
# OASIS3-MCT *regrid_environment*

August 2023

Valcke S., Jonville G.

CECI, Université de Toulouse, CNRS, CERFACS, Toulouse, France,
TR-CMGC-23-93 Technical Report

This work was carried out in the framework of the European project H2020 IS-ENES3 number 824084.

CERFACS

cnrs

is-enes
INFRASTRUCTURE FOR THE EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

## 1. Introduction

The `regrid_environment` directory offers a scripting environment to calculate the regridding weights and the regridding error for specific couple of grids and specific regridding algorithms with either the SCRIP library, ESMF or XIOS. The *regrid_environment* also allows the creation of sea-land masks of an atmospheric grid consistent with a specific ocean grid and a specific regridding library.

The SCRIP library (Jones 1999) is included in OASIS3-MCT. A version parallelized with a mix of OpenMP and MPI is available since June 2018 with the OASIS3-MCT_4.0 release (Piacentini et al. 2018).

A detailed analysis of the quality of the SCRIP library for different types of grids was realised in 2019 (Valcke & Piacentini, 2019; Jonville & Valcke, 2019). This analysis led to the conclusion that other regridding possibilities should be offered in OASIS3-MCT. A regridding benchmark was set up and the regridding functionality of different regridding libraries currently available for Earth System Modelling were evaluated. Details can be found in (Valcke et al. 2021). It was then decided to provide, with OASIS3-MCT sources, a unified scripting environment for SCRIP, ESMF and XIOS so that users can test those three libraries on their own grids for their own regriddings. This is what is now offered by *regrid_environment*.

ESMF, the Earth System Modelling Framework, (Collins et al, 2005; https://earthsystemmodeling.org) is an open-source software for coupling model components to form weather, climate, coastal, and other Earth science related applications. The scientist only codes the scientific part of a model into modular components and adapts them to the standard ESMF calling interface and standard data structures of the shared infrastructure software. The ESMF software provides the underlying layers necessary for an efficient parallel execution of the scientific applications on different computer architectures, allowing for the coupling of the module to other components, including transfer, transformation and regridding of the coupling data.

XIOS (http://forge.ipsl.jussieu.fr/ioserver) standing for XML-IO-Server is an open source library, dedicated to I/O management in climate codes. XIOS manages output of diagnostics and other data produced by climate components into files. It aims at simplifying the I/O management by supporting a maximum of on-line temporal and spatial processing, including regridding, of the data. The output definition is defined in an XML file which allows the output configuration to be changed without recompiling. Recently, XIOS has also been used not only as an IO server but also as a coupler, i.e. managing exchange of data not only between a component and a file but also between two components.

## 2. General description

The *regrid_environment* scripts and programs calculate the regridding weights and the regridding error for the regridding library, the grids, the regridding algorithm and the function chosen by the user. All operations can be done in parallel on the number of tasks chosen by the user.

A specific script run_createMasks.sh can also be used to create binary and fractional sea-land masks for an atmospheric grid consistent with a specific ocean grid and a specific regridding library chosen by the user.

### 2.1. Regridding libraries

The regridding library can be either SCRIP (“SCRP”), ESMF (“ESMF”) or XIOS (“XIOS”).

### 2.2. Grids

The seven grids supported, which are either logically-rectangular ("LR"), Gaussian Reduced ("D") or unstructured ("U"), are:

- "torc": NEMO ORCA2 rotated-stretched logically-rectangular (ocean, LR, 182x149)
- "nogt": NEMO ORCA1 rotated-stretched logically-rectangular (ocean, LR, 362x294)
- "bggd": LMDz regular latitude-longitude (atmosphere, LR, 144x143)
- "sse7": ARPEGE Gaussian reduced T127 (atmosphere, D, 24572x1)
- "icos": Dynamico low-resolution icosahedral grid (atmosphere, U, 15222x1)
- "icoh": Dynamico high-resolution icosahedral grid (atmosphere, U, 2016012x1)

### 2.3. Regridding algorithms

The regridding algorithm can be nearest-neighbour ("distwgt"), bilinear ("bili"), bicubic ("bicu"), first or second-order conservative remapping ("conserv_1st" or "conserv_2nd")[1]. Details on how each algorithm is implemented in each library are available in section 4.3, 5.1.2, 5.3.2, and 5.4.2 of (Valcke et al. 2021).

The following constraints exist:

- XIOS does not support distwgt, bili or bicu
- SCRP does not support conserv2nd for D and U grids
- SCRP does not support bili or bicu for U grids

### 2.4. Functions used to define the field to regrid

The analytical function can be either (see Figure 1):

a) sinusoid: a slowly varying standard sinusoid over the globe
b) harmonic: a more rapidly varying function with 16 maximums and 16 minimums in northern and southern bands
c) vortex: a slowly varying function with two added vortices, one in the Atlantic and one over Indonesia
d) gulfstream: the slowly varying standard sinusoid with a mimicked Gulf Stream

[1] With the FRACAREA or DESTAREA normalisation option (see details in Valcke et al. 2021 or in the OASIS3-MCT User Guide, section 4.3).

