Introduction to CESM
Hands-on Tutorial

Cecile Hannay, CGD/NCAR
Outline

• CESM Workflow: Super Quick Start
• Change the length of a run. Extend a run
• Output files, log and timing files
• Namelist modifications
  - Changing CO2
  - Changing output frequency and history files
• Troubleshooting and log files
• Code modification
• Types of runs: startup, branch and hybrid.
• The one-million things we didn’t cover in this tutorial
Navigating the CESM Maze

When you run CESM, you have to navigate between 5 directories:
- CESM code
- Inputdata directory
- Case directory
- Build/run directory
- Archive directory

Each of these directories contains sub-directories.

One of the goal of this tutorial is to teach you how to navigate between these directories.
In this section, we will cover the basic commands to run CESM. At the end of this section, you will run your own experiment.
Set of commands to run CESM

You can run CESM with only a set of 4 commands!

```
# go into scripts directory into the CESM code
cd $CESMROOT/cime/scripts

# (1) create a new case in the directory “cases” in your home directory
./create_newcase --case ~/cases/case01 --compset B1850 --res f09_g17

# go into the case you just created in the last step
cd ~/cases/case01/

# (2) setup your case
./case.setup

# (3) build the executable
./case.build

# (4) submit your run
./case.submit
```

It is that easy! 😊
Set of commands to run CESM for this tutorial

Today: we only have 4 hours => Let’s make it faster

# go into scripts directory into the CESM code
```bash
cd $CESMROOT/cime/scripts
```

# (1) create a new case in the directory “cases” in your home directory
```bash
./create_newcase --case ~/cases/case01 --compset B1850 --res f09_g17
```
```bash
case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```

# go into the case you just created in the last step
```bash
cd ~/cases/case01/
```

# (2) setup your case
```bash
./case.setup
```

# (3) build the executable
```bash
./case.build
```

# (4) submit your run
```bash
./case.submit
```

To make it faster: we use a cheaper version of the atmospheric model (CAM4) and a lower resolution. All the concepts you learn here will work with the standard model.
Overview of CESM directories before you do anything

This is where the CESM inputdata lives.
(topography, initial conditions, greenhouse gases,...)

This is where the CESM code lives.

The cime directory contains all the scripts you need to run CESM.

The components directory contains all the CESM code.
Overview of CESM directories before `create_newcase`

**CESM Code**

```
$CESMROOT
  components
  cime
    scripts
      (1) create_newcase
```

**INPUTDATA Directory**

```
$CESMDATAROOT/inputdata
  share
cpl
  atm
  Ind
  ocn
  ice
  glc
  wav
  rof
```

- **Go into scripts directory into the CESM code**
  ```
cd $CESMROOT/cime/scripts
  
  (1) create_newcase
```

- **Create a new case in the directory “cases” in your home directory**
  ```
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```

- **Go into the case you just created in the last step**
  ```
  cd ~/cases/case01/
```

- **Setup your case**
  ```
  ./case.setup
```

- **Build the executable**
  ```
  ./case.build
```

- **Submit your run**
  ```
  ./case.submit
```
Overview of CESM directories before create_newcase

$CESMROOT

components

scripts

(1) create_newcase

create_newcase requires 3 arguments

What is the casename?

Which model configuration?
Which set of components?
Which resolution?

./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
Overview of CESM directories after create_newcase

After create_newcase, we have a case directory and a build/run directory

# go into scripts directory into the CESM code
```
cd $CESMROOT/cime/scripts
```

# (1) create a new case in the directory “cases” in your home directory
```
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```

# go into the case you just created in the last step
```
cd ~/cases/case01/
```

# (2) setup your case
```
./case.setup
```

# (3) build the executable
```
./case.build
```

# (4) submit your run
```
./case.submit
```
Overview of CESM directories after case.setup

# go into scripts directory into the CESM code
```
cd $CESMROOT/cime/scripts
```

# (1) create a new case in the directory “cases” in your home directory
```
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```

