Atmosphere Breakout Session Exercises

CESM Tutorial
2021

Presented by Dani Coleman
with major contributions from Rich Neale
AMP, CGD, NCAR

NCAR NATIONAL CENTER FOR ATMOSPHERIC RESEARCH
U.S. DEPARTMENT OF ENERGY | Office of Science

Summary

- Run a control case: CAM6, data ocean, climatological forcings from around year 2000. Run for 5 days only, with 3-hourly instantaneous output of a few fields so we can see what is changing
- Then choose one or more exercises to try:
  1. Use historical SSTs/forcings instead of fixed *(compset change)*

  1b. Try running starting 1850 with spun-up pre-industrial model

  2. Increase orographic height over the western US *(dataset change)*

  2b. Modify sea surface temperature in the tropics

  3. Increase the triggering threshold for deep convection over land (*code change--simple*)

  4. Add a (fake) physics parameterization (*code change--advanced*)
- Compare your test exercise to your control

Control Case Setup: Atmosphere-only

To run with prescribed ocean (observed data) and prescribed sea-ice (thickness, area), use compset type F.

Compset Type	Atmosphere	Land	Ocean	Ice
F *(these exercises)*	*Interactive (f19)*	*Interactive (f19)*	Data: SST	Thermodynamic
B *(previous exercises)*	*Interactive (f19)*	*Interactive*	*Interactive (g17)*	*Interactive*

```
cd /glade/p/cesm/tutorial/cesm2.1_tutorial_2021/cime/scripts
```

Create a case with *your choice* of casename (cesm naming convention would be f.e21.F2000.f19_f19.cntl)

```
./create_newcase --case ~/cases/f2000 --res f19_f19 --compset F2000climo  \
                 --run-unsupported
```

Compare RES & COMPSET difference with the B-case we've been running:

(DON'T run this command) `create_newcase --case b.day1.0 --res f19_g17 --compset B1850`

Note: the COMPSET F2000climo is scientifically validated ONLY on the 1deg grid (f09).
We are running 2deg (f19) for expedience.

(Control Case Setup) continued

WARNING: for all these exercises, you must: *unsetenv CESM_BLD_TEMPLATE*

- Configure and Build *Hint: calling `./preview_namelists` from RUNDIR before build command generates the user_nl_* files and allows you to do mods while your build is queued/running*
- Add a history file (h1) to be written every 3 hours, with approximately one file per model month (30 days), containing instantaneous fields 2D (lat/lon) fields. Use these or choose your own:
  TS,PS,Z500,U850,U200,T850,T500,T200,CLDLOW,PRECT,LHFLX,SHFLX,FLNT,FLNS
  Recall atm namelist variables: fincl1, nhtfrq, mfilt
- While you're waiting, decide which of the next exercise you want to do

Unless you're changing the compset (Exercise 1), **you can do the the workflow up through the build just like this page with new casenames for each exercise. These slide titles are in gray boxes so you can search quickly through the document for them.**

- Check completion of build

`cat CaseStatus` (should show 'case.build success')

`ls /glade/scratch/$LOGNAME/$CASENAME/bld/cesm.exe` (should exist)

- Submit

You might need: `xmlchange JOB_QUEUE= whatever_you_need_to_use`

For answer to adding the history file variables, see CNTL Cheatsheet

Comparing Control to Experiments

Once you have completed an exercise, come back to this page for a guide to a quick comparison. Again, the titles in gray boxes are to return to for each case.

Think: *What differences do you expect? Can you see meaningful differences already? Or do you just expect to check that *something* is different because it should be?*

- **Use scripts from Wednesday's diagnostics lab to compare fields**
- **Run the AMWG mean diagnostics for a more comprehensive comparison (needs at least 14 months of data so can't do today)**

- **A quick comparison of two (finished) cases: create & view an ncfile containing the diffs**

You might need to load modules: `module load ncview; module load netcdf`

```
cd /glade/scratch/$LOGNAME/archive
```

If you want to make it easier, use your case names in the following settings so you can copy and use the ncdiff lines)

```
set CNTL = f2000;  set CASE1 = fhist
ncdiff $CASE1/atm/hist/$CASE1.*.nc \
        $CNTL/atm/hist/$CNTL.*.nc \
       $CASE1/atm/hist/diff_${CASE1}_${CNTL}.nc
```

These lines can be mouse-copied but unfortunately must be done one line at a time

Use ncview to browse variables in this file that contains only the diffs.

```
ncview $CASE1/atm/hist/diff_${CASE1}_${CNTL}.nc
```

Note: Confirm fields in your h1 files are instantaneous: use ncdump to compare to h0 files from cases done on the previous days. Look at meta data variable descriptions of averaged: (cell_methods = "time: mean ")

Optional: Shortcuts & Scripts

1) Changing directories quickly: Setting environment variables and aliases can make your work easier.

Command line envvars in: csh or tcsh shells

```
set CASENAME = f2000
set CASEDIR = ~/cases/$CASENAME
```

bash shell

```
export VAR='my value'
export VAR=$MY_OTHER_VAR
```

I set the following in my .tcshrc file, to invoke cdr from the casedir in order to go to the rundir

```
set scr = /glade/scratch/bundy  ; alias cdr 'cd $scr/$CASENAME/run'
set case = ~/cases/$CASENAME;   ; alias cdc 'cd $case'
```

2) Working during the build: Calling `./preview_namelist` from RUNDIR before build command generates the user_nl_* files and allows you to do mods while your build is queued/running

Or back at home, if you're building interactively, redirect output from the case.build into a file, and run the job in the background in order to keep working while the cntl builds,

csh: `qcmd -A $PROJ -- ./case.build >& ! out.build &`

To find job number: `jobs` *To bring jobs to fore/background* `fg %N` *or* `bg %N`

bash: `qcmd -A $PROJ -- ./case.build 1> out.build 2>&1`

3) Put the workflow commands in a shell script instead of using the command line

- Advantages of a script
  - An exact record of the commands used to make each case
  - Easy reproducibility and ability to make small changes for another case
- Disadvantages of a script
  - Necessary to understand a scripting language (example is in c-shell)
  - When the construction of a case requires editing or adding a file, you might have the script stop before the edit and resume afterward. (Note that mine is built to make this as easy as possible).
- **If you want to use a script:** copy mine for the control case `/glade/u/home/bundy/cases/f2000.csh` to **your casename in your directory**, open it in an editor, read and understand it and make the necessary modifications to use it for your cases
- Since this is a more advanced method, there are no more instructions for doing it this way.

