OASIS3-MCT training

December 2014

All the documentation about the coupler OASIS3-MCT can be found on the OASIS web site at http://oasis.enes.org and in the OASIS3-MCT sources in the oasis3-mct/doc directory.
It is also available on the USB key that was given to you.
The current OASIS3-MCT coupler uses internally the Model Coupling Toolkit (MCT) developed by the Argonne National Laboratory (http ://www.mcs.anl.gov/mct) to perform parallel regridding and parallel exchanges of the coupling fields.

I – Creation of the configuration file of OASIS3-MCT (namcouple) for the tutorial using the OASIS GUI

The aim of this first part is to create the *namcouple* text file used for the ***tutorial*** toy coupled model you will have to interface with OASIS3-MCT in part II. The *namcouple* contains all the user-defined information necessary ***to configure a particular coupled run***.

A. Extracting the OASIS3-MCT sources

1. Login with login and password provided and open a terminal session clicking Applications -> System Tools -> Terminal in the top bar. Download the OASIS3-MCT sources of the official version 2.0 :
   - Open a browser window and go to the OASIS web site to register and download the sources, at the address: https://verc.enes.org/oasis/download/oasis-registration-form
   - If you download directly the oasis3-mct.tar.gz file from the web site, just gunzip it and untar it
   - If you use the command svn checkout from the web site, create a directory (mkdir oasis3-mct), go into this directory (cd oasis3-mct) and apply the command svn checkout

B. OASIS3-MCT GUI

The OASIS3-MCT GUI is an application of OPENTEA, a graphical interface used by different codes, and developed at CERFACS by the Combustion team. It is located in the repository oasis3-mct/util/oasisgui. A short description of the OASIS GUI is given in the ***README file*** in oasis3-mct/util/oasisgui.

1. Create a .cshrc file in your $HOME. Open it and define the following PATH and alias "oasisgui" to launch easily the GUI, using the commands:
   - setenv OASISGUIHOME $HOME/oasis3-mct/util/oasisgui
   - setenv PYTHONPATH $OASISGUIHOME/XDRpy
   - alias oasisgui 'wish $OASISGUIHOME/opentea/opentea.tcl -config $OASISGUIHOME/myconfig.xml -code oasis3-mct &'

Update your environment using :
   - source .cshrc

Launch the GUI using the alias you just created:
   - oasisgui

You must fill up the different tabs from the left to the right. Each window must be validated (button "process" on the bottom on the right in each window). An orange bullet in the tab means that no information was stored yet in the window, a red bullet in the tab means that there is an error in the window and a green bullet in the tab means that the data entered in the window is correct.

Validated data will be progressively stored in an xml file when filling the GUI. This file must be explicitly saved when leaving the GUI (you will be automatically asked for this). Once you will have saved your xml data, you will be able to reload it using the (File/Load as new) menu at the left top of the GUI (***do not use the (File/Load as part) menu***).

You can find information on OASIS3-MCT by clicking on the blue bullets (?) "Learn more" in each window.

C. Tutorial example: the coupling algorithm

To create the configuration file for the tutorial example, you must understand the coupling algorithm that this toy should reproduce. This is explained below.

The "tutorial" toy model should reproduce the coupling between two simple codes, model1 and model2, with the OASIS3-MCT coupler.

The total run is 6 hours (21600 seconds). Model1 has a time step of 3600 seconds and runs on the logically-rectangular ORCA2T grid (182x149). Model2 has a time step of 1800 seconds and runs on the logically-rectangular LMDz grid (96x72).

In the coupling to implement, a field with **symbolic name** “FSENDOCN” in model1 and “FRECVATM” in model2 is sent from model1 to model2 every 2 hours and a field with **symbolic name** “FSENDATM” in model2 and “FRECVOCN” in model1 is sent from model2 to model1 every 3 hours as follows:

- At the beginning of the run, model1 receives “FRECVOCN” coming from a restart file ; at the end of the coupling period of 2 hours, model1 sends the coupling field

"FSENDOCN" to model2 that receives it as "FRECVATM" at the beginning of the next coupling period (see **figure fsendocn** below).

