Quick start
Installation and run guide for
CESM 1.0.5
On Snellius

Michael Kliphuis

Introduction

This informal document contains a quick start guide on how to install and run CESM version 1.0.5 on the Dutch national supercomputer Snellius. More information about this CESM version can be found at: https://www2.cesm.ucar.edu/models/cesm1.0/

Installation

Here comes the installation procedure:

1. Log in on Snellius

2. If you do not already have a directory ~/models/cesm create it by typing:

   ```
   cd
   mkdir –p models/cesm
   ```

3. Download the cesm1.0.5 code by typing:

   ```
   cd ~/models/cesm
   ```

   ```
   svn co https://svn-ccsm-release.cgd.ucar.edu/model_versions/cesm1_0_5 cesm1_0_5
   ```

   Unfortunately this link does not work anymore. It is unclear where it was moved to at the ucar.edu site so for now simply copy the code from the shared scratch space of my mkliphuis login by typing:

   ```
   cp /scratch-shared/mkliphuis/cesm/cesm1_0_5 .
   ```

In the $HOME space we now have all the model code and scripts we need to set up a CESM run.

4. Next create directories in the project space, where we can store the bigger files.
   We need one for the input and one for the output files:

   (Modify all project space occurences **uus20475** below into your own project space!)

```
cd /projects/0/uus20475
mkdir –p  cesm/cesm1_0_5/inputdata
mkdir –p  cesm/cesm1_0_5/outputdata
```

5. Download the inputdata for CESM. As in 3. it is not possible to download the data from the ucar.edu website. As a workaround copy the data from the shared scratch space of my mkliphuis login by typing:

```
cd /projects/0/uus20475/cesm1_0_5/inputdata
cp –r /scratch–shared/mkliphuis/cesm/inputdata/*  .
```

   Next modify some configuration files that enable CESM to run on Snellius.

6. The first file is the config_machines.xml

```
cd ~/models/cesm/cesm1_0_5/scripts/ccsm_utils/Machines
```

   Make sure that the file **config_machines.xml** has an entry:

(Again modify all project space occurences uus20475 below into your own project space!)

```
<machine MACH="snellius"
      DESC="Surf Lenovo cluster, OS is Centos8 Linux, we use Genoa nodes
      with 192 pes/node, batch system is SLURM"
      EXEROOT="/projects/0/uus20475/cesm/cesm1_0_5/outputdata/$CASE"
      OBJROOT="$EXEROOT"
      LIBROOT="$EXEROOT/lib"
      INCROOT="$EXEROOT/lib/include"
      DIN_LOC_ROOT_CSMDATA="/projects/0/uus20475/cesm/cesm1_0_5/inputdata"
      DIN_LOC_ROOT_CLMQIAN="/not/used"
      DOUT_S_ROOT="/not/used"
      DOUT_L_HTAR="FALSE"
      DOUT_L_MSROOT="not/used"
      CCSM_BASELINE="not/used"
      CCSM_CPRNC="/not/used"
      ESMF_LIBDIR="/not/used"
      OS="Linux"
      BATCHQUERY="squeue"
      BATCHSUBMIT="sbatch &lt;"
      GMAKE_J="24"
      MAX_TASKS_PER_NODE="192"
      MPISERIAL_SUPPORT="TRUE"
      PES_PER_NODE="192" />
```

7. Set other machine specific values like modules and paths to the NetCDF libraries by modifying file **env_mach_specific.snellius**

   ```
   cd ~/models/cesm/cesm1_0_5/scripts/ccsm_utils/Machines
   ```

   If env_mach_specific.snellius does not exist yet create it by typing:

   ```
   cp env_machopts.generic_linux_pgi  env_mach_specific.snellius
   ```

   We need to make sure that the model finds the proper NetCDF modules. In this case on Snellius we load the 2024 modules. The file **env_mach_specific.snellius** should start with lines:

   ```
   # set modules
   source /etc/profile.d/modules.csh
   module purge
   module load 2024
   module load foss/2024a
   module load netCDF-Fortran/4.6.1-gompi-2024a
   module load netCDF/4.9.2-gompi-2024a

   # internal CESM libraries like pio make use of
   # variable NETCDF_PATH and are implemented such that
   # they expect that .inc and .mod files
   # of the NETCDF_C as well as the NETCDF_Fortran library
   # are in a directory $(NETCDF_PATH)/include
   # They also expect that the NETCDF_C library libnetcdf.a
   # as well als NETCDF_Fortran library libnetcdff.a are in
   # a directory $(NETCDF_PATH)/lib
   # Make sure the model finds everything by creating your
   # own directory and put all the files there with a 'ln'

   setenv MY_NETCDF $HOME/models/cesm/cesm1_0_5/scripts/my_netcdf

   rm -rf $MY_NETCDF

   mkdir -p $MY_NETCDF/include
   ln -sf $EBROOTNETCDF/include/* $MY_NETCDF/include/
   ln -sf $EBROOTNETCDFMINFORTRAN/include/* $MY_NETCDF/include/

   mkdir -p $MY_NETCDF/lib
   ln -sf $EBROOTNETCDF/lib/* $MY_NETCDF/lib/
   ln -sf $EBROOTNETCDFMINFORTRAN/lib/* $MY_NETCDF/lib/
   ```

   Also other machine specific environment variables can be set here.
   This might be needed to set MPI, core file, IO, memory etc.

8. Set compiler and flags by modifying file **Macros.snellius**

   If Macros.snellius does not exist yet create it by typing:

   ```
   cp Macros.generic_linux_pgi  Macros.snellius
   ```

We want to use the 2024 modules and the foss/2024a toolchain (see 7.)
Therefore make sure that Macros.snellius file contains the lines (around line 143):

```
ifeq ($(USE_MPISERIAL),TRUE)
  FC := gfortran
  C  := gcc
else
  FC := mpif90
  CC := mpicc
endif
```

Make sure that the correct NetCDF libraries are used by setting the lines (around line 162):

```
NETCDF_PATH = $(HOME)/models/cesm/cesm1_0_5/scripts/my_netcdf/
INC_NETCDF    := $(NETCDF_PATH)/include
LIB_NETCDF    := $(NETCDF_PATH)/lib
MOD_NETCDF    := $(NETCDF_PATH)/include
```

Next set the compiler flags by setting lines (around line 184):

```
CFLAGS        := $(CPPDEFS)
FIXEDFLAGS    :=
FREEFLAGS     := -FR
FFLAGS        := $(CPPDEFS) -O2 -fconvert=big-endian -fallow-invalid-boz -
                  ffree-line-length-none -fallow-argument-mismatch -fno-
                  unsafe-math-optimizations -frounding-math -fsignaling-
                  nans -fbacktrace -DISNAN_INTRINSIC
FFLAGS_NOOPT  := $(FFLAGS) -O0
FFLAGS_OPT    := -O2
LDFLAGS       :=
AR            := ar
MOD_SUFFIX    := mod
CONFIG_SHELL  :=
```

Next make sure there are lines (around line 230):

```
ifeq ($(compile_threaded), true)
  FFLAGS       += -fopenmp
  FFLAGS_NOOPT += -fopenmp
  CFLAGS       += -fopenmp
  LDFLAGS      += -fopenmp
endif
```

And finally some lines to make sure that the internal libraries mct and pio are compiled correctly (around line 260):

```
ifeq ($(MODEL),mct)
   #add arguments for mct configure here
   CONFIG_ARGS += CC="$(CC)" CFLAGS="$(CFLAGS)" FC="$(FC)"
   FCFLAGS="$(FFLAGS)" F90="$(FC)" F90FLAGS="$(FFLAGS)" MPIFC="$(FC)"
   INCLUDEPATH="-I$(INC_MPI)"
endif
```

```
ifeq ($(MODEL),pio)
  ifneq ($(strip $(PIO_CONFIG_OPTS)),)
    CONFIG_ARGS += $(PIO_CONFIG_OPTS)
  endif
  CONFIG_ARGS += FC="$(FC) $(FFLAGS)" FCFLAGS="$(FFLAGS)
              -DFORTRANUNDERSCORE" MPIF90="$(FC) $(FFLAGS)
              -DFORTRANUNDERSCORE" CC="$(CC)" MPICC="$(CC)"
              MPI_INC="-I$(INC_MPI)" NETCDF_PATH="$(NETCDF_PATH)"
endif
```