CNTL Cheatsheet

1) create, setup,

```
set CASENAME = f2000
set CASEDIR = /glade/u/home/bundy/cases/$CASENAME
set RUNDIR = /glade/scratch/$LOGNAME/$CASENAME/run
cd /glade/p/cesm/tutorial/cesm2.1_tutorial_2021/cime/scripts/
./create_newcase --case $CASEDIR --res f19_f19 --compset F2000climo --run-unsupported
cd $CASEDIR
./case.setup
./preview_namelists                    [Optional] Generates the user_nl_* files to edit while waiting for build
```

2) start build in the background

```
qcmd -A $PROJ -- ./case.build      You might not need the -A $PROJ option right now
```

3) Meanwhile make namelist modifications, in a new shell or by redirecting output & suspending above with ">&! out.build &"

note in csh, >> appends to an existing file in case there are already settings there

```
set nl_file = user_nl_cam
echo "NHTFRQ(2) = -3">> $nl_file
echo "MFILT(2) = 240">> $nl_file
echo "FINCL2 = 'TS:I','PS:I', 'U850:I','T850:I','PRECT:I','LHFLX:I','SHFLX:I','FLNT:I','FLNS:I'">> $nl_file
echo """>> $nl_file
./preview_namelists
```

Calling preview_namelists checks your addition before putting the build into the queue.

If it stalls, something is wrong with your namelist! Try deleting extra spaces at ends of lines, extra lines

4) Wait on build to finish *Commands below check what's going on*

```
qstat -u [your user name]
jobs
cat CaseStatus
```

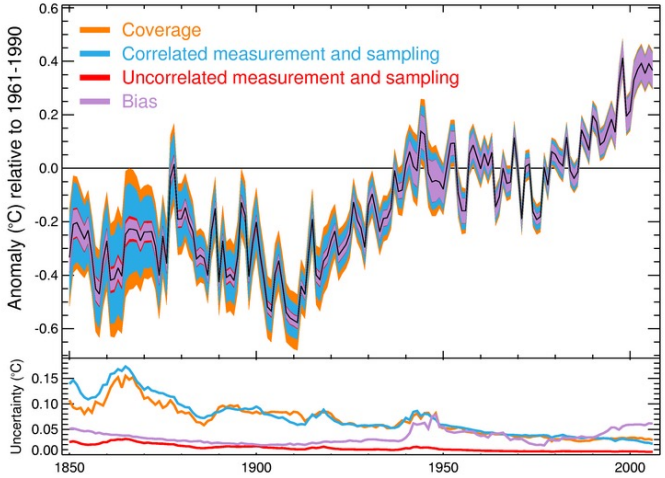
5) Run

```
./case.submit
qstat
```

Monitor output while running: `tail -f /glade/scratch/$LOGNAME/$CASENAME/run/atm.log*`

Exercise 1: Historical boundary conditions and forcings

CAM is capable of running over historical periods with time-varying sea surface temperatures (SSTs) as well as anthropogenic and natural forcings.



Kennedy J.J., Rayner, N.A., Smith, R.O., Saunby, M. and Parker, D.E. (2011b). Reassessing biases and other uncertainties in sea-surface temperature observations since 1850 part 1: measurement and sampling errors. J. Geophys. Res., 116, D14103, doi:10.1029/2010JD015218

More information about the AMIP protocol and HadSST data sets:

- https://climatedataguide.ucar.edu/climate-data/sst-data-hadisst-v11
- http://www-pcmdi.llnl.gov/projects/amip/AMIP2EXPDSN/BCS/bcsintro.php
- Taylor, K.E., D. Williamson and F. Zwiers, 2000: "The sea surface temperature and sea ice concentration boundary conditions for AMIP II simulations" PCMDI Report 60, Program for Climate Model Diagnosis and Intercomparison, Lawrence Livermore National Laboratory,

(Exercise 1): Finding a COMPSET

If you want to run a different configuration from what you've learned here, it is important to learn how to find and/or modify a compset.

Task: Using the tools above and/or web searches below, find a CESM compset with an active atmosphere version cam6.0, that uses historical forcing data including sea surface temperatures. If you find several candidates, look at the components option and/or webpage to decide. Is it scientifically validated? For what resolutions?

Tools in /glade/p/cesm/tutorial/cesm2.1_tutorial_2021/cime/scripts directory :

- `query_config --compsets`
  Gives a list of all the compsets available, and what components are included. This can show exactly what you ran before, and what is possible. To narrow down your choices:

  ```
  query_config --compsets | grep -i hist
  ```

- If you want to build your own, you can see all your options, perhaps modify one of the above with changes
  `query_config --components`

More explanation (and some more options) can be found in the docs
http://www.cesm.ucar.edu/models/cesm2/config/compsets.html
https://ncar.github.io/CAM/doc/build/html/users_guide/atmospheric-configurations.html

(Exercise 1): Setting up Historical

Once you have found your compset, create, configure, build and run, with a casename of your choice (eg. `fhist`), using the same history file output as in the control (see slides with gray-box titles for reference)

- How can you check that there is a difference between the set up of this and your control?
- How can you check that it is running the way you intended: using ssts & ghg forcings from time-varying, historical files?
- What year is the model running?

1b) Additional exercise: Start model in 1850

If you wanted to compare the effects of the standalone-atmosphere to the fully-coupled runs you were doing earlier this week, or you wanted to spin-up a pre-industrial run through the twentieth century, you would need to start your historical run in 1850 instead of the FHIST default 1979.

