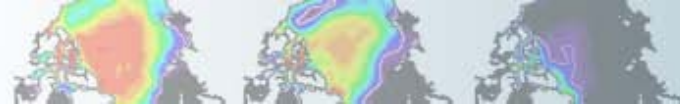


## CESM1 Tutorial: Basic Modifications: **env\_conf.xml**

More about branch runs....

A branch run is useful if you have an experiment which only slightly differs from your control, but you want to initialize with the spun-up basic state of your control. We can do this by initializing the model with restart files from each component model taken from the end of the control.

**Example:** You are running a present day control and have completed 200 years of steady-state (i.e. unchanging) forcing. You want to run a 2xCO<sub>2</sub> experiment off the end of your control. You accomplish this by creating a new case, configuring your model to run as a **BRANCH** case, and specifying the **reference case name** (your control) and **reference case date** (the end date of your control) in your env\_conf.xml file *before* you issue the configure command. You would also change your CCSM\_CO2\_PPMV value in env\_conf.xml to the desired value *before* configuring.



## CESM1 Tutorial: Basic Modifications: **env\_conf.xml**

### **EXERCISE.3: On your own.... (no step by step instructions).**

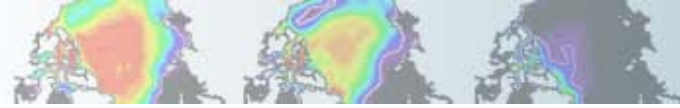
Repeat Exercise 2, except **BRANCH** from the end of Exercise.2. Turn off ice runoff in the land model and quadruple CO<sub>2</sub>. Run 1 month.

Question: You can't use create\_clone from Exercise 2 to create your initial scripts. Why?

Hints: For BRANCH run:

1. You will need to edit env\_conf.xml for branch run specifications, etc.
2. You you will need to manually copy your restart and rpointer files from the Exercise 2 short-term archive directory into your Exercise 3 run directory.

Review: Difference (unix command "diff") Exercise.2 and Exercise.3 buildnml scripts to review differences in namelist scripts for startup cases compared to branch cases.



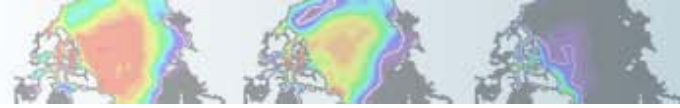
## CESM1 Tutorial: Basic Modifications: **Post configure changes**

### **WARNING!!!!**

The safest way to modify your scripts is via the `env_conf.xml` file **before** you configure.

If you must change your `buildnml` scripts after issuing the `configure` command, be sure you understand exactly what you are doing and how it will impact the model.

It is very easy to create an incompatibility in the scripts. (You should not assume the model will crash. It may run, just give you wrong answers).



## CESM1 Tutorial: Basic Modifications: **Post configure changes**

Examples of **what can be changed** after the configure command has been issued:

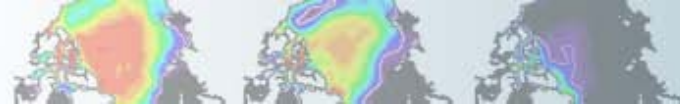
- Swap out a default inputdata set for a home-grown dataset
- Namelist changes for POP or CPL
- Namelist changes that you forgot to include in env\_conf.xml before configuring.
- Namelist changes from buildnml scripts created when cloning another case.

Note: after cloning a case, if you want to cleanup your namelists, invoke the configure `-cleannamelist` command. Otherwise, edit the buildnml scripts by hand.

Examples of **what canNOT be changed** after the configure command has been issued:

- `RUN_TYPE` specifications





## CESM1 Tutorial: Basic Modifications: **Time Step Changes**

### Where and When to Change Time Steps Examples of Pre and Post configure changes

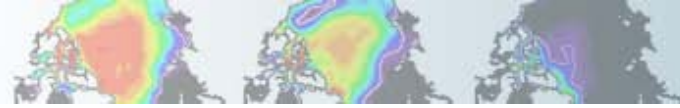
When the model crashes due to large, temporary instabilities, one method to overcome the problem is to change the time step.

This is typically done in either the atmosphere or ocean components.

CAM: `dttime`, in CAM namelist. Change in `env_conf.xml` pre or post configure.