9. Set the batch system values by modifying file **mkbatch.snellius**

   If mkbatch.snellius does not exist yet create it with:

   ```
   cp mkbatch.generic_linux_pgi  mkbatch.snellius
   ```

   Then make sure that mkbatch.snellius starts with lines:

   ```
   #! /bin/csh -f

   set mach = snellius
   ```

   We want to use the genoa nodes on Snellius so make sure that the part between EOF1 is:

   ```
   cat >! $CASEROOT/${CASE}.${mach}.run << EOF1
   #!/bin/tcsh -f
   #===================================================================
   # GENERIC_USER
   # This is where the batch submission is set.  The above code computes
   # the total number of tasks, nodes, and other things that can be useful
   # here.  Use PBS, BSUB, or whatever the local environment supports.
   #===================================================================

   # Loadleveler directives start with # @
   #
   # In this job an MPI program is started.
   #
   #SBATCH --time=00:30:00
   #SBATCH -p genoa
   #SBATCH -n ${ntasks}
   #SBATCH --job-name=${jobname}

   #limit coredumpsize 1000000
   #limit stacksize unlimited

   EOF1
   ```

Also make sure that the part between the second EOF1 is:

```
cat >> ${CASEROOT}/${CASE}.${MACH}.run << EOF1
# ----------------------------------------------------------------------
# Run the model
# ----------------------------------------------------------------------

sleep 25
cd \$RUNDIR
echo "\`date\` -- CSM EXECUTION BEGINS HERE"

#=============================================================
# GENERIC_USER
# Launch the job here.  Some samples are commented out below
#=============================================================
srun ./ccsm.exe >& ccsm.log.\$LID

wait
echo "\`date\` -- CSM EXECUTION HAS FINISHED"

EOF1
```

10. Next to NetCDF-C functions the model also needs NetCDF-Fortran functions.
    Make sure the Linker finds them while compiling everything into an executable.
    You can make sure this will work by typing:

    ```
    cd  $HOME/models/cesm/cesm1_0_5/scripts/ccsm_utils/Build
    ```

    Then open file Makefile and modify:

    ```
    ifeq ($(strip $(SLIBS)),)
       SLIBS := -L$(LIB_NETCDF) -lnetcdf
    else
       SLIBS += -L$(LIB_NETCDF) -lnetcdf
    endif
    ```

    into:

    ```
    ifeq ($(strip $(SLIBS)),)
       SLIBS := -L$(LIB_NETCDF) -lnetcdf -lnetcdff
    else
       SLIBS += -L$(LIB_NETCDF) -lnetcdf -lnetcdff
    endif
    ```

OK that's it! Now the model has been installed and set up such that it can be run on Snellius!

Set up a run case

You can create a default Pre-Industrial (1850) climate run case with **1** x **p**re-**i**ndustrial greenhouse gase **c**oncentrations (1pic) on a 1° ocean/ice grid and a 2° atmosphere/land grid (also known as resolution f19g16) as follows:

1. First go to the directory from where we will set up a case:

   ```
   cd $HOME/models/cesm/cesm1_0_5/scripts
   ```

2. Then type:

./create_newcase -case b.PI_1pic_f19g16 -compset B_1850 -res 1.9x2.5_gx1v6 -mach snellius

3. Now make sure it runs on 768 Genoa cores by typing:

   ```
   cd b.PI_1pic_f19g16

   cp env_mach_pes.xml env_mach_pes.xml_orig

   cp ../env_mach_pes_768_cores_genoa.xml env_mach_pes.xml
   ```

   If you want to run on a different number of cores then modify env_mach_pes.xml
   You can ask me (Michael Kliphuis) for guidance. The way these 768 cores are balanced over the components ocean, atmosphere, land, sea-ice and coupler is very efficient though.