Hint: On day 2, you learned different ways to start up the model, and how to use initial conditions/restart files from a different case to start up a hybrid case. Use that method

Hint: Data from spun-up model runs can be found in inputdate (DIN_LOC_ROOT) . You'll need to know what generation of the model you're running

(Exercise 1a): Cheatsheet

for simple historical case `fhist`

- Create, setup, build

```
set CASENAME = fhist
set CASEDIR = /glade/u/home/bundy/cases/$CASENAME; set RUNDIR = /glade/scratch/$LOGNAME/$CASENAME/run
cd /glade/p/cesm/tutorial/cesm2.1_tutorial_2021/cime/scripts
./create_newcase --case ~/cases/$CASENAME --res f19_f19 --compset FHIST --run-unsupported
cd ~/cases/$CASENAME
./case.setup
qcmd -- ./case.build
```

- Change namelist settings to get a h1 file (can do while build is running)

Modify user_nl_cam. Note it might have some settings from the compset build so append, don't overwrite

```
NHTFRQ(2) = -3
MFILT(2) = 240
FINCL2 = 'TS:I','PS:I','Z500:I','U850:I','U200:I','T850:I','T500:I','T200:I','PRECT:I','LHFLX:I','SHFLX:I','FLNT:I','FLNS:I'
```

- Check namelist `./preview_namelists`
- Run: `./case.submit`
- Is it working correctly? Quick look in log files (while running). Compare to control. (May need to look in the case.st_archive.o* log file in cntl casedir to find where log files were archived!).

At the top of log file, check model start date.

Then just compare co2vmr values being set between the two runs. Which do you expect to be higher?

```
/glade/u/home/bundy/cases/fhist> grep co2 logs/atm.log.170817-125053
 chem_surfvals_set: ncdate=     19790101   co2vmr=  3.365249938964844E-004
```

How about ssts? Look in ocn.log file for the sst file. Note that 'lb' and 'ub' are lower and upper time bounds being read.

fhist

```
(shr_dmodel_readstrm) file lb: /glade/p/cesmdata/…/sst_HadOIBl_bc_1.9x2.5_1850_2017_c180507.nc    1548
(shr_dmodel_readstrm) file ub: /glade/p/cesmdata/…/sst/sst_HadOIBl_bc_1.9x2.5_1850_2017_c180507.nc    1549
```

f2000

```
(shr_dmodel_readstrm) file lb: /glade/p/cesmdata/…/sst/sst_HadOIBl_bc_1.9x2.5_2000climo_c180511.nc      12
(shr_dmodel_readstrm) file ub: /glade/p/cesmdata/…/sst/sst_HadOIBl_bc_1.9x2.5_2000climo_c180511.nc       1
```

(Exercise 1b): Cheatsheet

for hybrid historical case `fhist.1850`

- Create a new case like in Exercise (1a), except using f09_f09 resolution because that's what the available IC files are.

(I had to start over after it failed to run, don't feel bad if you did too!). Find the full name of resolution by choosing from the list generated by this command in the scripts directory: `query_config --grids | grep f09`

```
set CASENAME = fhist.1850.f09; set CASEDIR = ~/cases/$CASENAME; set RUNDIR = /glade/scratch/$LOGNAME/$CASENAME/run
cd /glade/p/cesm/tutorial/cesm2.1_tutorial_2021/cime/scripts
./create_newcase --case ~/cases/$CASENAME --res f09_f09_mg17 --compset FHIST
```

- Change to hybrid and find a good refcase

```
cd ~/cases/$CASENAME
./xmlchange RUN_TYPE=hybrid 1
./case.setup
```

Finding the initial data to use can be tricky. I found many options under DIN_LOC_ROOT = /glade/p/cesmdata/cseg/inputdata/cesm2_init
and chose one of the latest B1850 pre-industrial control run. I chose one at random but I would never use it for science without advice/checking climate

- Tell model the reference case and reference date to use.

```
xmlchange RUN_REFCASE=b.e20.B1850.f09_g17.pi_control.all.299,RUN_REFDATE=0134-01-01
```

- Find xml variable and set model start date

```
xmlquery --partial DATE      returns all available variables whose names contain "DATE"
xmlchange RUN_STARTDATE=1850-01-01
```

if you forgot this (like I did the first time through) and the model ran with the default 1979, you have to remove rpointer.* from the RUNDIR and copy them again from the REFCASE directory in the next step

- Copy REFCASE data into rundir (do now since build checks for it).

```
cp /glade/p/cesmdata/cseg/inputdata/cesm2_init/b.e20.B1850.f09_g17.pi_control.all.299/0134-01-01/* $RUNDIR
```

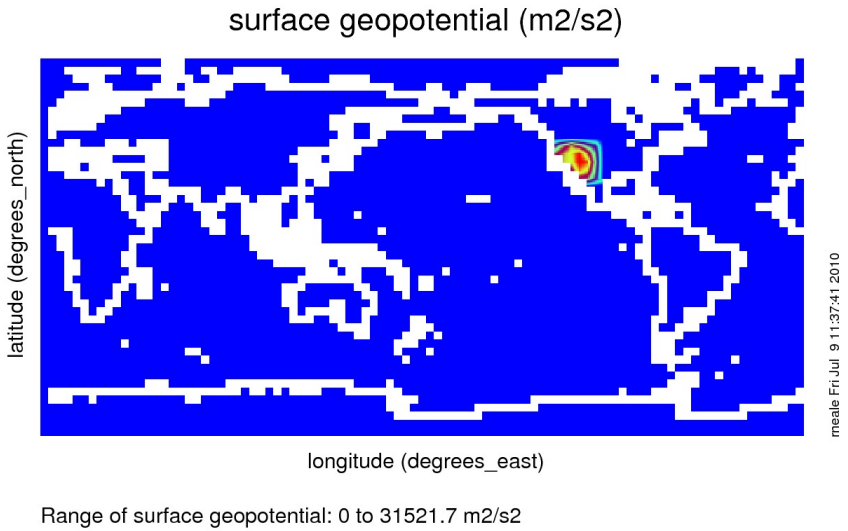
- Build `qcmd -- ./case.build`
- Change namelist settings to get a h1 file (below), Run

```
NHTFRQ(2) = -3; MFILT(2) = 240; FINCL2 =
'TS:I','PS:I','Z500:I','U850:I','U200:I','T850:I','T500:I','T200:I','PRECT:I','LHFLX:I','SHFLX:I','FLNT:I','FLNS:I'
```

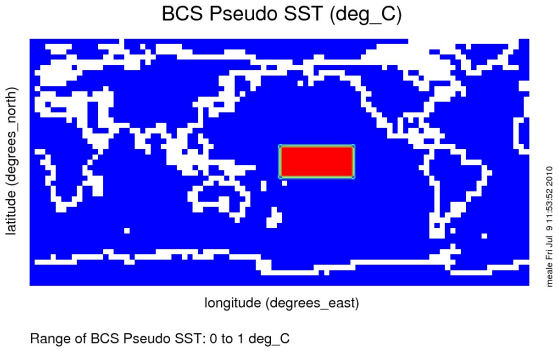
- Run `./case.submit`
- Check: `cat CaseStatus` As designed, this run should fail. There is an easy fix, given in the log file that CaseStatus points you to. The rest of this exercise is for you to find and correct the error. Doing such will occupy much of your future running climate models.

