OASIS3-MCT tutorial

November 2012

I – Tutorial

The "tutorial" toy should reproduce the coupling between two simple models, "model1" and "model2", with the OASIS3-MCT coupler. The source files to be completed are found in the directory "oasis3-mct/examples/tutorial".

The total run is of 6 hours. The time step of "model1" is 3600 sec with a coupling period of 2 hours. The time step of "model2" is 1800 sec with a coupling period of 3 hours. The "model1" runs on a logically-rectangular (182x149) grid and "model2" runs on a logically-rectangular (96x72) grid.

At each coupling exchange, "model1" (see model1.F90) receives the field "FRECVOCN", e.g. its boundary conditions produced by "model2", and "model2" receives the field "FRECATM", e.g. its boundary conditions produced by "model1". At the end of the time step, "model1" sends its coupling field "FSENDOCN" to "model2" and "model2" sends its coupling field "FSENDATM" to "model1"; "model1" also outputs to a file the field "FOCNWRIT", which is the same field than "FSENDOCN".

All the documentation about the coupler OASIS3-MCT can be found on the IS-ENES web site: http://oasis.enes.org
The current OASIS3-MCT coupler uses internally the Model Coupling Toolkit (MCT) developed by the Argonne National Laboratory to perform parallel regridding and parallel exchanges of the coupling fields : http ://www.mcs.anl.gov/mct

A. Compiling and running "tutorial" with OASIS3-MCT

1. Login with login and password provided and open a terminal session clicking on the icon in the top bar.
2. To compile the OASIS3-MCT coupler:
   - Go into directory oasis3-mct/util/make_dir
   - Adapt the "make.inc" file, which includes your platform header makefile make.pgi_cerfacs.
   - Type "make realclean –f TopMakefileOasis3" and then "make –f TopMakefileOasis3"

   - ➢ The libraries "libmct.a", "libmpeu.a", "libpsmile.MPI1.a" and "libscrip.a" that need to be linked to the models are available in the directory $ARCHDIR/lib ($ARCHDIR is defined in your platform header file)
3. To compile the tutorial models:
   - Go into directory oasis3-mct/examples/tutorial
   - Type "make clean; make"
   - ➢ The executables "model1" and "model2" are available in the current directory.
4. To run the two tutorial component models:
   - Adapt to your platform and execute the script run_tutorial: "./run_tutorial"; the results of the component models are now in subdirectory:
     rundir = ${HOME}/oasis3-mct/examples/tutorial/work_tutorial.
   - ➢ In their current form, "model1.F90" and "model2.F90" are not interfaced with the OASIS3-MCT library and do not perform any exchanges. They just run for 6 timesteps of 1 hour each without doing anything specific.
5. To interface "model1.F90" and "model2.F90" with the OASIS3-MCT library and run the tutorial coupled model:
   - Modify "model1.F90" and "model2.F90" so that the models first receive their input coupling fields and then, at the end of the timestep, send their output coupling field, as detailed above. The lines where to introduce the OASIS3-MCT specific calls are marked with "”!! TOCOMPLETE …". Most of the variables that will be used in the oasis routines calls are already defined at the beginning of the programs.
     A step-by-step approach is recommended, i.e. validate one-by-one each call implemented by compiling the toy coupled model.
   - The OASIS3-MCT configuration file "namcouple", located in the directory oasis3-mct/examples/tutorial/data_oasis3, is written to perform these exchanges every 7200 sec, i.e. every two timesteps. The interpolations specified to transform "FSENDOCN" into "FRECVATM" and "FSENDATM" into "FRECVOCN" are BILINEAR. Interpolation form "model1" to "model2" uses a pre-existing weight and addresses file (my_remapping_file_bilinear.nc) while the weight and addresses file used for the interpolation from "model2" to "model1" is calculated at the first coupling time step with the SCRIP library.
   - Run the toy coupled model with the script "run_tutorial".
   - In this configuration, where both models call their oasis_get before their oasis_put, a deadlock will occur in the coupled toy model. Find which modifications are needed in the toy to remove this deadlock without changing the order of the oasis_get and oasis_put (see section 2.10.1 of User Guide for more details).
   - ➢ The results of the tutorial coupled model are now in subdirectory "${HOME}/oasis3-mct/examples/tutorial/work_tutorial". In "work_tutorial", the file "nout.000000", written by one of the master process of the models, contains the information read in the configuration file "namcouple" by all the processes. The OASIS3-MCT debug files debug.root.01 and debug.root.02 are written by the processor of each model. The level of debug information in these files depends on the value of NLOGPRT defined in the namcouple. Here NLOGPRT=10 and the files contain normal diagnostics production, with initial debug info and calling trees of the routines (see the section 3.2 of the User Guide for more details).
   - ➢ You can visualize the results with ferret and the scripts script_ferret_*.jnl. Type "ferret" and then "go xxx.jnl" where xxx.jnl is the name of the ferret script you want to run. (Note that to run another ferret script, you have to restart ferret.)