POP: `dt_count` in POP namelist. Edit `pop.buildnml.csh` post configure.

GOTCHAS?

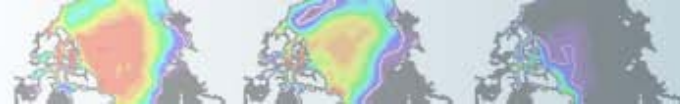


## CESM1 Tutorial: Basic Modifications: **Time Step Changes**

### GOTCHAS

Examples of why we need to know what we are doing!

1. **CLM time step** = CAM time step, `dttime` must be changed in CLM namelist as well
2. **RTM time step** (in `Ind.buildnml.csh`) must be compatible with the CLM time step. The river runoff model (embedded within CLM) requires the number of `rtm` time steps per 3 hours be specified. Therefore, if `dttime` changes, so must `rtm_nsteps`.
3. **ATM\_NCPL** in `env_conf.xml` specifies the number of coupling intervals per day between the atmosphere and the coupled system. The atmosphere and land are coupled every time step. If the CAM time step is changed, `ATM_NCPL` must be updated to reflect the new number of coupling intervals per day.

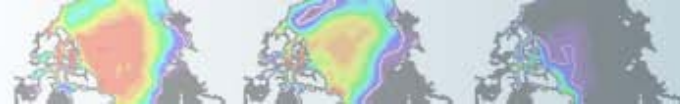


## CESM1 Tutorial: Basic Modifications: **More Exercises**

**Exercise.4:** Clone case from Exercise.1. Quadruple the specified methane trace gas value in the atmosphere, decrease the solar constant by 0.1%, and set the orbital parameters to 1600. (Note: this is an exercise and does not represent any historical period). Run 5 days (default). Resubmit for another 5 days. What happens to *CONTINUE\_RUN* in *env\_run.xml* after completion of the initial 5day run (assuming you set *RESUBMIT* = 1) ?

1. `create_clone -case ~/b.day2.4 -clone ~/b.day2.1`
2. Clean up log and timing files that don't belong to your new case
3. Clean namelists from b.day2.1 by invoking `configure -cleannamelist`
4. `xmlchange -file env_run.xml -id RESUBMIT -val 1`
5. `xmlchange -file env_run.xml -id ORBITAL_YEAR -val 1600`
6. `xmlchange -file env_conf.xml -id CAM_NAMELIST_OPTS -val solar_const=1359.53,ch4vmr=3166.44e-9`

Continued next page



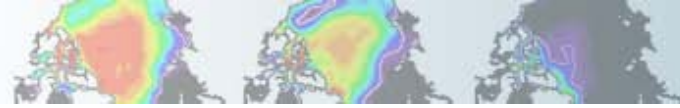
## CESM1 Tutorial: Basic Modifications: **More Exercises**

### Exercise.4: Continued:

7. *configure -case*
8. Update README.case
9. *b.day2.4.bluefire.build*
10. Edit *b.day2.4.bluefire.run* and add the following to *#BSUB* lines:
  - a) *#BSUB -P 37591059*
  - b) *#BSUB -U 37591059#1*

**Note this is for tutorial purposes ONLY.**

11. *b.day2.4.bluefire.submit*



## CESM1 Tutorial: Basic Modifications: **More Exercises**

**Exercise.5:** Clone case from Exercise.4. Include all modifications from Exercise.4. Instead of specifying orbital year, assign individual parameters (eccentricity=0, obliquity=23., and precession=0.) Include new modification to use a different short wave absorption parameterization in POP called “jerlov”. (The default sw absorption parameterization is geography-specific and called “chlorophyll”. “Jerlov” is typically used for paleoclimate simulations where the geography is different from present day). Run 5 days (default).