Exercise 2a: Dataset change

- Change input boundary datasets by increasing surface geopotential height by 50% in the western USA



See alternative case 2b to add a Sea Surface Temperature (SST) anomaly instead



(Exercise 2a) Dataset change

1) Follow control case set up through the build, with a new casename (eg. f2000.oro)
2) Find name & path of boundary dataset (*Hint: even though I named my case oro for orography, the file that specifies the geopotential height is a **topo**graphy file*)
3) Make a local copy of the boundary file in your case directory
4) Modify the boundary file from command line with NetCDF Operator suite (*see next slide for sophisticated command)*
5) Check your new dataset (*use ncdiff, ncview)*
6) Edit atmosphere namelist to point to modified dataset and check resolved nml
7) Run and check that the correct files is being used

What to look for when experiment is complete?

(See Comparing to Control script for suggestions on how to look)

✓ *Surface temperature (TS), pressure (PS), winds (U850), cloud, rainfall (PRECT),*

(Exercise 2a) NetCDF Operators (NCOs)

NetCDF Operators (NCOs) are valuable tools that can edit NetCDF files from the command line

http://nco.sourgeforce.net . We will use a function called ncap2 (netCDF Arithmetic Averager), to demonstrate the power of the NCOs. There are many simpler tools, eg. ncks (kitchen sink)

You must find the input dataset and copy it to your casedir before proceeding…

```
ncap2 -O -s 'lat2d[lat,lon]=lat ; lon2d[lat,lon]=lon' \
-s 'omask=(lat2d >= 30. && lat2d <= 50.) && (lon2d >= 235. && lon2d <= 260.)'\
-s 'PHIS=(PHIS*(1.+omask*0.5))'\
fv_1.9x2.5_nc3000_Nsw084_Nrs016_Co120_Fi001_ZR_GRNL_031819.nc \
fv_1.9x2.5_nc3000_Nsw084_Nrs016_Co120_Fi001_ZR_GRNL_031819.orox50.nc
```

1. *Define 2D latitude and longitude arrays*
2. *Create a mask – setting = 1 for the desired lat/lon range; elsewhere = 0*
3. *Apply mask to 2D field (PHIS = surface geopotential = gZ) – 1.5x PHIS in region of interest*
4. *Infile Outfile*

Mouse-copy all of the FOUR lines of this command into ONE line in your shell

Check your changes by viewing a netcdf file containing the diffs (new – old)

hint: ncdiff old.nc new.nc diff.nc, ncview diff.nc (make sure to check all values!)

(Exercise 2a) Cheatsheet

1) Follow **control case set up** through the build and namelist modifications but with a new casename.

```
set CASENAME = f2000.oro; run create_newcase, case.setup, preview_namelists
```

2) Find name & path of boundary dataset

```
cd ~/cases/$CASENAME; grep -i topo env*.xml CaseDocs/*_in
```

(Returns: /glade/u/home/bundy/cases/f2000.oro/CaseDocs/atm_in: **bnd_topo = '/glade/p/cesmdata/cseg/inputdata/atm/cam/topo/fv_1.9x2.5_nc3000_Nsw084_Nrs016_Co120_Fi001_ZR_GRNL_031819.nc')**

3) Make a local copy of the boundary file in your CASE directory

```
set file_orig = fv_1.9x2.5_nc3000_Nsw084_Nrs016_Co120_Fi001_ZR_GRNL_031819.nc
cp /glade/p/cesmdata/cseg/inputdata/atm/cam/topo/$file_orig .
```

4) Modify file with NCOs and check (detailed on previous slide)

```
set file_new = $file_orig:r.orox50.nc
ncap2 -O -s 'lat2d[lat,lon]=lat ; lon2d[lat,lon]=lon' \
  -s 'omask=(lat2d >= 30. && lat2d <= 50.) && (lon2d >= 235. && lon2d <= 260.)'\
  -s 'PHIS=(PHIS*(1.+omask*0.5))' $file_orig $file_new
```

5) Check your changes

```
ncdiff -v PHIS $file_orig $file_new PHIS_diff.nc; ncview PHIS_diff.nc
```

You have to click ok to make ncview check ALL of the data; subsampling only finds the zeros

6) Edit user namelist to point to modified dataset, check that it makes it to rundir resolved namelist

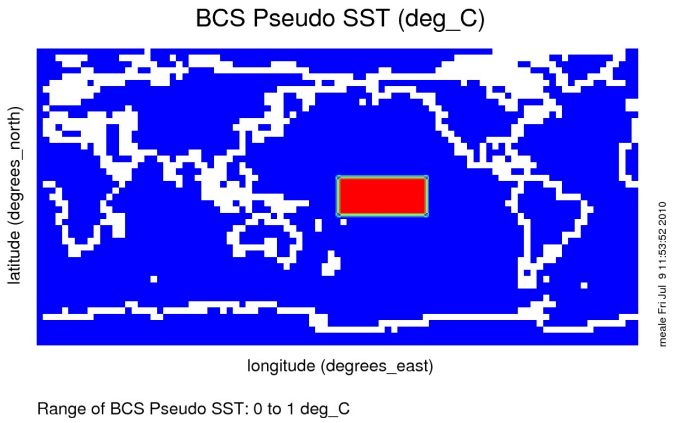
```
echo "bnd_topo = '$CASEDIR/$file_new'" >>& user_nl_cam
./preview_namelists
grep topo /glade/scratch/$LOGNAME/$CASENAME/run/atm_in
```

7) Build & Run and check that the correct files is being used in the logs!

```
cat CaseStatus | grep build
if build successful ./case.submit
Once running: grep orox50 $RUNDIR/atm.log*
OR after running: grep orox50 /glade/scratch/$LOGNAME/archive/$CASENAME/logs/atm.log*
```

Exercise 2b: *Dataset change*

- ACTION: Change input boundary datasets (Sea Surface Temperature) by increasing it's value by 2K in the tropical Central Pacific



(Exercise 2b) Dataset change

1) Follow control case set up through the build with a new casename (eg. f2000.sst)
2) Find name & path of sst dataset (*Hint: look in env_*.xml*)
3) Make a local copy of the sst file in your case directory
4) Modify sst file with NetCDF Operator suite (*see next slide for sophisticated command*)
5) Check your new dataset (*use ncdiff, ncview)*
6) Edit env_run.xml to point to modified dataset and check that the change when into the resolved namelists
7) Run model and check that the correct files is being used

What to look for when experiment is complete?