- In addition, model1 also outputs to a file the **time averaged field** "FSENDOCN" every 2 hours (see the **figure fsendocn_file**).
- At the beginning of the run, model2 receives "FRECVATM" coming from a restart file ; at the end of the coupling period of 3 hours, model2 sends the coupling field "FSENDATM" to model1 that receives it as "FRECVOCN" at the beginning of the next coupling period (see the **figure fsendatm** below).
- The interpolation from "FSENDOCN" into "FRECVATM" uses a pre-existing weight and addresses file called "my_remapping_file_bilinear.nc". The interpolation specified to transform "FSENDATM" into "FRECVOCN" is the SCRIPR/BILINEAR one (i.e. the weight and addresses file used for the interpolation are calculated at the first coupling time step with the SCRIP library).

Figure fsendocn

LAG=3600 for FSENDOCN dt=3600s, CPL=7200s, maxtime=21600s

The field is sent when (t+LAG=n x CPL)

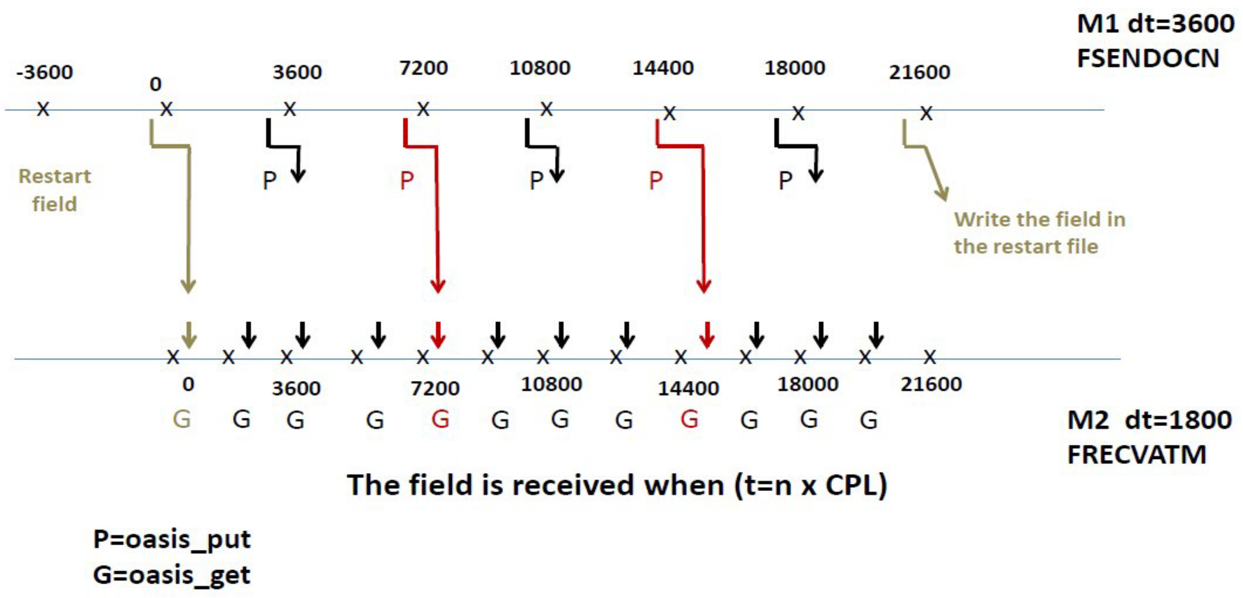
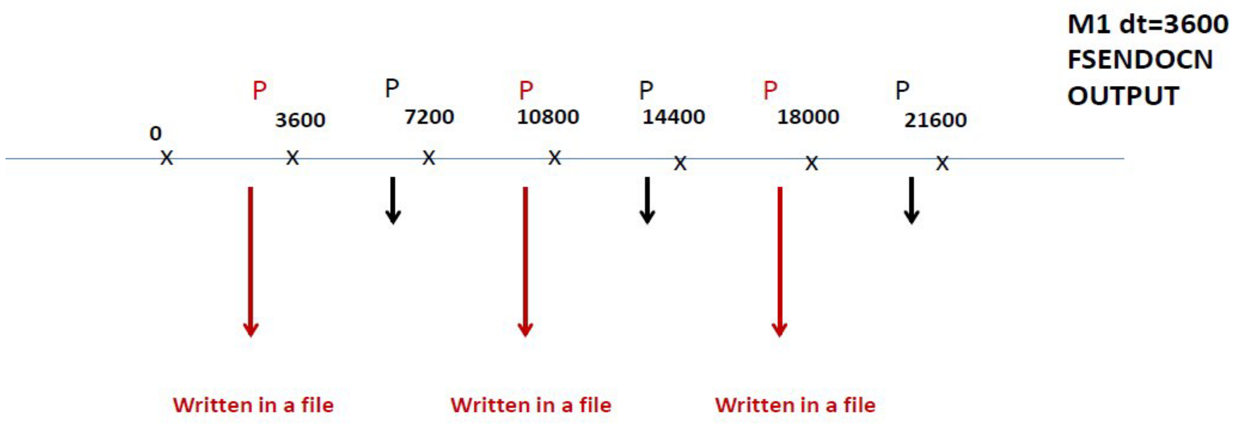
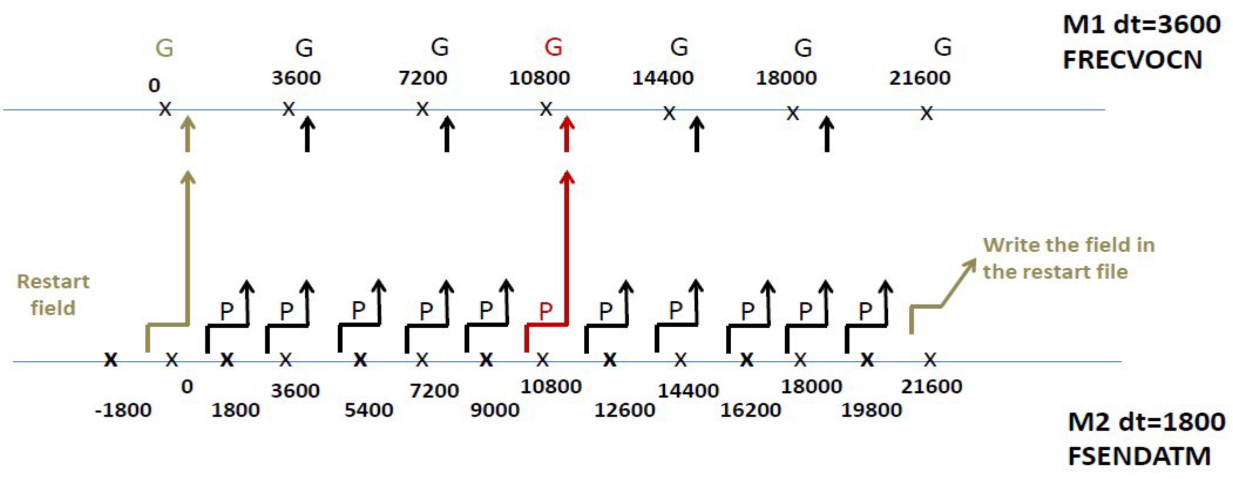