# go into the case you just created in the last step
```
cd ~/cases/case01/
```

# (2) setup your case
```
./case.setup
```

# (3) build the executable
```
./case.build
```

# (4) submit your run
```
./case.submit
```

---

**CASE Directory**

- `~/cases/case01`
- (2) `case.setup`
- (3) `case.build`
- (4) `case.submit`
Overview of CESM directories after `case.build`

1. Go into scripts directory into the CESM code
   ```bash
cd $CESMROOT/cime/scripts
```
2. Create a new case in the directory “cases” in your home directory
   ```bash
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```
3. Go into the case you just created in the last step
   ```bash
cd ~/cases/case01
```
4. Setup your case
   ```bash
./case.setup
```
5. Build the executable
   ```bash
./case.build
```
6. Submit your run
   ```bash
./case.submit
```
Overview of CESM directories after case.submit

### CESM Code
- **$CESMROOT**
  - components
  - cime
    - scripts
      - (1) create_newcase
- cam
- cice
- cism
- clm
- mosart
- pop
- rtm
- ww3

### CASE Directory
- ~/cases/case01
  - (2) case.setup
  - (3) case.build
  - (4) case.submit

### Build/Run Directory
- ~/scratch/case01
  - run
  - $RUNDIR
  - bld
    - $EXEROOT

### INPUTDATA Directory
- $CESMDATAROOT/inputdata
  - share
  - cpl
  - atm
  - Ind
  - ocn
  - ice
  - glc
  - wav
  - rof

Run the model

# go into scripts directory into the CESM code
```
cd $CESMROOT/cime/scripts
```

# (1) create a new case in the directory “cases” in your home directory
```
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```

# go into the case you just created in the last step
```
cd ~/cases/case01/
```

# (2) setup your case
```
./case.setup
```

# (3) build the executable
```
./case.build
```

# (4) submit your run
```
./case.submit
```
How do I know it is running?

To check whether your job is running

```bash
qstat -a
```

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Memory</th>
<th>Time</th>
<th>S</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>124.master.aws-tutoria</td>
<td>channay</td>
<td>regular</td>
<td>case01.run</td>
<td>--</td>
<td>4</td>
<td>144</td>
<td>--</td>
<td>01:00:00</td>
<td>Q</td>
<td>--</td>
</tr>
<tr>
<td>125.master.aws-tutoria</td>
<td>channay</td>
<td>regular</td>
<td>case01.st_archiv</td>
<td>--</td>
<td>1</td>
<td>1</td>
<td>--</td>
<td>00:20:00</td>
<td>H</td>
<td>--</td>
</tr>
</tbody>
</table>

The job is in the queue

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Memory</th>
<th>Time</th>
<th>S</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>124.master.aws-tutoria</td>
<td>channay</td>
<td>regular</td>
<td>case01.run</td>
<td>0</td>
<td>4</td>
<td>144</td>
<td>--</td>
<td>01:00:00</td>
<td>R</td>
<td>00:00:23</td>
</tr>
<tr>
<td>125.master.aws-tutoria</td>
<td>channay</td>
<td>regular</td>
<td>case01.st_archiv</td>
<td>--</td>
<td>1</td>
<td>1</td>
<td>--</td>
<td>00:20:00</td>
<td>H</td>
<td>--</td>
</tr>
</tbody>
</table>

The job is running

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Memory</th>
<th>Time</th>
<th>S</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>124.master.aws-tutoria</td>
<td>channay</td>
<td>regular</td>
<td>case01.run</td>
<td>0</td>
<td>4</td>
<td>144</td>
<td>--</td>
<td>01:00:00</td>
<td>C</td>
<td>--</td>
</tr>
<tr>
<td>125.master.aws-tutoria</td>
<td>channay</td>
<td>regular</td>
<td>case01.st_archiv</td>
<td>--</td>
<td>1</td>
<td>1</td>
<td>--</td>
<td>00:20:00</td>
<td>C</td>
<td>--</td>
</tr>
</tbody>
</table>