(See slide 6: Comparing to Control)

- ✓ *Surface temperature (TS), surface pressure (PS), cloud (CLDLOW), rainfall (PRECT), winds (U850)*

(Exercise 2b) *Dataset change*

NetCDF Operators (NCOs) are valuable tools that can edit NetCDF files from the command line

http://nco.sourgeforce.net . We will use a function called **ncap2 (netCDF Arithmetic Averager)**, to demonstrate the power of the NCOs. There are many simpler tools, eg. ncks (kitchen sink)

Note: you must first find the input dataset and copy it to your casedir before proceeding…

```
ncap2 -O -s 'lat2d[lat,lon]=lat ; lon2d[lat,lon]=lon' \1
-s 'omask=(lat2d >= -10. && lat2d <= 10.) && (lon2d >= 180. && lon2d <= 240.)'\2
-s 'SST_cpl=(SST_cpl+omask*2.)'\3
sst_HadOIBl_bc_1.9x2.5_2000climo_c180511.nc \                              4
sst_HadOIBl_bc_1.9x2.5_2000climo_c180511.nc.warmtcp.nc
```

1. *Define 2D latitude and longitude arrays*
2. *Create a mask – setting = 1 for the desirl,lpled lat/lon range; elsewhere = 0*
3. *Apply mask to 2D field (SST_cpl): +2K in region of interest (equatorial Pacific)*
4. *Infile/Outfile*

Mouse-copy all of the FOUR lines of this command into ONE line in your shell

Check your changes by viewing a netcdf file containing the diffs (new – old)

(Exercise 2b) Cheatsheet

1) Follow **control case set up** through the build (Slide 7 CNTL Cheatsheet) but with a new casename.

```
set CASENAME = f2000.sst; run create_newcase, case.setup, preview_namelists
```

2) Find name & path of boundary dataset. Look in xml file first

```
grep -i sst env*.xml | grep -i file
```

Returns a lot hence the | (pipe) into grep. Decide to change the one that is a filename

env_run.xml: <entry id="SSTICE_DATA_FILENAME" value="$DIN_LOC_ROOT/atm/cam/sst/sst_HadOIBl_bc_1.9x2.5_2000climo_c180511.nc">

3) Make a local copy of the sst file in your CASE directory

How to know find DIN_LOC_ROOT? Probably set in an xml file somewhere but we can find the whole path by searching the resolved

```
grep sst_Had CaseDocs/*_in
```

Returns: CaseDocs/ice_in: stream_fldfilename = "/glade/p/cesmdata/cseg/inputdata/atm/cam/sst/sst_HadOIBl_bc_1.9x2.5_2000climo_c180511.nc"

```
set file_orig = sst_HadOIBl_bc_1.9x2.5_2000climo_c180511.nc
cp /glade/p/cesmdata/cseg/inputdata/atm/cam/sst/$file_orig .
```

4-5) Modify file with NCOs and check using NCVIEW (see previous slide)

```
set file_new = $file_orig:t:r.warmtcp.nc
ncap2 -O -s 'lat2d[lat,lon]=lat ; lon2d[lat,lon]=lon' \
   -s 'omask=(lat2d >= -10. && lat2d <= 10.) && (lon2d >= 180. && lon2d <= 240.)'\
   -s 'SST_cpl=(SST_cpl+omask*2.)' $file_orig $file_new
ncdiff -v SST_cpl $file_orig $file_new SST_cpl_diff.nc
ncview SST_cpl_diff.nc
```

6) Get the namelist change into the actual namelists and check the resolved namelists

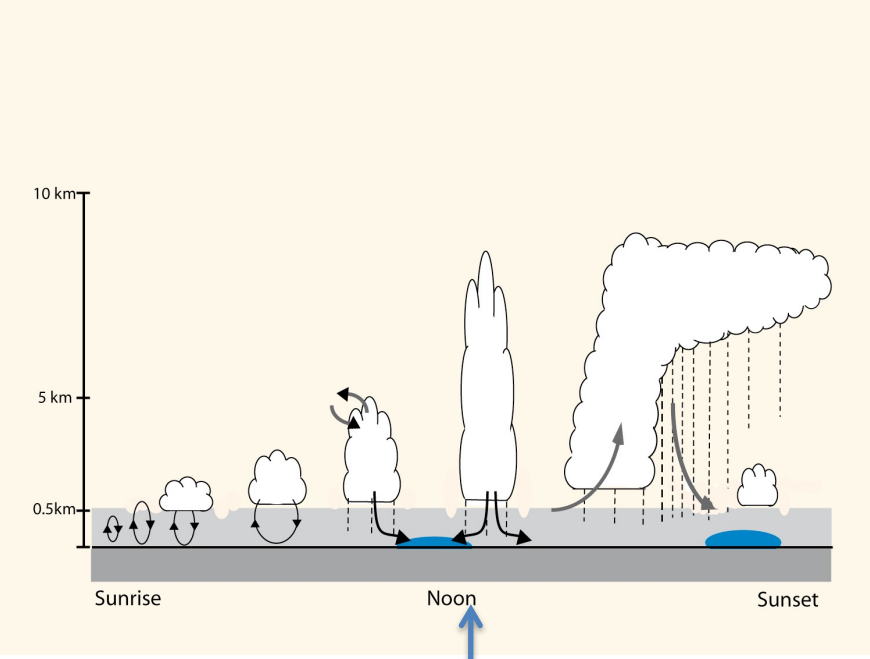
```
./xmlchange SSTICE_DATA_FILENAME="$CASEDIR/$file_new"
./preview_namelists
grep sst $RUNDIR/*_in      Note: these don't show up in *usr_nl*, but are in the final input
```