6. Explain why the number of time steps in the file FSENDATM_model2_02.nc is not the same than in the file FRECVOCN_model1_03.nc.
   - ➢ Keep your results by renaming /work_tutorial into /work_tutorial_A_mono

In this configuration, "model1" and "model2" run with one process each. This is indicated in the launching script "run_tutorial" (see "nproc_exe1" and "nproc_exe2").

B. <u>Running the models in parallel with OASIS3-MCT</u>

To run "model1" and "model2" in parallel, the number of processes has to be modified in the launching procedure.

- Specify for example "nproc_exe1=3", "nproc_exe2=3" in the script run_tutorial .
- In oasis3-mct/examples/tutorial, execute "run_tutorial", which then launches the models on 3 processes each.
- Keep your results by renaming /work_tutorial into /work_tutorial_B_para
- ➢ Visually compare the results with the non-parallel case A.

OASIS3-MCT supports different parallel decompositions for the models (see the section 2.4 of the User Guide). Tutorial routine "decomp_def.F90" can define the local partition for the BOX or the APPLE decompositions. By default, the APPLE decomposition is used. To test the BOX decomposition, recompile the tutorial models with, in Makefile: "CPPKEYDECOMP_M1=DECOMP_BOX" or
"CPPKEYDECOMP_M2=DECOMP_BOX".

<u>**II – Test_interpolation**</u>

This toy tests the quality of the interpolation between a source grid and a target grid by calculating the error of interpolation on the target grid. There is only one time step and the coupling is performed at t=0. At t=0, "model1" sends its coupling field "FSENDANA" to "model2".
The two models write output files. The amount of informations in these files depends on the value of the local variable "FILE_Debug", which equals 0, 1 or 2, in the same way as does NLOGPRT (see the section 3.2 of the User Guide for more details). By default "FILE_Debug=1".

A. <u>Quality of the conservative interpolation from "ORCA2T" to "LMDz"</u>

If you look into the configuration file "data_oasis3/namcouple" and in the namelist file "data_oasis3/name_grids.dat", you can see that the toy performs a conservative interpolation between "ORCA2T" and "LMDz" using the predefined remapping file myfile_torc_to_lmdz_CONSERV_FRACNNEI.nc, calculating the interpolated field on the source grid via the "src" option (see the section 4.3 of the User Guide for more details). This will be detailed in paragraph C.