The job is completed
Overview of CESM directories after the run completes

**CASE Directory**
- `~/cases/case01`
- (2) case.setup
- (3) case.build
- (4) case.submit

**Build/Run Directory**
- `~/scratch/case01`
- `bld`
- $EXEROOT
- `run`
- $RUNDIR

**Archive Directory**
- `~/scratch/archive/case01`
- `atm`
- `...`
- `ice`
- `Ind`
- `ocn`
- `rest`

**INPUTDATA Directory**
- $CESMDATAROOT/inputdata
- `share`
- `cpl`
- `atm`
- `Ind`
- `ocn`
- `ice`
- `glc`
- `wav`
- `rof`

**CESM Code**
- $CESMROOT
- components
- cime
- scripts (1) create_newcase
  - cam
  - cice
  - cism
  - clm
  - mosart
  - pop
  - rtm
  - ww3

After the run completes, history files are moved from run directory to archive directory.

# go into scripts directory into the CESM code
```bash
cd $CESMROOT/cime/scripts
```

# (1) create a new case in the directory “cases” in your home directory
```bash
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```

# go into the case you just created in the last step
```bash
cd ~/cases/case01/
```

# (2) setup your case
```bash
./case.setup
```

# (3) build the executable
```bash
./case.build
```

# (4) submit your run
```bash
./case.submit
```
Exercise 1

Use the 4 CESM commands to create a new case. Do the case setup, build and submit your job. Use qstat to check your job is running. Check that your run completes. Check which files are in the archive directory.

```
# go into scripts directory into the CESM code
cd $CESMROOT/cime/scripts

# (1) create a new case in the directory “cases” in your home directory
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37

# go into the case you just created in the last step
cd ~/cases/case01/

# (2) setup your case
./case.setup

# (3) build the executable
./case.build

# (4) submit your run
./case.submit
```
More info about CESM component sets

CESM = plug and play

Color code:
- active
- data
- stub

Diagram:
- CESM components: atm, cpl, ocn, Ind, Sea, Land, Ice, River
- Active (green), Data (blue), Stub (purple)

Legend:
- B_ (top left)
- G_ (top right)
- I_ (bottom left)
- F_ (bottom right)
More info about CESM grids

```
./create_newcase --case ~/cases/case01 --compset B1850-tutorial --res f19_g37
```

Which resolution?

res specifies the model resolution (or grid)

The default resolution is **f09_g17**

Each model resolution can be specified by its alias or long name.

Example of equivalent alias and long name:
- alias: **f09_g17** (atm/lnd_ocn/ice)
- long name = `a%0.9x1.25_l% 0.9x1.25 _oi%gx1v7_r%r05_g%gland4_w%ww3a_m%gx1v7`

| atm | Ind | ocn/ice | river | Ind-ice | wave | ocn/ice mask |
CESM Workflow: Extend a run

In this section, we will learn how to resubmit a run and how to change the run length.
Overview of CESM directories and the xml files

The case directory contains xml files. env_* .xml contains variables used by scripts. Some can be changed by the user.

- env_case.xml: set by create_newcase and cannot be modified
- env_mach_pes.xml: specifies layout of components
- env_build.xml: specifies build information
- env_run.xml: sets run time information (such as length of run, number of submissions, …).

User interacts with this file (env_run.xml) most frequently
Some useful variables in the xml files

The length of the run is set in env_run.xml file with the variables STOP_N and STOP_OPTION.

By default, the model is set to run for 5 days:

- STOP_N = 5
- STOP_OPTION = ndays

You can query the values of these variables with the command xmlquery:

```
./xmlquery STOP_OPTION,STOP_N
```

If you want to increase the run length from 5 days to 1 month, you can use:

```
./xmlchange STOP_OPTION=nmonths,STOP_N=1
```

Other useful variables in the env_run.xml are:

- CONTINUE_RUN
- RESUBMIT

More information about the variables in env_run.xml:
http://www.cesm.ucar.edu/models/cesm2/settings/current/drv_input.html
Overview of CESM directories and the restart files

If you want to extend a run, the latest restart files are in your run directory.

case01.cam.h1.0001-01-01-00000.nc  
case01.cam.r.0001-01-06-00000.nc  
case01.cam.rh0.0001-01-06-00000.nc  
case01.cam.rs.0001-01-06-00000.nc  
case01.cice.r.0001-01-06-00000.nc  
...  
case01.pop.ro.0001-01-06-00000  
case01.ww3.r.0001-01-06-00000  

rpointer.atm  
rpointer.drv  
rpointer.glc  
rpointer.ice  
rpointer.lnd  
rpointer.ocn.restart  
...  
rpointer.rof

You have all the restart files are in the archive directory.
Exercise 2: Extend a run

Restart your run case01 and extend it for one month.

You can look at the documentation about the env_run.xml variables:
STOP_N, STOP_OPTION, RESUBMIT, CONTINUE_RUN
http://www.cesm.ucar.edu/models/cesm2/settings/current/drv_input.html

# go into the case directory
```
cd ~/cases/case01/
```

# change the length of the run
```
./xmlchange STOP_N=1,STOP_OPTION=nmonths
./xmlchange CONTINUE_RUN=TRUE
```

# submit your run
```
./case.submit
```

When you extend a run, there is no need to setup and build
Output files, log files and timing files
Overview of CESM directories

Log files and timing files

Flags:

- Runs in RUNDIR
- Moves to CASE/logs when run completes

Logs files:
- in RUNDIR while it is running
- moved to CASE/logs when run completes

Timing files in CASE/timing when run completes
Exercise 3: Estimate the run length and number of submissions based on your timing stats

Consider the model throughput * for case01. What values of STOP_OPTION, STOP_N and RESUBMIT should you use to do a 30-year run assuming you have a 12 hour maximum wallclock time * per submission.

Model Throughput = number of model year per day
Maximum wallclock time = maximum amount of time allowed to perform a run
Consider the model throughput * for case01. What values of STOP_OPTION, STOP_N and RESUBMIT should you use to do a 30-year run assuming you have a 12 hour maximum wallclock time * per submission.

```bash
# Look at the timing files. Find the model throughput
cd ~/cases/case01/timing

Overall Metrics:
  Model Throughput: 11.79 simulated years/day

## Estimate STOP_OPTION, STOP_N and RESUBMIT

We can run 10 model years/day

The maximum wallclocktime is 12 hour => In one submission we can run 5 years

To run 30 years, we need to set:
  STOP_N = 5
  STOP_OPTION = nyears
  RESUBMIT = 5
```

<table>
<thead>
<tr>
<th># years</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>First submission</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 resubmissions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Overview of CESM directories
History files, log files, timing files

Build/Run Directory
- bld
- run:
  - $RUNDIR/case01.*

CASE Directory
- ~/cases/case01
  - (2) case.setup
  - (3) case.build
  - (4) case.submit
- logs
- timing

INPUTDATA Directory
- $CESMDATAROOT/inputdata
  - share
  - cpl
  - atm
  - lnd
  - ocn
  - ice
  - glc
  - wav
  - rof

CASE Directory
- atm
- lnd
- ocn
- ice
- glc
- wav

Archive Directory
- ~/scratch/archive/case01

History files:
- in RUNDIR while it is running
- moved to archive when run completes

History files, log files, timing files

CESM Code
- $CESMROOT
  - components
    - cime
      - scripts
        - (1) create_newcase
    - cam
    - cice
    - cism
    - clm
    - mosart
    - pop
    - rtm
    - ww3

CASE Directory
- ~/cases/case01
  - (2) case.setup
  - (3) case.build
  - (4) case.submit
- logs
- timing

Build/Run Directory
- ~/scratch/case01
- ~/scratch/archive/case01

Archive Directory
- ~/scratch/archive/case01
- atm
- lnd
- ocn
- rest

History files:
- in RUNDIR while it is running
- moved to archive when run completes
Exercise 4: Look at model output

When exercise 2 completes, look at the monthly history file

# go into the case directory
cd ~/scratch/archive/case01/atm/hist

# You should see a file# named case01.cam.h0.0001-01.nc
ls

# Look at the file content
ncdump -h case01.cam.h0.0001-01.nc

# Visualize at the file content
ncview case01.cam.h0.0001-01.nc
Namelist Modifications
CESM namelists

Each component has a namelist. You can look at your namelists in the run directory: *_.in

The namelists contain a set of variables used at runtime

A few examples of namelist variables
- co2vmr
- dust_emis_fact
- bnd_topo
Overview of CESM directories + namelist files

CASE Directory

~cases/case01
(2) case.setup
(3) case.build
(4) case.submit
user_nl_cam
user_nl_cice
user_nl_cism
user_nl_clm
user_nl_cpl
user_nl_mosart
user_nl_pop
user_nl_ww

CaseDocs
atm_in
cism_in
drv_flds_in
drv_in
ice_in
Ind_in
mosart_in
pop_in
wav_in

SourceMods
...
Overview of CESM directories
+ 4 CESM commands

**CESM Code**

$CESMROOT

- components
- cime
- scripts
  - (1) create_newcase

**CASE Directory**

~/$cases/case01
(2) case.setup
(3) case.build
(4) case.submit

user_nl_cam
user_nl_cice
user_nl_cism
user_nl_clm
user_nl_cpl
user_nl_mosart
user_nl_pop
user_nl_ww

**SourceMods**

**CaseDocs**

atm_in
cism_in
drv_flds_in
drv_in
ice_in
Ind_in
mosart_in
pop_in
wav_in

**Build/Run Directory**

/blad~/scratch/case01

---

**INPUTDATAROOT Directory**

$CESMDATAROOT/inputdata

share
cpl
atm
Ind
ocn
ice
glc
wav
rof

The script
preview_namelists
is called by
case.build and
case.submit

**When you do an namelist modification in user_nl_*
Run the command ./preview_namelist
It creates the files *_in in CaseDocs and run...**
Exercise 5: Double the CO2 concentration

Create a new case B1850_2x for the B1850_compset at f19_g37. Double the CO2 concentration. Check that the value of co2vmr changed in CaseDocs/atm_in and in the run directory. You don’t need to build (.case.build) and submit (.case.submit)