7) Build. Check that the correct files is being used

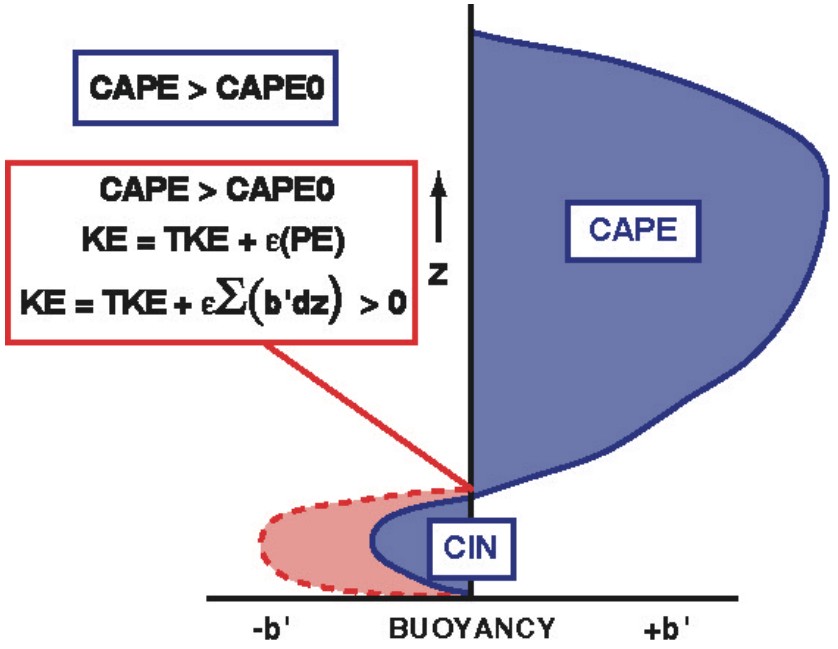
```
cat CaseStatus | grep build
if build successful ./case.submit
while running       grep warmtcp $RUNDIR/*.log*
after running       grep warmtcp /glade/scratch/$LOGNAME/archive/$CASENAME/logs/atm.log*
```

Exercise 3: Code Change

- Examine the mean effects of delaying the initiation of convection by increasing the minimum required convective available potential energy (CAPE) to initiate convection over land;



Models peak around noon
Too early



(Exercise 3) : Description

- Create a new case (eg f2000.cape) like the control. Setup and build.
  - Note that case.build is smart enough to only re-compile the changed files and those dependent on them. So build now, and then again after you've made modifications.
- Meanwhile… Find the Fortran module that does deep convection.
  - Hint: the parameterization has only changed incrementally since 1995 so a Scientific Description of any version of CESM/CCSM/CAM should give you the name for it
  - Hint: the model source code is up a few directories from the path you use to call create_newcase. CAM model source code is under components/atm/src
  - Hint: the calculations in CAM are divided into dynamics and physics. Very roughly speaking, dynamics deals with sideways processes and physics with those that act in the vertical direction. Under which do you think the convection code will be?
- Modify the code
  In the file (copied to your `$CASEDIR/SourceMods/src.cam` directory, of course), modify the minimum CAPE threshold necessary for triggering convection from 70 J/kg to 700 K/kg. (More difficult: do it only over land.

What to look for ? (Slide 6: Comparing to control)

- ✓ *Surface temperature (TS), surface fluxes (LHFLX,SHFLX), cloud (CLDLOW), rainfall (PRECT) over land*
- ✓ *Does it have any effect on the diurnal cycle?*

(Exercise 3) : Cheatsheet p1.

- Follow guide for control case including build with new casename (eg. f2000.cape)

Note that case.build is smart enough to only re-compile the changed files and those dependent on them. So build now, and then again after you've made modifications.

```
set CASENAME = f2000.cape;
create_newcase, case.setup, preview_namelists, build
```

Copy fortran code file that controls deep convection calculation (Zhang and McFarlane, 1995) to local code modification directory for the atmosphere

```
cd $CASEDIR
cp /glade/p/cesm/tutorial/cesm2.1_tutorial_2021/components/cam/src/physics/cam/zm_conv.F90 \
SourceMods/src.cam
```

- Continued on next page…

(Exercise 3) : Cheatsheet p2.

Change the following section of code in sub-routine ***zm_convr***

```
if (cape(i) > capelmt) then
 lengath = lengath + 1
 ideep(lengath) = i
end if
```

To

```
if (landfrac(i) > 0.5_r8) then
 capelmt_mask = 10._r8*capelmt
else
 capelmt_mask = capelmt
end if

if (cape(i) > capelmt_mask) then
 lengath = lengath + 1
 ideep(lengath) = i
end if
write(iulog,*) 'HELLO WORLD'
```

Near the top of subroutine zm_convr, right after:

```
 real(r8) pblt(pcols)
```

Add `real(r8) :: capelmt_mask`

Where land fraction is >50%,

Scale the existing trigger value by 10x

Create a local variable called capelmt_mask to use as the convection trigger

Put text in the log files to make sure the changes got in (search while running!)

Declare the capelmt_mask variable

Check your changes by 'diff new old' files

(Exercise 3) : Cheatsheet p3.

- build `./case.build`

Only the first big build needs to go through the queue, (for re-builds ok to call case.build from the command line)

  - check that build was successful: CaseStatus file in CASEROOT should show 'build success'

    If it wasn't successful, CaseStatus should provide a link to the log file from the build.

  - check that **new code was incorporated into build**, **check time stamp on object files**.
    Assuming you built once before modifying code, the modified routine (and those affected by it) should be newer than most of the object files *and* the executable should show the newest time stamp, too.

timestamp on files shows that zm_conv are newer than most of the other files.

```
ls -ltr /glade/scratch/$LOGNAME/$CASENAME/bld/atm/obj
... [more files above] ...
-rw-r--r-- 1 bundy ncar  10504 Jul 25 14:47 clubb_intr.o
-rw-r--r-- 1 bundy ncar 269592 Jul 25 14:47 micro_mg_cam.o
-rw-r--r-- 1 bundy ncar 107487 Jul 25 14:47 microp_driver.mod
-rw-r--r-- 1 bundy ncar  10416 Jul 25 14:47 microp_driver.o
-rw-r--r-- 1 bundy ncar    883 Jul 25 16:08 tmp_filepath
-rw-r--r-- 1 bundy ncar  31004 Jul 25 16:08 zm_conv.mod
-rw-r--r-- 1 bundy ncar 230648 Jul 25 16:08 zm_conv.o
-rw-r--r-- 1 bundy ncar  47994 Jul 25 16:08 convect_deep.mod
-rw-r--r-- 1 bundy ncar  84350 Jul 25 16:08 zm_conv_intr.mod
-rw-r--r-- 1 bundy ncar 141424 Jul 25 16:08 zm_conv_intr.o
... [a few files below] ...
```

- run `./case.submit`

While it is running, you can look in the log files to see if your print statement is being called.

```
/glade/scratch/bundy/f2000.cape/run> grep HELLO *.log*
   cesm.log.1586924.chadmin1.180809-161410:37: HELLO WORLD
   ....
```

Take this print statement out before a long run, but for now it lets you know that your changes are active.