1. `create_clone -case ~/b.day2.5 -clone ~/b.day2.4`
2. Clean up log and timing files that don't belong to your new case.
3. Keep namelists from previous run because we are keeping all of the modifications. We will adjust orbital parameters in the resolved `cpl.buildnml.csh` file.
4. Edit `Buildconf/cpl.buildnml.csh` and change the following:
  - a. Remove `orb_iyear` and `orb_iyear_align`
  - b. Change `orb_mode` to `'fixed_parameters'`



## CESM1 Tutorial: Basic Modifications: **More Exercises**

### Exercise.5: continued

c. Add the following lines:

```
orb_eccen = 0.
```

```
orb_obliq = 23.
```

```
orb_mvlp = 0.
```

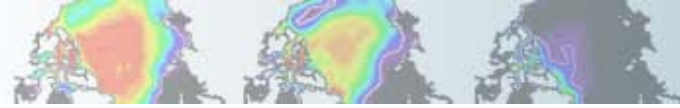
5. Edit Buildconf/pop2.buildnml.csh and change

```
sw_absorption_type = 'jerlov'
```

6. Edit env\_run.xml, change *CONTINUE\_RUN* back to *FALSE*, this is an initial run. Note: you cloned the scripts from b.day2.4 which automatically updated this parameter to *TRUE* upon its resubmission in Exercise 4.

7. *configure -case* (even though namelist are edited by hand, you still need to invoke the configure commands to create your build, run, and archive scripts).

8. Update README.case



## CESM1 Tutorial: Basic Modifications: **More Exercises**

### Exercise.5: continued

9. *b.day.2.5.bluefire.build*

10. Edit *b.day2.5.bluefire.run* and add the following to *#BSUB* lines:

a) *#BSUB -P 37591059*

b) *#BSUB -U 37591059#1*

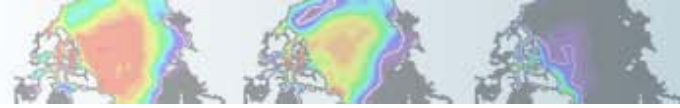
**Note this is for tutorial purposes ONLY.**

11. *b.day2.5.bluefire.submit*

12. Go over all log files and timing files. If you want, you can start to look at the history output. Only the atm/ocn will have daily output to view. (ncview). Where is the short term history output located?

13. Compare *b.day2.5* history data to *b.day2.4* history data. (ncdiff).

**14. Optional:** Resubmit Exercise.4 and Exercise.5 for 1 month to obtain monthly history files for all model components. What will you need to change?



## CESM1 Tutorial: Basic Modifications: **More Exercises**

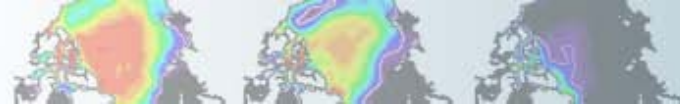
### **Exercise.6: On your own...**

Branch from Exercise.5. Run 5 days (or 1 month). Include all previous modifications. Also double the CO<sub>2</sub> and N<sub>2</sub>O values. CO<sub>2</sub> needs to be consistent across model components. Hint: Some things can be modified via env\_conf.xml and others post configure. Be sure to update your README.case file to keep track of your changes. The model will run regardless of whether or not you remember to include all of your changes. Check your buildnml scripts and the processed namelist files (\$RUNDIR/<model>\_in files) to make sure all changes are included. If you like, resubmit and continue the run for 1 more month.

### **Exercise.7: On your own...**

Do a “startup” run. Change the atm/lnd timestep to 1200 seconds. (Default value is 1800). Run 5 days (or 1 month). Be sure all time steps are compatible. Be sure to update your README.case file to keep track of your changes. Review logs to verify time step changes.





## CESM1 Tutorial: Basic Modifications: **Bottom Line**

**What user-modified “configured” files are actually used at runtime?**

Buildconf/cam.buildnml.csh → \$RUNDIR/atm\_in

Buildconf/clm.buildnml.csh → \$RUNDIR/lnd\_in

Buildconf/pop2.buildnml.csh → \$RUNDIR/pop2\_in

Buildconf/cice.buildnml.csh → \$RUNDIR/ice\_in

Buildconf/cpl.buildnml.csh → \$RUNDIR/drv\_in

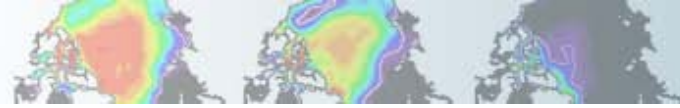
**Bottom Line: User modifications should be implemented in the  
\*.buildnml.csh files.**

**What files are for documentation purposes?**

Buildconf/\*.input\_data\_list, Buildconf/pop2doc

CaseDocs, CaseStatus, LockedFiles, README.case

Note: Buildconf/camconf, Buildconf/clmconf, and Buildconf/ciceconf are used when creating the \*.buildnml.csh files. The user does NOT need to touch these files.