```bash
# go into scripts directory into the CESM code
cd $CESMROOT/cime/scripts

# create a new case in the directory “cases” in your home directory
./create_newcase --case ~/cases/B1850_2x --compset B1850-tutorial --res f19_g37

# go into the case you just created in the last step
cd ~/cases/B1850_2x

# setup your case
./case.setup

# Edit user_nl_cam. Add the line:
co2vmr=569.4e-6

# preview namelists
./preview_namelists

# Check the value of co2vmr in
~/cases/B1850_2x/CaseDocs/atm_in

# Check the value of co2vmr in
~/scratch/B1850_2x/run/atm_in
```

The script preview_namelists is called by case.build and case.submit
Where to find info about namelists?

http://www.cesm.ucar.edu/models/cesm2.0/

In “Prognostic Components” or in “Components Configuration Settings”, you can find information about namelist variables in “Component Fortran Namelist settings”
Where to find info about namelists?

http://www.cesm.ucar.edu/models/cesm2.0/

Namelist definitions for every component
Where to find info about namelists?

http://www.cesm.ucar.edu/models/cesm2.0/

Browse variables names
Show details about variables
Customizing CAM history files

In this section, we will cover:

• how to change the output frequency
• how to output extra variables
• how to output extra history files
• how to control the number of time samples written to a history file

This can be achieved with 3 namelist variables:

- \textit{nhtfrq}: sets the output frequency
- \textit{fincl}: add variables to the history file
- \textit{mfilt}: maximum number of time samples written to a history file
Let’s change the output frequency in CAM

By default, CESM outputs *monthly average* history files but you can output at other frequency.

For instance: to change the output frequency of a CAM history file from *monthly average* to *daily average*, we use the namelist variable: $nhtfrq=-24$
Customizing CAM history files: nhtfrq, mfilt

The default history file from CAM is a monthly average.

We can change the output frequency with the namelist variable \textit{nhtfrq}.
If \textit{nhtfrq} = 0, the file will be a monthly average.
If \textit{nhtfrq} > 0, frequency is input as number of timesteps.
If \textit{nhtfrq} < 0, frequency is input as number of hours.

For instance to change the history file from monthly average to daily average, we set the namelist variable:
\textit{nhtfrq} = -24

To control the number of timesteps in the history file, we can use the variable \textit{mfilt}.
For instance, to specify that we want one time sample on each history file, we set the namelist variable:
\textit{mfilt} = 1
Customizing CAM history files: fincl

You can output up to 10 history files: “h0”, “h1”, …, “h9”.

The file “h0” contains the default variables (in the code: “call add_default”). This includes the variables necessary for the AMWG package.

For the files “h1” to “h9”, the user has to specify the variables to output.

To control the list of fields in the history files, we can use the namelist variables

\[ h0 \quad h1 \quad \ldots \quad h9 \quad fincl1 \quad fincl2 \quad \ldots \quad fincl10 \]

For instance, the line:

\[ fincl1 = ‘PRECT’ \]

is used to add the field ‘PRECT’ to the file “h0”
Customizing CAM history files: fincl

Using a ":" following a field gives the **averaging flag** for the output field. Valid flags are:
- I for instantaneous,
- A for average,
- M for minimum,
- X for maximum.

For instance, the line:

```
fincl1 = ‘PRECT:M’
```

is used to add the minimum of ‘PREC’ to the file “h0”
Exercise 6: Add history files and change output frequency

Create a new case called “b1850_high_freq” with the B1850-tutorial compset at f19_g37.
Add “h1” file containing daily averages of TREFHT and PRECT
Add “h2” file containing instantaneous values of PSL and U10 every 6 hours.
Run 5 days.

# go into scripts directory into the CESM code
cd $CESMROOT/cime/scripts

# create a new case in the directory “cases” in your home directory
./create_newcase --case ~/cases/B1850_high_freq --compset B1850-tutorial --res f19_g37

# go into the case you just created in the last step
cd ~/cases/B1850_high_freq

# setup your case
./case.setup

# Edit user_nl_cam. Add the lines:
fincl2='TREFHT','PRECT'
fincl3='PSL','U10'
nhtfrq=0,-24,-6
mfilt =1,5,4

# build
./case.build

# submit
./case.submit
Troubleshooting

In this section, we make the model crash and we look for the info in the log files
Overview of CESM directories with log files

- **CESM Code**
  - $CESMROOT
    - components
      - cam
      - cice
      - cism
      - clm
      - mosart
      - pop
      - rtm
      - ww3
    - scripts
      - (1) create_newcase
  - cime

- **CASE Directory**
  - ~/cases/case01
    - (2) case.setup
    - (3) case.build
    - (4) case.submit

- **Build/Run Directory**
  - ~/scratch/case01

- **Archive Directory**
  - ~/scratch/archive/case01

- **INPUTDATA Directory**
  - $CESMDATAROOT/inputdata
    - share
    - cpl
    - atm
    - Ind
    - ocn
    - ice
    - glc
    - wav
    - rof

- **Logs files:**
  - in RUNDIR while it is running
  - moved to CASE/logs when run completes

If the run crashes, examine log files in the run directory $RUNDIR
Exercise 7: Troubleshooting with the log files

Create a new run called “b1850_high_freq_bugfixing”. Use the B1850-tutorial compset at f19_g37. Output daily averages values of the values of the variable T2M in the output file h1. What happens?

```bash
# go into scripts directory into the CESM code
cd $CESMROOT/cime/scripts

# create a new case in the directory “cases” in your home directory
./create_newcase --case ~/cases/b1850_high_freq_bugfixing --compset B1850-tutorial --res f19_g37