Figure fsendocn_file
FSENDOCN sent to a file CPL=7200s, maxtime=21600s
The field is sent when (t=n x CPL)



P=oasis_put

Figure fsendatm
LAG=1800 for FSENDATM dt=1800s, CPL=10800s, maxtime=21600s
The field is received when (t=n x CPL)



The field is sent when (t+LAG=n x CPL)

P=oasis_put
G=oasis_get

D. Creation of the namcouple file using the OASIS GUI

Please read carefully the following instructions to construct the namcouple:

1. Read the description of the GUI in the first window and validate it by processing it using the "Process" button on the bottom right.
2. Window "Gen" (Generalities):
   - Enter the total simulation length corresponding to the total run of the tutorial of 6 hours.
   - Put the debug level at 1, which corresponds to debug information written by OASIS3-MCT only by the master of each model (see the blue bullet for more details) in files "debug.root.01" and "debug.root.02".
   - Put the Time statistics option to 0 so to not perform any time statistics for the tutorial toy.
   - Do not activate the "Enable grouped fields" as there will be no fields sent together in the tutorial toy.
   - Validate the window by processing it using the "Process" button.
3. Window "Model":
   - Enter the names of the two models that will be coupled using the button "add Model", double clicking on the new row created and filling the part "no label" under "Model names". The names of the model in the namcouple are the ones that will be used when interfacing them with OASIS3-MCT using oasis_init_comp. In your case please defined them as "model1" and "model2".
   - For each model you can also specify the maximum Fortran unit number used by the models but it is optional.
   - When processing the window, you are asked to save the data in an xml file. Save the new xml file in the data directory of the tutorial toy:
     oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial.xml.
     The namcouple text file will be saved, when all the windows will be filled up and processed, in a sub-directory named after the xml file in directory :
     oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial
4. Window "Grids":
   - Enter the names of the grids associated to the coupling fields of the different models, clicking on the button "add Grid in use", double clicking on the new row created and filling the part "no label" under "Grid names". In your case, the grids defined in the models are **torc** for model1 and **lmdz** for model2.
   - For each grid you must specify if the grid is global (periodical) or regional and the number of overlapping points. ORCA2T (torc, 182x149) is periodical with 2 overlapping points and LMDz (lmdz, 96x72) is also periodical but without any overlapping point.
   - You may define the dimensions of each grid or not, but if they are specified for one grid, they must be specified for all grids. It is easier afterwards to visualize the results with Ferret if you put the dimensions.
5. Window "Fld name" (Field name):
   - For each coupling field , a "user reference name" must be given, clicking on the button "add Coupling field", double clicking on the new row created and filling the part "no label" under "Field label". This "user reference name", as the GUI Id generated automatically in the second column, is only used by the GUI and will not appear in the namcouple.
     Then for each coupling field, the symbolic name used in the source and target models must be specified as well as the source and target grids. These symbolic names in the namcouple are the ones that will be used when interfacing with OASIS3-MCT using oasis_def_var.
     To define the coupling fields, please refer to the paragraph C.

- When processing the window to validate the data, an automatic "GUI Id" is associated to each field by the GUI. This Id will be used in the last tabs to defined the different transformations associated to each field. If the coupling fields are not grouped the Id is the Field label.

6. From window "Fld status" to window "Cons" (for Conservation for global transformations):
   - Define all characteristics and transformations associated to each coupling field identified by its Id under the tab Fields in each window. You have some documentation available by clicking on the blue (?) "Learn more".
     - ✓ The name of the restart file for the field **FSENDOCN** sent by model1 already exists and is named **fdocn.nc**, the name of the restart file for the field **FSENDATM** sent by model2 also already exists and is named **fdatm.nc**. The field **FSENDOCN** will additionally be time averaged and written to a file; the name of a restart file (not used initially but used between two runs), for example **f2avg.nc**, must be also defined for this field. Defined the status so to be able to visualize and debug the results. For the field FSENDOCN written to a file, the status must be **OUTPUT**. In this case the output file and the debug file are the same.
     - ✓ As each model reads a restart file, define the corresponding LAG for each field (see section 2.10.1 of User Guide for more details).
     - ✓ To be able verify quickly the results and the remapping, we suggest you to activate the CHECKIN and CHECKOUT options for each field (*the results will appear below the attribute diags in the debug files*).
     - ✓ The interpolation specified to transform "FSENDATM" into "FRECVOCN" should be the **SCRIPR/BILINEAR** one (i.e. the weight and addresses file used for the interpolation are calculated at the first coupling time step with the SCRIP library) under the SCRIP tab. See the User Guide section 4.3 for more details on the BILINEAR interpolation.
     - ✓ The interpolation from "FSENDOCN" into "FRECVATM" should use a pre-existing weight and addresses file called "my_remapping_file_bilinear.nc", to be specified under the MAP tab.
     - ✓ Do not forget to process each window one after the other.
7. Window "Sum" (for Summary):
   - Process the window to obtain a summary of all the coupling fields and their transformations.
8. Window "Config" (for Configuration file):
   - ➢ Process the window to obtain the namcouple; the textfile will be saved in the sub-directory named after the corresponding xml file, e.g. ***oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial,*** if, as suggested above in D3, you saved the xml file in oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial.xml. Copy the namcouple file from ***/data_oasis3/namcouple_tutorial*** to ***/data_oasis3*** **.**

If you close and reopen the oasisgui application, upload the xml file located in ***oasis3-mct/examples/tutorial/data_oasis3/ namcouple_tutorial.xml*** using the (File/Load as new) menu.