1. Compiling and running test_interpolation
    - Go to the directory "oasis3-mct/examples/test_interpolation"
    - Compile the toy using : "make clean ; make"
    - Adapt to your platform and execute the script run_test_interpolation: "./run_test_interpolation". The results of the component models are now in subdirectory: rundir = ${HOME}/oasis3-mct/examples/test_interpolation/work_test_interpolation.
2. Analysis of the results
    - To evaluate the quality of the interpolation between the source grid, "ORCA2T", and the target grid, "LMDz", an analytical field (in file "$rundir/FSENDANA_model1_01.nc") is exchanged between the source grid and the target grid and the error of interpolation is calculated. The error is defined as the absolute value of the difference between the interpolated field (in file "$rundir/FRECVANA_model2_01.nc") and the analytical field calculated on the target grid, divided by the interpolated field. The error is usually multiply by 100 to have it in percent (in file "$rundir/error.nc").
    - ➢ You can visualize the results with ferret and the scripts script_ferret_*.jnl. Type "ferret" and then "go xxx.jnl" where xxx.jnl is the name of the ferret script you want to run. (Note that to run another ferret script, you have to restart ferret.). Adapt the scale in the file script_ferret_error.jnl to the min and max of the error you will find in the output file "$rundir/model2.out_101". Why is the error more important at the North pole and near the coasts ?
    - ➢ Keep your results by renaming /work_test_interpolation into /work_test_interpolation_A_mono

B. Quality of other interpolations

In the tests below, the SCRIP library is used to calculate the weights and addresses remapping file at the first time step t=0.

1. Bilinear interpolation between "ORCA2T" and "LMDz" using the SCRIP library
    - Modify the configuration file "data_oasis3/namcouple" to perform a bilinear interpolation using the SCRIP library (see the section 4.3 of the User Guide for more details).
    - Run again the toy test_interpolation executing the script "./run_test_interpolation".
    - Keep your results by renaming /work_test_interpolation into /work_test_interpolation_B1_mono. You can see that a new remapping file has been created rmp_torc_to_lmdz_BILINEAR.nc.
    - ➢ Visually compare the results with the case A.
2. Bilinear interpolation from "LMDz" to "ORCA2T" using the SCRIP library
    - Exchange the names of the source grid and the target grid in the namelist file "data_oasis3/name_grids.dat" and put the correct grid's dimensions and periodicity in the "data_oasis3/namcouple".
    - Run again the toy test_interpolation executing the script "./run_test_interpolation".
    - Keep your results by renaming /work_test_interpolation into /work_test_interpolation_B2_mono.
    - ➢ Analyse the error of interpolation (adapt the scale in the file script_ferret_error.jnl to better visualise the error if necessary).

C. Time statistics: comparison of time using src or dst options of the "MAPPING" transformation

It is possible to get time performance informations with OASIS3-MCT thanks to the variable "TIMER_Debug" defined in the module "mod_oasis_timer.F90" (see the section 6.3.2 of the User Guide for more details).

1. Modification of the sources of "OASIS3-MCT":
   - Go to the directory $(HOME)/oasis3-mct/lib/psmile/src. In "mod_oasis_timer.F90" put "TIMER_Debug=1". With this value, time statistics will be calculated over all the processors of each model and each master processor will write a timer file, "model1.timers_0000" and "model2.timers_0000", containing the time statistic results.
   - Go back to the directory "oasis3-mct/examples/test_interpolation".
   - Compile the toy and the libraries using : "make clean ; make"
2. Time performance for the "MAPPING" transformation using the "src" option **(**see the section 4.3 of the User Guide):
   - Modify the configuration file "data_oasis3/namcouple" to test the performance of the bilinear interpolation from "LMDz" to "ORCA2T" with the pre-existing remapping file rmp_lmdz_to_torc_BILINEAR.nc, created by the SCRIP library in the section (B-2.), using the "src" option. In this case, the interpolated field is calculated on the source grid, from the source field neighbours and the weights, and then sent to the target.
   - Keep your results in /work_test_interpolation_C1_mono .
   - ➢ Look into the files "model1.timers_0000" and "model2.timers_0000". All the informations given in these files are explained in the section 6.3.2 of the User Guide.
   - Specify for example "nproc_exe1=2", "nproc_exe2=4" in the script run_test_interpolation to run the toy in parallel. Execute "./run_test_interpolation".
   - Keep your results in /work_test_interpolation_C1_para .
   - ➢ Compare the performances obtained in the monoprocessor and the parallel case.
3. Time performance for the "MAPPING" transformation using the "dst" option:
   - Test again the "MAPPING" transformation between "LMDz" and "ORCA2T" with the remapping file rmp_lmdz_to_torc_BILINEAR.nc in monoprocessor and in parallel (putting "nproc_exe1=2", "nproc_exe2=4" in "run_test_interpolation") using the "dst" option. In this case, the interpolated field is calculated on the target grid, from the source field neighbours and the weights.
   - Keep your results respectively in /work_test_interpolation_C2_mono, /work_test_interpolation_C2_para.
   - ➢ Compare the performances obtained in the monoprocessor and the parallel case.
   - ➢ Compare the time statistics to the results obtained with the "src" option. Why is it more efficient to do the calculation on the target grid in this case ?