# go into the case you just created in the last step
cd ~/cases/b1850_high_freq_bugfixing

# setup your case
./case.setup

# Edit user_nl_cam. Add the lines:
fincl2='T2M'
nhtfrq=0,-24

# build
./case.build

# submit
./case.submit
```
Code Modification

In this section, we will learn how to do simple code modifications such as adding a new variable.
Your choice: The Red Pill or the Blue Pill

The Matrix (1999): Neo, the main character is offered the choice between a red pill and a blue pill.

- The blue pill would allow him to remain in the Matrix (a fictional computer-generated world)

- The red pill would lead to his "escape" from the Matrix into the real world and embracing the sometimes painful truth of reality.

Courtesy: Andrew Gettelman
Principles for modifying the code

View of the CESM directories. Where to modify the code?
Principles for modifying the code

CESM Code

$CESMROOT

components

cime

scripts

(1) create_newcase

 cam
 cice
 cism
 clm
 mosart
 pop
 rtm
 ww3

CASE Directory

~/cases/case01
(2) case.setup
(3) case.build
(4) case.submit

Build/Run Directory

/glade~/scratch/case01

bld $SEXEROOT

run $RUNDIR

src.cam
src.cice
src.clm
src.pop2
...

These modifications only affect the current case
This is where you will put your modifications

INPUTDATA Directory

$CESMDATAROOT/inputdata

share cpl atm Ind ocn ice glc wav rof

These modifications affect all the cases built out of this SRCROOT.
Principles for modifying the code

CESM Code

$CESMROOT

components

cime

scripts
(1) create_newcase

cam

cice

cism

clm

mosart

pop

rtm

ww3

CASE Directory

~/cases/case01
(2) case.setup
(3) case.build
(4) case.submit

Buildconf

LockedFiles

SourceMods

Tools

CaseDocs

Build/Run Directory

/glade~/.scratch/case01

bld
$EXEROOT

run
$RUNDIR

# go into scripts directory
.cd $CESMROOT/cime/scripts

# (1) create a new case
./create_newcase --case ~/cases/case01 --compset
B1850-tutorial --res f19_g37

# go into the case you just created
.cd ~/cases/case01/

# (2) setup your case
./case.setup

# (3) build the executable
./case.build

# (4) submit your run
./case.submit

Make your source mods

These modifications affect all the cases built out of this SRCROOT.

These modifications only affect the current case

This is where you will put your modifications
Modifying a subroutine

Steps to modify the code:

- Find the subroutine you want to modify
- Copy this subroutine in SourceMods
- Make your mods
- Compile and run the model
Output an extra variable

• One common thing you may want to do is to add code to output a new variable

• For instance, CAM has a field to output the temperature at 500 mbar (T500) but not at 750 mb.
Let’s add a field to output the temperature at 750 mbar (T750)

This can be done by a succession of calls:

\[
\text{call addfld(’T750’, …)} \quad \rightarrow \quad \text{Add a field to master field list}
\]
\[
\text{call outfld(’T750’, …)} \quad \rightarrow \quad \text{Collect values for this field and write to history file}
\]
Exercise 8: Add an additional output variable

Create a new case called “b1850_T750” using the B1850-tutorial compset at f19_g37 resolution. Add the variable T750

```bash
# go into scripts directory into the CESM code
cd $CESMROOT/cime/scripts

# Create a new case
./create_newcase --case ~/cases/b1850_T750 --compset B1850-tutorial --res f19_g37

# Case setup
cd ~/cases/b1850_T750
./case.setup

# Locate the file where T500 is computed and copy it SourceMods/src.cam
cp $CESMROOT/components/cam/src/physics/cam/cam_diagnostics.F90 SourceMods/src.cam

# Edit the file SourceMods/src.cam/cam_diagnostics.F90 and add the lines:
!++ add a variable for T750
    call addfld('T750', horiz_only, 'A', 'K','Temperature at 750 mbar pressure surface')

!++ add a variable for T750
    if (hist_fld_active('T750')) then
        call vertinterp(ncol, pcols, pver, state%pmid, 75000._r8, state%t, p_surf, &
        extrapolate='T', ps=state%ps, phis=state%phis)
        call outfld('T750 ', p_surf, pcols, lchnk )
    end if
```
Exercise 8: Add an additional output variable

Create a new case called “b1850_T750” using the B1850-tutorial compset at f19_g37 resolution. Add the variable T750. Output daily values of T500 and T750 in the h1 file