Next, of course, is to check the history files. You probably won't be able to tell in a 5-day run what is changing, but it's time to start thinking about how long of a run you do need to do.

Exercise 4 *Code Change*

Add a parameterization

Exercise 4 is a more general, free-form exercise to prepare you to do major modifications or replace an existing parameterization.

Warning: details in this exercise refer to out-dated code.

However, it is a useful skill as a modeler to be able to adapt out-of-date information

This exercise will provide guidance if you want to **add a parameterization** to CAM, including these topics:

a) The requirements for a parameterization and interface
b) Finding source code (reading documentation, browsing and smart searching)
c) A 'stub' parameterization to add to CAM (the subroutines & calls without content code)
d) References to CAM physics code details

First: Create, setup and build a new case following the control case instructions but with a unique name, eg. *f2000.param*

We will re-build after modifying the source code, meanwhile using some files created by the build to navigate through the source code.

(Exercise 4) *Code Change*

Parameterization requirements

A physics parameterization

1. Must calculate a tendency (rate of change)
2. Must not change the model state
3. Must conserve vertical integrals of
   - `mass`
   - `momentum`
   - `total energy`
   - `dry static energy`

The **tendencies** from each physics parameterizations are used in `physpkg.F90` (subroutines `tphysbc` and `tphysac`) to calculate the new model **state**, along with checks that energy and water balance.

eg. in /glade/p/cesm/tutorial/cesm2.1_tutorial_2021/components/atm/cam/src/physics/cam/physpkg.F90

`call convect_shallow_tend(…)` is followed by the update to the state:

```
call physics_update(state, ptend, ztodt, tend)
```

(Exercise 4) *Code Change*

Interfacing recommendations

Any parameterization should consist of two parts:

1. An **interface** layer to communicate between CAM and the parameterization
2. The **parameterization** package, with as little of CAM structures in it as possible

This helps make a parameterization which is portable between different models or model versions.

(Exercise 4): *Code Change*

Routines that every parameterization interface might need

The public methods of a CAM interface

(eg. `PARAM` means the name of the parameterization, like `convect_shallow`)

`PARAM_register`

Register fields in the physics buffer, register constituents in the constituent arrays.

`PARAM_init`

Package specific initialization at beginning of run, including setting time-invariant variables

`PARAM_timestep_init`

Per-timestep initialization, (e.g. time interpolation from a boundary dataset.

`PARAM_timestep_tend`

Calls the package run method which computes the tendencies for each model timestep.

And methods for parameterizations that introduce constituents:

`PARAM_init_cnst , PARAM_implements_cnst`

See online document for details about interfaces, along with other useful CAM physics info
http://www.cesm.ucar.edu/models/atm-cam/docs/phys-interface/ (Utility Modules section)
or browse code for examples.

(Exercise 4): *Code Change*

CAM example of interface & parameterization: ***initialization***

Interface = models/atm/cam/src/physics/cam/convect_shallow.F90 methods call:
Parameterization/ package: uwshcu.F90 (University of Washington shallow convection scheme)

1. Registration (allocates memory at beginning of model run)

```
physpkg.F9:phys_register()
    call convect_shallow_register
    convect_shallow.F90:convect_shallow_register()
        call uwshcu_register
```

2. Initialization (done once at the beginning of model run or restart)

```
physpkg.F90:phys_init()
      call convect_shallow_init
      convect_shallow.F90:convect_shallow_init()
         call uwshcu_init
```

`physpkg` doesn't know anything about the shallow convection package, so a new parameterization could be swapped in with modifications only to the **interface** `convect_shallow.F90`. Also, changes in CAM model shouldn't touch `uwshcu.F90`; just `convect_shallow.F90`.

(Exercise 4): *Code Change*

CAM example of interface & parameterization: ***time-stepping***

3. Time-stepping (as the model is running).
`tphysbc` gets a tendency `ptend` from an interface method (1), updates the model `state` with that tendency (2), and checks conservation (3).

```
physpkg.F90:phys_run1()
1. call convect_shallow_tend(state, ptend,…)
      convect_shallow.F90:subroutine convect_shallow_tend(state,
ptend,…)
        intent(in) state
        intent(out) ptend

        call compute_uwshcu_inv(… state    ,ptend …

2. call physics_update(state, ptend,…)
3. call check_energy_chng
```

`Intents` show that the interface can only modify ptend, not state. The `ptend` is returned from the parameterization method

(Exercise 4): *Code Change*

Finding source code

Now that you have been introduced to the requirements for a parameterization and recommendations for an interface how do you find an existing parameterization or know where to put a new one?

- If replacing an existing parameterization
  - **Read documentation** to find names of existing routines
  - **Browse** the code
  - Use **grep/find** or grepccm/findccm (described later)

  to find all the source code references

- If adding a new parameterization
  - **Browse** or use grep/find tools to find a parameterization to use as a model

- The existing parameterizations can serve as **examples to follow**, but some have better code than others!

(Exercise 4): *Code Change*

Reading documentation

If you want to replace an existing parameterization, you may need to first **learn what is already in CAM**, in order to find the correct search terms.

For instance, search the CAM5 Scientific Description for "`shallow convection`."

This doesn't tell you much more than the name of the new scheme in CAM5 and what it replaces, but it's a start.

http://www.cesm.ucar.edu/models/cesm1.0/cam/docs/description/cam5_desc.pdf

```
4.3 Shallow Convection Scheme

Shallow convection scheme in CAM5 is from Park and Bretherton [2009]
that is a replacemnt of Hack [1994b]…
```

(Exercise 4): *Code Change*

Browsing the source code

The source code is under the `$rootdir` (recall where you go to call `create_newcase`).