II – Tutorial

A. Compiling OASIS3-MCT and running an empty "tutorial" toy model

1. To compile the OASIS3-MCT coupler:
   - Go into directory oasis3-mct/util/make_dir
   - Adapt the value of $ARCHDIR in the platform header makefile **make.pgi_cerfacs** to have the compiled sources in e.g. $(HOME)/oasis3-mct_comp (or another local directory)
   - Adapt the "make.inc" file to include 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

2. To compile the tutorial models:
   - Go into directory oasis3-mct/examples/tutorial
   - Type "make clean; make" (note that the Makefile in this directory automatically includes your OASIS3-MCT header makefile – see the first line in the Makefile)
   - ➢ The executables model1 and model2 are available in the current directory.

3. To run the two tutorial component models:
   - Edit the script run_tutorial to adapt it to your platform ("training_computer") and execute (see "4. Execute the model" in run_tutorial for how the models are launched):
     > ./run_tutorial
     The results of the component models are now in subdirectory $rundir defined in run_tutorial (i.e. ${HOME}/oasis3-mct/examples/tutorial/work_tutorial by default).
   - ➢ 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 time steps of 1 hour each without doing anything specific (see the files "model1.out_100" and "model2.out_100" in the output directory work_tutorial)

B. Interfacing and running the "tutorial" toy model with OASIS3-MCT

1. Modify "model1.F90" and "model2.F90" so to implement the initialisation, definition and declaration calls of OASIS3-MCT library. The lines where to introduce the OASIS3-MCT specific calls are marked with "”!! TOCOMPLETE …". As a first step, suppose that each model will run on only one process. ***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 and checking that the toy model runs fine until the call.

2. Modify "model1.F90" and "model2.F90" so to implement the coupling exchanges, as described in the paragraph C. In the time step of the models, try first by implementing the reception of the input coupling fields ("oasis_get") at the beginning of the time step and then, the sending of the output coupling field ("oasis_put"), at the end of the time step (see again **figures fsendocn and fsendatm**).

3. Run the toy coupled model with the script "run_tutorial" (after mofifying the part "4. Execute the model"). In this configuration, where both models call their oasis_get before their oasis_put, a deadlock may occur in the coupled toy model if the coupling fields received at the beginning of the run are not automatically read from the coupling restart files. If this happens, 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 the master process of one model, contains the information read in the configuration file "namcouple". The OASIS3-MCT debug files debug.root.01 and debug.root.02 are written by the master process of each model. The level of debug information in these files (which corresponds to the debug level defined in the GUI in the second window "Gen") depends on the value of the first number on the line below $NLOGPRT in the namcouple (see the section 3.2 of the User Guide for more details).
- ➢ If you have put “EXPOUT” for the coupling field status, you can visualize the content of the netCDF debug files 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.)
- ➢ 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_B3_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").

C. Running the models in parallel with OASIS3-MCT

To run and couple model1 and model2 in parallel, one has to ensure that the partition definition transferred to OASIS3-MCT via the call to oasis_def_partition is properly done. In model1 and model2, the information to provide as arguments of the oasis_def_partition is calculated by the subroutine decomp_def that you can check in more detail. Then, 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_C_para
- ➢ Visually compare the results with the non-parallel case B3_mono.

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 (make clean ; make) the tutorial models with, in Makefile: "CPPKEYDECOMP_M1=DECOMP_BOX" or
"CPPKEYDECOMP_M2=DECOMP_BOX".

III – Test_interpolation

This toy, which sources are in oasis3-mct/examples/test_interpolation, 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.