   ```
   ./configure –case
   ```

4. Next we need to modify fortran file shr_sys_mod.F90.

   In this code there are calls to system() and chdir(). These functions are first declared as external integer and this gives an error when compiling with GNU.
   By simply uncommenting the declarations we overcome this problem.

   A quick way to implement this is to type:

   ```
   cd $HOME/models/cesm/cesm1_0_5/scripts/b.PI_1pic_f19g16
   cp ../shr_sys_mod.F90 SourceMods/src.share/
   ```

5. Now build the executable ccsm.exe

   ```
   ./b.PI_1pic_f19g16.snellius.clean_build

   ./b.PI_1pic_f19g16.snellius.build
   ```

   This will take about 7-8 minutes and the executable will end up in directory:

   ```
   /projects/0/uus20475/cesm/cesm1_0_5/outputdata/b.PI_1pic_f19g16/run
   ```

Start the run case

Now start the run case as follows:

1. `cd $HOME/models/cesm/cesm1_0_5/scripts/b.PI_1pic_f19g16`

2. In our low resolution (f19g16) runs we typically let the model output a so called history file every model month containing the monthly mean averages for Temperature, Humidity, Winds, Salinity etc. We also typically let the model output a restart file every model month. You tell the model to do this by opening file env_run.xml and set:

   ```
   <entry id="AVGHIST_OPTION"   value="nmonths"  />
   <entry id="AVGHIST_N"   value="1"  />

   <entry id="REST_OPTION"   value="nmonths"  />
   <entry id="REST_N"   value="1"  />
   ```

3. As a test case let us run the model for 1 model month, do this by opening env_run.xml again and set:

   ```
   <entry id="STOP_OPTION"   value="nmonths"  />
   <entry id="STOP_N"   value="1"  />
   ```

4. Make sure that you reserve enough wallclock time for the job by opening file: b.PI_1pic_f19g16.snellius.run and set:

   ```
   #SBATCH --time=00:30:00
   ```

   Or longer, if needed!

5. Finally start the run by typing:

   ```
   sbatch b.PI_1pic_f19g16.snellius.run
   ```

6. The job is then submitted to the queue, you can check if it is running by typing:

   squeue
   you will see something like:

   ```
   JOBID     PARTITION   NAME    USER     ST    TIME  NODES NODELIST(REASON)
    8619141  genoa       b.PI_1pi mkliphui R     0:05    4       tcn[607,625,633,645]
   ```

   When it runs ST has value R.

7. If you want to kill the job then type:

   ```
   scancel 8619141
   ```

8. The output will end up in the directory:

   ```
   /projects/0/uus20475/cesm/cesm1_0_5/outputdata/b.PI_1pic_f19g16/run
   ```

9. You can do a quick check on the CESM 1.0.5. output data with the application 'ncview'

   Remember that on Snellius, if you type:

   ```
   module spider ncview
   ```

   You can check the versions of the ncview.
   After that you find out that you need to type

   ```
   module load 2023 ncview/2.1.8-gompi-2023a
   ```

   in order to be able to use the application

   It is best to check the output of the ocean.

   First type:

   ```
   cd /projects/0/uus20475/cesm/cesm1_0_5/outputdata/b.PI_1pic_f19g16/run
   ```

