Running Glimmer-CISM on the OLCF xk6 with several test cases
Aug 3, 2012

1. Login to OLCF Jaguarpf system:
ssh -Y $NAME@home.ccs.ornl.gov (or –X if on linux)
type in PIN+PASSCODE,

then ssh to jaguarpf:
ssh -Y $NAME@jaguarpf.ccs.ornl.gov (or –X if on linux)

type in PIN+PASSCODE,
Note: if it fails after two tries, control C out and start again or you will be locked out. ($NAME is the code that goes with token and PASSCODE is the number shown on the token)

Details on using Jaguarpf can be found here:
http://www.olcf.ornl.gov/support/user-guides-policies/jaguar-xk6-user-guide/

1. copy over run data to your lustre space:

```
cd /tmp/work/$NAME
cp –r /tmp/work/4ue/cism_tests_tutorial /tmp/work/$NAME
```

In this directory is a README file explaining the tests we will run for this tutorial,
a. dome30, a 30x30 domain with a circular shaped dome of ice.
b. gis_5km, Greenland 5km resolution test case from a cold start condition

2. create the software environment to build and run the model

```
module load python netcdf/4.1.2 p-netcdf/1.2.0;
module swap pgi pgi/12.5.0
```

3. In home space:/ccs/home/$NAME=$HOME, take a tarball of the most recent version of seacism:

```
cp /tmp/work/4ue/cism_2.0.tar .
```

(this will take several minutes)

(Typically, you would download from repo if you have access
svn co https://svn-cism-model.cgd.ucar.edu/glimmer-cism2/branches/seacism)

```
tar –xvf cism_2.0.tar
```

Untars a directory called “seacism” (this also takes several minutes)

```
cd seacism
./bootstrap
```

On jaguar, we build the model in home space:/ccs/home/$NAME)
and run model in temporary lustre space /tmp/work/$NAME/
This is because the compute nodes only see lustre space

4a. Build model to run in parallel and use trilinos to perform velocity solve

```
make distclean (if not a fresh build)
./configure-scripts/xk6-config
make
```

4b. Build model to run in serial and use a local solver for velocity (for small computers without MPI or trilinos)

```
make distclean (if not a fresh build)
./configure-scripts/xk6-config-serial
make
```

5. copy executable to the run data location in lustre space

```
cp /$HOME/seacism/example_drivers/simple_glide/src/simple_glide
/tmp/work/$NAME/cism_tests_tutorial/
```

6. submit a job to the queue to our specially allocated nodes
a. **cd /tmp/work/$NAME/cism_tests_tutorial/**
b. choose the desired case, **cd /$case**
c. using the submission script for jaguar, submit the job, **qsub ijob**

The ijob script :

The top section contains all the settings for xk6

```
#!/bin/bash                                    type of shell
#PBS -A trn001                                 project to charge
#PBS -q batch                                  type of queue (debug. Killable, etc)
#PBS -N GIS_5km_1yr                            job name (for the user )
#PBS -l gres=widow1%widow2%widow3              where does the data sit on lustre?
#PBS -j oe -l walltime=1:00:00,size=752        how low and how many nodes

#export MPICH_UNEX_BUFFER_SIZE=1000000000    sometimes needed for MPI
#export MPICH_MAX_SHORT_MESSAGE_SIZE=1000  (not ideal)
export SCRIPT=gis_5km.config                          configuration/input file
export EXE=/tmp/work/$USER/ cism_tests_tutorial/      path to executable
export BASE=/tmp/work/$USER/cism_tests_tutorial/gis_5km  path to run data
cd $PBS_O_WORKDIR           go to lustre – where we must run on compute nodes
```

These are the run commands on the xk6

```
echo JFNK build
time aprun -n750 $EXE/simple_glide $BASE/$SCRIPT  >& output/out
```

*note we asked for 752 processors but are only running with 750 because we have to ask for processors in multiples of 16, since there are 16 processors/node

The script is sent to the queueing system and is given a $job_number. For this tutorial, dedicated nodes have been allocated, so we will see these jobs start immediately. When the job is running, the timing output will go to a $job_number.OU file. We can monitor the progress of the case simulations by looking at this file. The screen output, which provides details about the run progress and solver details, goes to output files specified:

For dome: output/out.$SOLVER.#proc. This job submission script runs the model with a JFNK velocity solver on 1,2,4,9,amd 15 procs, and then the same processor layout but using the Picard solver. Then two runs of the same code in serial are performed with the serial executable.

For gis_5km, the run output details are in the output/out.gis file.

Look at the output data of our jobs (its overwritten each time in the ijob script):
Module load ncview
Cd output in the $case directory
If you get an xterm error, try logging in again with ssh using the –X (linx) or (-Y) flag

In the dome30 output directory:
ncview dome.30.out.nc
a gui should pop up in separate window. Click on the “thk” variable (thickness), then can click the FF button to watch the dome spread over the 10 time step simulation. Click on other variables to see them as well.

In the gis_5km/output directory:
ncview gis_5km.ice2sea.thin.out.nc
Use the same ncview commands as above, except this time there is only one time step.