To evaluate the quality of the interpolation between a source grid and a target grid, an analytical field is defined by model1 on the source grid and then interpolated on the target grid and sent to model2 (or sent to model2 and then interpolated on the target grid) where the error of interpolation is calculated. The error is defined as the absolute value of the difference between the interpolated field and the analytical field calculated on the target grid, divided by the interpolated field (multiplied by 100 to have it in %).

A. Quality of the conservative interpolation from “torc” to “lmdz”

1. Configuration file
   - Open the xml file ***data_oasis3/namcouple_map_conserv_torc_lmdz.xml*** with the GUI to see the characteristics of the coupling. This file was used to create the namcouple file of the toy loacted in ***data_oasis3/*** and ***data_oasis3/namcouple_map_conserv_torc_lmdz*** .
   - By default, the toy performs a conservative interpolation between “torc” and “lmdz” using the predefined remapping file myfile_torc_to_lmdz_CONSERV_FRACNNEI.nc. The “torc” and “lmdz” grids are defined in the “grids.nc”, “masks.nc” and “areas.nc” OASIS3-MCT grid data files.
   - To make test_interpolation work for a specific set of grids, the names of the source and target grids must also be specified in the file "data_oasis3/name_grids.dat".
2. 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 the work_test_interpolation directory (as defined in the script run_test_interpolation).
3. Analysis of the results
   - After running the test_interpolation toy model, the analytical field on the source grid “torc” is available in file "work_test_interpolation/FSENDANA_model1_01.nc", the interpolated field on the target grid “lmdz” in file "work_test_interpolation /FRECVANA_model2_01.nc" and the error field, as defined above, in file "work_test_interpolation /error.nc". The masked points are equal to -10000. You can find the minimum and maximum of the error in the output file "work_test_interpolation /model2.out_101".
   - ➢ 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.). 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_A3_mono/

B. Quality of other interpolations

In the tests below, the SCRIP library is used to calculate the weights and addresses remapping file.

1. Bilinear interpolation between ***"torc" and "lmdz"*** using the SCRIP library

- Open the GUI again using oasisgui if you closed the interface. Upload the xml file ***data_oasis3/namcouple_map_conserv_torc_lmdz.xml***
- Save it as a new xml file in ***data_oasis3/namcouple_scrip_bili_torc_lmdz.xml*** using the (File/Save as) menu to avoid to overwrite the namcouple used in III-A.
- Modify the GUI to perform a BILINEAR interpolation using explicitly the SCRIP library available in OASI3-MCT to calculate the interpolation weights and addresses (see the section 4.3 of the User Guide for more details). If you decide to keep the option MAPPING activated, you must change the name of the remapping file, else the name will be given by the SCRIP. Process the modified tabs and the "Sum" and "Config" tabs.
- Copy the namcouple file created in ***data_oasis3/namcouple_scrip_bili_torc_lmdz*** in data_oasis3.
- 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 (named either rmp_torc_to_lmdz_BILINEAR.nc or with the name you specified under MAPPING).

➢ Visually compare the results with the case A3_mono.

2. Bilinear interpolation from ***"lmdz" to "torc"*** using the SCRIP library

- Upload the xml file ***data_oasis3/namcouple_scrip_bili_torc_lmdz.xml*** in the GUI.
- Save it as a new xml file in ***data_oasis3/namcouple_scrip_bili_lmdz_torc.xml*** using the (File/Save as) menu to avoid to overwrite the namcouple used in III-B1.
- Modify the GUI to perform a BILINEAR interpolation between "lmdz" and "torc" using the SCRIP library, without any MAPPING option. Do not forget to process the modified tabs and the "Sum" and "Config" tabs.
- Copy the namcouple file created i***n data_oasis3/namcouple_scrip_bili_lmdz_torc*** in data_oasis3.
- Exchange the names of the source grid and the target grid in the namelist file "data_oasis3/name_grids.dat".
- 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 information with OASIS3-MCT, selecting a positive value for the TIMER_debug variable in the GUI in the second window "Gen" (see also the section 6.4.2 of the User Guide for more details). In the namcouple, it will correspond to the second number under the field $NLOGPRT.