```plaintext
## Add to user_nl_cam
fincl2='T500', 'T750'
nhtfrq=0,-24

## Build and submit
./case.build
./case.submit

## Check your solution. When the run is completed, check the field T750 and T500 are in the file h1

cd /glade/scratch/$user/archive/b1850_T750/
ncdump -h b1850_T750.cam.h1.0001-01-01-00000.nc

float T500(time, lat, lon);
   T500:units = "K";
   T500:long_name = "Temperature at 500 mbar pressure surface";
   T500:cell_methods = "time: mean";
float T750(time, lat, lon);
   T750:units = "K";
   T750:long_name = "Temperature at 750 mbar pressure surface";
   T750:cell_methods = "time: mean";
```
Start up, Hybrid and Branch runs

In this section, we look at the run type
Run types: Startup, Branch, Hybrid

**STARTUP: CONTROL**
- Extend a run by setting “CONTINUE_RUN = TRUE”
- Nothing can change (no namelist change, no source mods).
- Bit-for-bit with control

**CONTINUE_RUN**
- Initialized from restart files of previous CESM run.
- Can add extra history files and variables.
- No source mods or namelist change (except extra output)
- Cannot change start date.
- Can retain the same case name (or not)
- Bit-for-bit with control

**BRANCH**
- Initialized from restart files of a previous CESM run.
- Can add namelist changes and source modification.
- Can use a different start date.
- Not bit-for-bit with control

**HYBRID**
- Initialized from restart files of a previous CESM run.
- Can add namelist changes and source modification.
- Can use a different start date.
- Not bit-for-bit with control

---

Community Earth System Model Tutorial
Run types: Startup, Branch, Hybrid

- **STARTUP: CONTROL**
  - RUN_TYPE = STARTUP
  - CONTINUE_RUN = TRUE

- **STARTUP** → **CONTINUE_RUN**
  - RUN_TYPE = STARTUP
  - CONTINUE_RUN = TRUE

- **BRANCH**
  - RUN_TYPE = BRANCH
  - RUN_REFCASE
  - RUN_REFDATE

- **HYBRID**
  - RUN_TYPE = HYBRID
  - RUN_REFCASE
  - RUN_REFDATE
  - RUN_STARTDATE

**CESM** often uses the hybrid run type, to start from a spunup state.
Exercise 9: Hybrid run
Initializing a run from a previous run

Create a case “b1850_hybrid” using the B1850-tutorial compset at f19_g37 resolution. Set the run as an hybrid run from the run: b1850_initial
The initial run is in: /home/isimpson/scratch/archive/b1850_initial

```bash
# go into scripts directory into the CESM code
# create a new case
cd $CESMROOT/cime/scripts
./create_newcase --case ~/cases/b1850_hybrid --compset B1850-tutorial --res f19_g37

# go in case directory and case setup
cd ~/cases/b1850_hybrid
./case.setup

# Set an hybrid run
./xmlchange RUN_REFCASE=B1850-initial
./xmlchange RUN_REFDATE=0002-01-01
./xmlchange RUN_STARTDATE=0001-01-01

# Copy initial conditions in run directory
cp /home/isimpson/scratch/archive/b1850_initial/rest/0002-01-01-00000/* ~/scratch/b1850_hybrid/run

# Add to user_nl_cam
fincl2='PRECT'
nhtfrq=0,-24
mfilt=1,1

# Build and submit
./case.build
./case.submit
```
The million things we didn’t cover

In this section:
- Tutorial website
- How to download the code
- How to port the code
- Where to find more
Tutorial website

http://www.cesm.ucar.edu/events/tutorials/2018/

A one-week tutorial can be find here

Coursework

Consider to apply next year...
Tutorial website

http://www.cesm.ucar.edu/events/tutorials/2018/

A one-week tutorial can be find here
Download cesm2.0.0

git clone –b release-cesm2.0.0 https://github.com/ESCOMP/cesm.git cesm2.0.0
cd cesm2.0.0
./manage_externals/checkout_externals

More info at


About porting:

See porting session on the tutorial website:
http://www.cesm.ucar.edu/events/tutorials/2018/
Where to find help?

http://www.cesm.ucar.edu/models/cesm2.0/

CESM webpage is a gold mine for model documentation

If you cannot find an answer in the model documentation, post your question on the CESM Bulletin Board