D. Other optimization options for the "MAPPING" transformation : bfb, sum, opt

By default, the "bfb" option is activated (see the section 4.3 of the User Guide for more details). It forces the mapping to be done in a bit for bit manner and the results are independent of the domain decomposition. The "sum" option forces the mapping to be done using partial sums. The source values belonging to one particular process and used in the calculation of one target point

may be locally partially summed before the global sum, involving possible other source values from other processes, is done. The partial sums will result in non bit reproducible results between simulations done with different number of processes. Finally, the "opt" option allows the code to choose either "bfb" or "sum" based upon some logic about which might be faster.
If the models run on one processor the results is independent of the option "bfb", "sum" or "opt".

1. "MAPPING" transformation using the "sum" option :
   - Modify the configuration file data_oasis3/namcouple in order to perform a bilinear interpolation from "LMDz" to "ORCA2T" with the pre-existing remapping file rmp_lmdz_to_torc_BILINEAR.nc using the ("src" + "sum") options.
   - Specify "nproc_exe1=2", "nproc_exe2=4" in the script run_test_interpolation to run the toy in parallel. Execute "./run_test_interpolation".
   - Keep your results in /work_test_interpolation_D1_para
   - ➢ Compare the interpolated field, the error and the time statistics results with the case C1_para (corresponding to the case "src" + "bfb" options). Which strategy is the more performant ?
2. "MAPPING" transformation using the "opt" option :
   - Modify the configuration file "data_oasis3/namcouple" in order to perform a bilinear interpolation from "LMDz" to "ORCA2T" with the pre-existing remapping file rmp_lmdz_to_torc_BILINEAR.nc using the ("src" + "opt") options.
   - Keep "nproc_exe1=2", "nproc_exe2=4" in the script run_test_interpolation to run the toy in parallel. Execute "./run_test_interpolation".
   - Keep your results in /work_test_interpolation_D2_para
   - ➢ Compare the interpolated field, the error and the time statistics results with the case C1_para and D1_para. Which strategy was chosen in this case ("sum" or "bfb") ?

<u>E. Test of the quality of the bilinear interpolation between the participant's grids.</u>

The NCL program "create_aux_files.ncl" located in oasis3-mct/examples/test_interpolation creates files grids.nc, masks.nc and areas.nc from the two files "data_oasis3/grid_model1.nc" and "data_oasis3/grid_model2.nc" containing respectively the ORCA2T and the LMDz grids. The aim is to adapt it to construct the files with the grids of the participants and then test the quality of interpolation between these grids.
All the documentation about NCL can be found on the NCAR web site: http://www.ncl.ucar.edu/

1. Creation of the grids.nc, masks.nc and areas.nc files, needed to calculated the weights and addresses mapping files, with the grid definitions of the participant's models:
   - If you do not already have these files, modify the name of the NetCDF files "f1" and "f2" containing the longitude, latitude, corners, mask and areas of each grid of your model you should have brought with you.
   - Modify the name of the grids, masks and areas, "torc" and "lmdz", with the name of your grids. They should have 4 characters. For simplicity "grd1" and "grd2" will design the generic names of the grids of the participants.
   - Run the ncl program using "ncl create_aux_files.ncl". NCL does not manage dots (.), needed by OASIS3-MCT to recognize the name of the variables, all the variables are defined with an "_" in in the files grids.nc, masks.nc and areas.nc obtained by NCL.