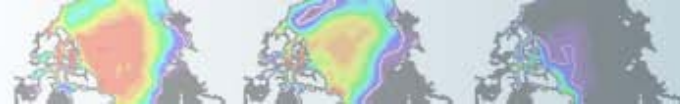
## CESM1 Tutorial: Basic Modifications: **Run Scripts**

**Review run script:** *mycase.bluefire.run*

Common BSUB command to change:

#BSUB -q regular	queue type
#BSUB -o poe.stdout.%J	machine standard out
#BSUB -e poe.stderr.%J	machine standard error
#BSUB -J b.day2.t	job name
#BSUB -W 0:50	wallclock time requested
#BSUB -P <b>37591059</b>	<b>project number</b>
<b>#BSUB -U 37591059#1</b>	<b>dedicated nodes for tutorial</b>

Note: Maximum allowable wall clock time per job submission varies from machine to machine. Bluefire (NCAR's IBM) is 6 hours. Submissions requesting under an hour typically have shorter wait times in the queue.



## CESM1 Tutorial: Basic Modifications: **Log Files**

### Log Files:

During model execution:

After model completion:

atm.log.yyddmm-nnnnnn.gz

lnd.log.yyddmm-nnnnnn.gz

ocn.log.yyddmm-nnnnnn.gz

ice.log.yyddmm-nnnnnn.gz

cpl.log.yyddmm-nnnnnn.gz

ccsm.log.yyddmm-nnnnnn.gz

### Model runtime standard output

\$RUNDIR/\*

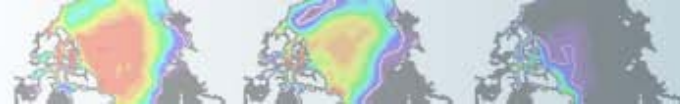
\$LOGDIR/\*

Files are gzipped after model completion.

Restore by typing *gunzip <logfile>*.

yyddmm = year, month, day

nnnnnn = time id stamp



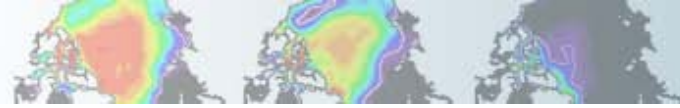
## CESM1 Tutorial: Basic Modifications: **Other Tips**

### **CHECK** your processed namelists!

- After the model builds, it is always good to double check your \$RUNDIR/<model>\_in namelist files. These are the files the model will actually use at runtime and are based on your \*.buildnml.csh files. If there is an error in \*.buildnml.csh, it will show up in \*\_in.
- Also, it is always good to verify that the model is using what you think it is using!

### **DOCUMENT** everything you do!

- A paper trail of your procedures and thoughts is good scientific practice. The README.case file is the perfect place to write notes. You will thank yourself months (years) later, when you are trying to figure out what you did oh-so-long ago!



## CESM1 Tutorial: Basic Modifications: **Post Run Tips**

### Check logs

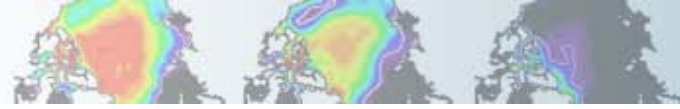
- Check your log files to make sure there are no hidden problems and to verify the model is running smoothly and as you expect. The log files may also help you verify your modifications were included in your run.

### Check output

- Check your history files. It is a good idea to run a small test sample of your experiment before launching your full production run. For example, if you want to run a 500 year control with various modifications, first run 10 years. Check the history output files and verify the model is running as you designed before continuing with the full 500 years. It is always best to find errors early, rather than later, in the run.

### Check timings

- Check your timings. After model completion, a timing subdirectory will be placed in your scripts directory. Check the timings after several job completions to verify that the model is running efficiently and as expected. Double check your timings with the CESM default timings for your specific model resolution and machine. Default timings can be found at: <http://www.cesm.ucar.edu/models/cesm1.0/timing/>



## CESM1 Tutorial: Basic Modifications

**Have Fun!!!**