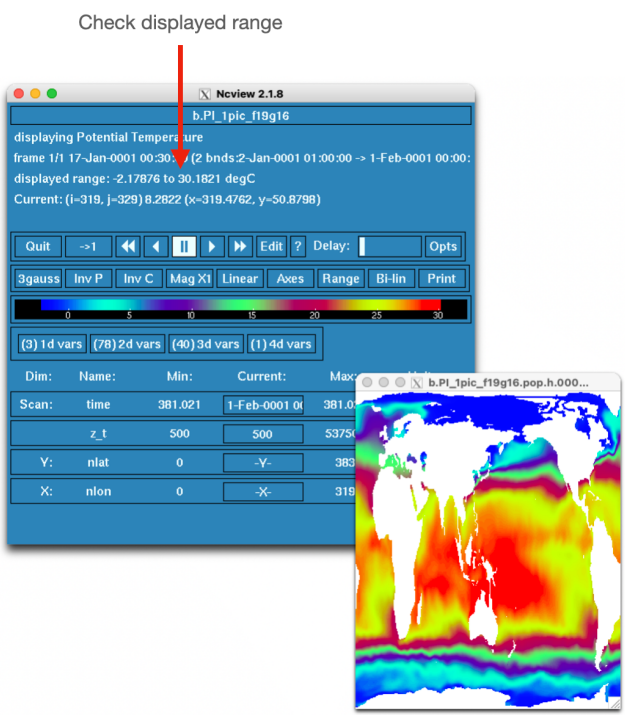
   Then type:

   ```
   ncview  b.PI_1pic_f19g16.pop.h.0001-01.nc
   ```

   It suffices to check the 3d vars TEMP and SALT which stand for the temperature and salinity of the ocean.

   When you select the TEMP variable you will see the temperature values at the surface of the ocean (SST) and they should be somewhere in a range between -3° C and 40° C

   When you select the SALT variable you will see the salinity values at the surface (SSS) and they should be in the range between 2 g/kg and 45 g/kg



10. The performance of the low resolution model on 768 cores is now about 48 modelyears/24h. You can check this in the timing file in the run directory:

    `$HOME/models/cesm/cesm1_0_5/scripts/b.PI_1pic_f19g16/timing`

    You can also check the performance in the coupler log file in the output directory:

    `/projects/0/uus20475/cesm/cesm1_0_5/outputdata/b.PI_1pic_f19g16/run`

    e.g. in file (yours will have a different timestamp): `cpl.log.241120-144340`

    Such a file has lines like:

    ```
    tStamp_write: model date = 10105  0 wall clock = 2024-11-20 14:45:17 avg dt = 5.09 dt
    memory_write: model date = 10105  0 memory   =  470.78 MB (highwater) 8234.74 MB (usage)
    ```

    Meaning that generating january 5 of year 1 took 5.09 sec and needed 8234.74 MB memory
    5.09 sec per day means the model generates (1/5.09) days/sec ≈ 46 modelyears/24h.
    This was for a run of only 1 modelmonth. If you run it for a whole modelyear the performance becomes slightly better, in the order of the earlier mentioned 48 modelyears/24h.

Here follows a note for Wim Rijks of SURF:

This performance is about 33% worse than the performance we had **before** the maintenance on Snellius on September 15 this year (2024). I used to get about 72 modelyears/24h and now only 48 modelyears/24h.

At IMAU there are three people who are running CESM on Snellius, these are: me and my collegues René Wijngaard (r.r.wijngaard@uu.nl) and Casey Patrizio (c.r.patrizio@uu.nl).

Even though we use different versions of CESM we all deal with a similar performance loss. We would highly appreciate it, if it is possible to get back SBU computerhours. Especially Casey Patrizio has a problem now since his SBUs are already spent and this is much sooner than expected because of the slower performance. Can he get the SBUs back even during your investigation of the problem?

For my own research in which we checked the collapse of the Gulf Stream/AMOC I did many CESM 1.0.5 runs on 768 Genoa cores with my klipdccp login on Snellius. Most of the runs were done before September 15 (2024) with performances of about 72 modelyears/24h.

An example of such a run -which was done in May 2024- can be checked by looking at the timing file:

```
/home/klipdccp/models/cesm/cesm1_0_5/scripts/b.e10.BRCP4.5_CN.f19_g17.rcp4
.5_future_start_y2100_extended_600_branch.001/timing/
ccsm_timing.b.e10.BRCP4.5_CN.f19_g17.rcp4.5_future_start_y2100_extended_60
0_branch.001.240508-150757
```

And corresponding coupler log file:

```
/projects/0/prace_imau/prace_2013081679/cesm1_0_5/b.e10.BRCP4.5_CN.f19_g17
.rcp4.5_future_start_y2100_extended_600_branch.001/run/cpl.log.240505-
164635
```