- To obtain the final files grids.nc, masks.nc and areas.nc needed by OASIS3-MCT to calculate the remapping weights files use the following NCO commands for i=1 and i=2:
  - ➢ ncrename -v grdi_lon,grdi.lon grids.nc
  - ➢ ncrename -v grdi_lat,grdi.lat grids.nc
  - ➢ ncrename -v grdi_clo,grdi.clo grids.nc
  - ➢ ncrename -v grdi_cla,grdi.cla grids.nc
  - ➢ ncrename -v grdi_msk,grdi.msk masks.nc
  - ➢ ncrename -v grdi_srf,grdi.srf areas.nc

2. Quality of the bilinear interpolation between "grd1" and "grd2" of the participants:
   - Put the name of your source "grd1" and target "grd2" grids in the namelist data file "data_oasis3/name_grids.dat" and change the dimensions of the grids in the configuration file "data_oasis3/namcouple".
   - Modify the configuration file "data_oasis3/namcouple" in order to perform a bilinear interpolation from the source "grid1" to the target "grid2" using the SCRIP library.
   - Specify "nproc_exe1=1", "nproc_exe2=1" in the script run_test_interpolation. Execute "./run_test_interpolation".
   - Keep your results in /work_test_interpolation_E1_mono
   - ➢ Analyse the quality of the interpolation by plotting the interpolated field and the error of interpolation.
3. Quality of the bilinear interpolation between "grd2" and "grd1" of the participants:
   - Modify the name of your source and target grid in the namelist data file "data_oasis3/name_grids.dat" and change the dimensions of the grids in the configuration file "data_oasis3/namcouple".
   - Execute "./run_test_interpolation".
   - Keep your results in /work_test_interpolation_E2_mono
   - ➢ Analyse the quality of the interpolation by plotting the interpolated field and the error of interpolation.

III – Compilation and running of Oasis3-mct and Tutorial on the remote computer of the participants

Before interfacing the real models with OASIS3-MCT, it is necessary to verify that the coupler and the toy tutorial compile and run on the remote computers of the participants.

1. Send OASIS3-MCT to the participant's remote computer:
   - Tar and zip the repository oasis3-mct:
   - ➢ cd $(HOME) ; "tar -cvf oasis3-mct.tar oasis3-mct" ; "gzip oasis3-mct.tar"
   - Send the file to your remote computer using "ssh":
   - ➢ scp oasis3-mct.tar $user@remote_computer:/remote_directory
2. Compile the OASIS3-MCT coupler:
   - Untar and uzip oasis3-mct.tar.gz
   - ➢ "gunzip oasis3-mct.tar.gz" ; "tar -xvf oasis3-mct.tar"

- Go into directory "oasis3-mct/util/make_dir". Create a header makefile make.myplatform to compile OASIS3-MCT on your machine. Adapt also make.inc.
- Type "make realclean –f TopMakefileOasis3" and then "make –f TopMakefileOasis3"

3. Compile and run tutorial :
    - Go into "oasis3-mct/examples/tutorial"
    - Type "make clean" and then "make" to compile the toy.
    - Adapt the script "run_tutorial" to run on your remote platform. Execute "./run_tutorial".
    - Keep your results in /work_tutorial_C

IV - Interfacing of OASIS3-MCT in participant's real models

The models and OASIS3-MCT must be compiled with the same compilers on the same platform.

V - Compilation and running of the coupled models