1. Time performance for the "MAPPING" transformation using the "src" option for the bilinear interpolation between ***"lmdz" and "torc"*** **(**see section 4.3 of the User Guide):

- Copy the rmp_lmdz_to_torc_BILINEAR.nc from the previous working directory work_test_interpolation_B2_mono/ to the ***data_oasis3/*** directory.
- Upload the xml file ***data_oasis3/namcouple_scrip_bili_lmdz_torc.xml*** in the GUI.
- Save it as a new xml file in ***data_oasis3/namcouple_map_bili_lmdz_torc.xml*** to avoid to overwrite the namcouple used in III-B2.
- Use the GUI to assign 1 to the TIMER_debug variable, turn off the use of the SCRIP library, and turn on the MAPPING from “lmdz” to “torc” with the pre-existing remapping file rmp_lmdz_to_torc_BILINEAR.nc created in B2, specifying that the field is interpolated to the target grid on the source model process(es) and then sent to the target model process(es) ("src" option in the namcouple). Do not forget to save the namcouple and to copy it in ***data_oasis3***.
- Run the test_interpolation toy model with for example 4 processes per model and keep your results in work_test_interpolation_C1_para_src/ .
- ➢ Look at some performance timings in the files "model1.timers_0000" and "model2.timers_0000". The information given in these files are explained in the section 6.4.2 of the User Guide.

2. Time performance for the "MAPPING" transformation using the "dst" option:

- Upload the xml file ***data_oasis3/namcouple_map_bili_lmdz_torc.xml*** in the GUI.
- Use the GUI to turn on the MAPPING from “lmdz” to “torc” with the pre-existing remapping file rmp_lmdz_to_torc_BILINEAR.nc specifying that the field is first sent to the target model process(es) and then interpolated to the target grid on the target model process(es) ("dst" option in the namcouple).
- Test again the "MAPPING" transformation between “lmdz” and “torc” with the remapping file rmp_lmdz_to_torc_BILINEAR.nc in parallel with 4 processes per model using the "dst" option. In this case, the field is interpolated from the source to the target grid on the target processes.
- Keep your results respectively in work_test_interpolation_C2_para_dst/.
- ➢ Compare the time statistics to the results obtained with the "src" option. In which case and in which file is there a “pmap_001” or a “gmap_001” timer?

<u>D. Other optimization options for the "MAPPING" transformation : bfb and sum</u>

By default, the "bfb" option is activated (see the section 4.3 of the User Guide for more details) for the mapping strategy. 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,
is 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 on which might be faster.

If the models run on one process the results are of course independent of the option "bfb" and "sum" (or "opt").

1. "MAPPING" transformation using the "sum" option :
   - Upload the xml file ***data_oasis3/namcouple_map_bili_lmdz_torc.xml*** in the GUI.
   - Modify the configuration file in order to perform a bilinear interpolation from “lmdz” to “torc” with the pre-existing remapping file rmp_lmdz_to_torc_BILINEAR.nc using the ("src" + "sum") options. Do not forget to process the modified tabs and the “Sum” and “Config” tabs and to copy the new namcouple in ***data_oasis3.***
   - Run the toy in parallel, always on 4 processes for each model. Keep your results in work_test_interpolation_D1_para_src_sum/.
   - ➢ Compare the interpolated field, the error and the time statistics with the case in work_test_interpolation_C1_para_src/, corresponding to "src" and “bfb” options. Verify that the latter gives indeed bit reproducible results compared to the monoprocessor case and that this is not the case with “sum” option. Do you see any performance differences between the two cases?

2. "MAPPING" transformation using the "opt" option :
   - Upload the xml file ***data_oasis3/namcouple_map_bili_lmdz_torc.xml*** in the GUI if you closed it.
   - Modify the configuration file in order to perform a bilinear interpolation from “lmdz” to “torc” with the pre-existing remapping file rmp_lmdz_to_torc_BILINEAR.nc using the "src" + "opt" options. Do not forget to process the modified tabs and the “Sum” and “Config” tabs and to copy the new namcouple in ***data_oasis3.***
   - Run the toy model in parallel and keep your results in work_test_interpolation_D2_para_src_opt/
   - ➢ Compare the interpolated field, the error and the time statistics results with the case C2_para_src and D1_para_src_sum. 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>

This part is dedicated to create the files grids.nc and masks.nc (and areas.nc if areas are available) needed to calculated the weights and addresses mapping files, with the grid definitions of the participant's models. You can use either an NCL program or F90 programs.