The source for CAM is in `$rootdir/models/atm/cam/src`
With subdirectories
`advection/ chemistry/ control/ cpl_esmf/ cpl_mct/ cpl_share/`
`dynamics/ physics/ utils/`

This exercise mostly makes changes to code in the physics/cam subdirectory but I'll point out `control/` as a useful directory, holding physical constants, interpolation routines and, indeed, the top level module for the CAM model component, `cam_comp.F90`

You can find a lot by looking through source code files. For our shallow convection example, notice there is a file called `models/atm/cam/src/physics/cam/convect_shallow.F90`
Looking at, you can see there are different convection options (

```
! The following namelist variable (shallow_scheme) controls which shallow conv package is used.
!        'Hack'    = Hack shallow convection (default)
!        'UW'      = UW shallow convection by Sungsu Park and Christopher S. Bretherton
!        'off'     = No shallow convection
```

(Exercise 4): *Code Change-*

Optional tools to search CAM code

- UNIX commands `find` and `grep` can do the job
- However, CAM source code is
  - in a complicated directory structure *and*
  - any particular model build uses only a subset of the source code (eg. dycores)

So we have tools that use `find` and `grep` **within** this directory structure

- These tools are available with this tutorial although their use is optional;

If you want to use them in the future, copy them to your home machine before you go

`grepccm string` *searches for string in all files in the directories used by this particular model build*

`findccm filename` *searches for a file named filename in all the directories used by this model build*

- For now, copy `/glade/u/home/bundy/bin/grepccm` and make executable

```
cp ~bundy/bin/grepccm ~; chmod 755 ~/grepccm
cp ~bundy/bin/findccm ~; chmod 755 ~/findccm
```

- Change to directory containing CAM's Filepath in order to use!

```
cd /glade/scratch/$LOGNAME/$CASENAME/bld/atm/obj
ls Filepath
```

(Exercise 4): *Code Change*

Optional tools to search CAM code (details)

Filepath is a text file listing all the directories containing source code that are *actually* used in the model build.

For example (yours may look different):

```
% cat /glade/scratch/$LOGNAME/$CASENAME/bld/atm/obj/Filepath
/glade/u/home/bundy/cam/case/tutorial/f2000/SourceMods/src.cam
/glade/p/cesm/tutorial/cesm2.1_tutorial_2021/components/atm/cam/src/chemistry/mozart
…
/glade/p/cesm/tutorial/cesm2.1_tutorial_2021/components/atm/cam/src/physics/cam
/glade/p/cesm/tutorial/cesm2.1_tutorial_2021/components/atm/cam/src/dynamics/eul
/glade/p/cesm/tutorial/cesm2.1_tutorial_2021/components/atm/cam/src/control
/glade/p/cesm/tutorial/cesm2.1_tutorial_2021/components/atm/cam/src/utils
```

You must be in a directory containing Filepath to use `grepccm` and `findccm`

- `grepccm` searches for ***a string*** in each file in each directory listed in Filepath

```
cd /glade/scratch/$LOGNAME/$CASENAME/bld/atm/obj/
~/grepccm zhang
~/grepccm "calculates cape"
```

- `findccm` searches for ***a file*** in each directory listed in Filepath

```
cd /glade/scratch/$LOGNAME/$CASENAME/bld/atm/obj/
~/findccm convect_shallow.F90
```

•

(Exercise 4): *Add a parameterization- Do it!*

This exercise has two steps:

1. Add a new parameterization: copy a stub parameterization to your SourceMods dir
2. Interface the new param with a stub.
3. Call the interface methods from CAM physics code
4. Build and run to test your set up.

Once you've completed these 'toy' exercises, you will be free to expand upon them or start using your own work.

(Exercise 4)

Add a new (fake) parameterization

To create a new file, you need to place valid source code in SourceMods/src.cam
My advice is to copy an existing interface & parameterization to a new name,
and remove most of the code to create a 'stub' parameterization.
Code that is newer to the model might have better coding practices… or might not.

Whenever you create a new file, you need to force the build process to notice it:

```
% cd /glade/scratch/$LOGNAME/$CASENAME/bld/atm/obj
% rm Depends
% rm Srcfiles
% cd $CASEROOT
% case.build
```

Check that the new file was compiled by looking at timestamp of .mod/.o files in bld dir

```
% ls -ltr /glade/scratch/$LOGNAME/$CASENAME/bld/atm/obj/
```

You should **see** param, param_interface files and timestamp of physpkg.* should be **later** than original build.

(Exercise 4): *Add a parameterization*

What's next?

After completing these 'toy' exercises, you may want to:

- have the new param add a field to the history file (addfld/outfld)
- do something with the physics buffer (pbuf) in the new parameterization
- write code in param.F90 to do something more. Check that it's working.
- make a modification of your choice to an existing parameterizations
- … your choice

Due to the open-ended nature of this set of exercises,
please don't use the helper resources for your own project…
if you've advanced to this point,
you should start the practice of finding answers yourself. ;)

See following slides for references about the model code

References

More about coding in CAM

WARNING: Some of these are quite outdated

- CAM reference manual

http://www.cesm.ucar.edu/models/cesm1.2/cam/docs/rm5_3/

Although outdated, this provides details of how the CAM routines are called, the data structures (state, ptend), the array dimensions (chunks and columns), and descriptions of subcolumns and radiative constituents.

- CAM coding standards

https://wiki.ucar.edu/display/ccsm/Draft+of+Coding+Standards+for+CAM

- Unit testing

http://www.cesm.ucar.edu/events/ws.2014/Presentations/SEWG/santos.pdf

- Physics interface design

http://www.cesm.ucar.edu/models/atm-cam/docs/phys-interface/

References

Stand-alone , Single-Column CAM

- Users who are developing CAM might be interested in the specific CAM configure/run instead of dealing with the entire CESM structure

  `components/cam/bld/configure (among others)`

- Single-column CAM is a good tool for developing physics parameterizations
  - no dynamics
  - runs with field-experiment data

  http://www.cesm.ucar.edu/models/simpler-models/scam/

References

CESM bulletin board