1. Creation of the grids.nc, masks.nc (and eventually areas.nc) files
   - **Using NCL**: all the documentation about NCL can be found on the NCAR web site: http://www.ncl.ucar.edu/
     - ✓ Go into the directory test_interpolation/create_grids_masks_with_NCL. By default the example creates files grids.nc, masks.nc and areas.nc from the two files "data/grid_model1.nc" and "data/grid_model2.nc" containing respectively the ORCA2T and the LMDz grids.
     - ✓ Copy your grid files in the directory data.
     - ✓ Adapt the script "script_create_grids_masks_ncl.sh" using your grid names (defining the initial names of the grids and their final names in the files grids.nc and masks.nc) and your grid file names.
     - ✓ Run "./script_create_grids_masks_ncl.sh"

     You may have to adapt the NCL program create_aux_files.ncl in function of the format of your initial grid files.

- **Using F90**
  - ✓ Go into the directory test_interpolation/create_grids_masks_with_F90. By default the example creates files grids.nc, masks.nc and areas.nc from the two files "data/grid_model1.nc" and "data/grid_model2.nc" containing respectively the ORCA2T and the LMDz grids.
  - ✓ Compile using "make clean, make"
  - ✓ Adapt the script "script_create_grids_masks_f90.sh" using your grid names (defining the initial names of the grids and their final names in the files grids.nc and masks.nc) and your grid file names.
  - ✓ Run "./script_create_grids_masks_ncl.sh"

  You may have to adapt the F90 programs in function of the format of your initial grid files.

2. Quality of the bilinear interpolation between "grd1" and "grd2" grids of the participants:
   - Put the name of your source and target grids in the namelist data file "test_interpolation/data_oasis3/name_grids.dat"
   - Download the xml file ***data_oasis3/namcouple_map_bili_lmdz_torc.xml*** in the GUI**.**
   - Save a new xml file in ***data_oasis3/namcouple_scrip_bili_grd1_grd2.xml*** to avoid to overwrite the namcouple used in III-C. "grd1" and "grd2" represent the name of your grids. Do not forget to copy the namcouple in ***data_oasis3***.
   - Modify the configuration file in order to perform a BILINEAR interpolation from your source grid "grd1" to your target grid "grd2" 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_E2_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" grids of the participants:
   - Modify the name of your source and target grid in the namelist data file "data_oasis3/name_grids.dat"
   - Download the xml file ***data_oasis3/namcouple_scrip_bili_grd1_grd2.xml*** in the GUI**.**
   - Save a new xml file in ***data_oasis3/namcouple_scrip_bili_grd2_grd1.xml*** to avoid to overwrite the namcouple used in III-E2.
   - Modify the configuration file in order to perform a BILINEAR interpolation from the source "grid2" to the target "grid1" using the SCRIP library. Do not forget to copy the namcouple in ***data_oasis3***.
   - Execute "./run_test_interpolation".
   - Keep your results in work_test_interpolation_E3_mono/
   - ➢ Analyse the quality of the interpolation by plotting the interpolated field and the error of interpolation.

IV – 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

V - 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. To simplify the creation of the real coupled model it can be useful to follow the different steps below:

- create an independant module containing all the oasis routines that will be called by the models, when interfacing your models with OASIS3-MCT.
- create a namcouple for the real coupled model using the option MAPPING for the interpolations using the weight files calculated with test_interpolation, after verification. To simplify the set up, in a first step, it can be interesting to use only one simple interpolation, as the nearest neighbour remapping, for all the coupling fields.

After interfacing the real models with OASIS3-MCT, if some problems appear during the coupling, it can be useful to follow the steps below to debug the configuration:

- create a toy corresponding to your coupled model, using your real grids and your real namcouple, and reproducing the real coupled algorithm by adapting the toy tutorial.
- couple one real model with the toy model corresponding to the other real model (the toy model must send realistic values to the real model)
- couple the other model with the toy model corresponding to the other real model

VI - Compilation and running of the coupled models