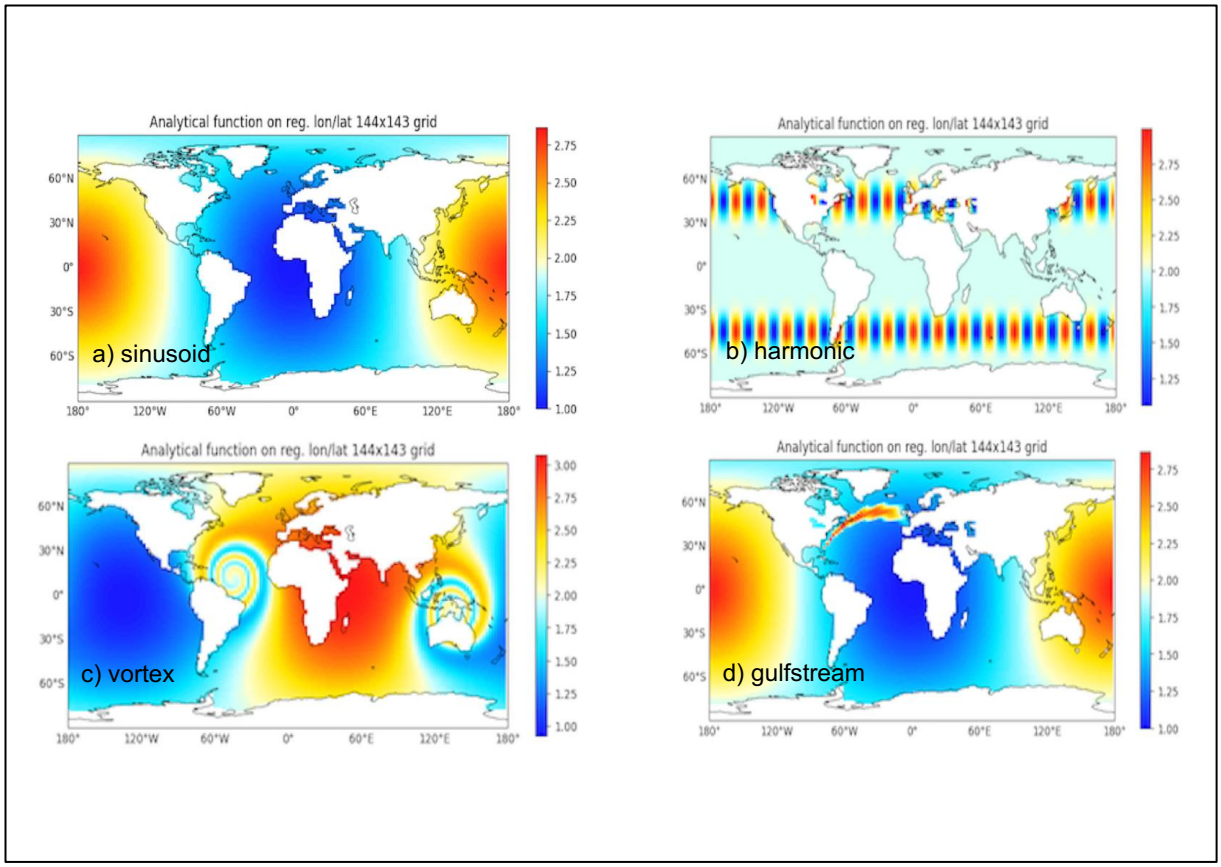


Figure 1

The source field is defined with the chosen function on the source grid and is then regridded on the target grid. The regridding error is defined as the difference between the values of the field regridded on the target grid and the values of the analytical function expressed on the target grid points and divided by the interpolated field (and multiplied by 100 to have it in %).

### 2.5. Atmospheric masks

To set up a consistent atmosphere-ocean system and have a well-posed coupled problem, the atmospheric mask should be created beforehand from the ocean mask for each specific regridding library (see the script run_createMasks.sh).

In that case, the original sea-land mask of the ocean model is taken as it is. For the atmospheric model, the fraction of water in each cell is obtained by the conservative remapping (conserv_1st_destarea) of the ocean mask (mask function) on the atmospheric grid performed with the specified regridding library. Then, the atmospheric coupling mask is created associating a valid/active index to cells containing at least a surface fraction of water (the default threshold is 1/1000). Under this threshold of water, the atmospheric cell is considered completely masked. Note that masked atmospheric cells will then have nul water fractions. This method ensures that the total sea and land surfaces are the practically same on the ocean and atmosphere grids, allowing global conservation of sea or land integrated quantities.

Note that this best practice has been followed for the regridding available in *regrid_environment* when the ocean grid, either torc of nogt, is the source grid. The original sea-land mask of torc or nogt is taken as is and the atmosphere coupling mask has been adjusted following that method for each regridding library. The different masks are available in files `/OASIS/xxxx_masks/masks_yyyy_xxxx.nc`, where `xxxx` is the regridding library (SCRP, ESMF or XIOS) and `yyyy` is the ocean grid acronym (torc or nogt).

## 3. Regridding library installation

The first step to use `regrid_environment` is to install the SCRIP, ESMF and XIOS

### 3.1. SCRIP

The SCRIP library is included in OASIS3-MCT. Instructions on how to install OASIS3-MCT, and therefore the SCRIP library, are provided in section 6 of the User Guide (https://oasis.cerfacs.fr/en/documentation/).

### 3.2. ESMF

On-line instructions to download and install ESMF are available at https://earthsystemmodeling.org . To download the sources, one can use, e.g. for ESMF 8.2.0 (we recommend to use the latest ESMF version available):

```
git clone https://github.com/esmf-org/esmf.git --branch ESMF_8_2_0 --depth 1
```

Then the following environment variables have to be defined according to the platform used (here for CERFACS platform kraken):

```
export PYTHON=`which python`
export ESMF_DIR=/scratch/globc/valcke/SOFTS/esmf
export ESMF_COMPILER=intel
export ESMF_INSTALL_PREFIX=/scratch/globc/valcke/opt/INTEL/esmf
export ESMF_INSTALL_HEADERDIR=include
export ESMF_INSTALL_MODDIR=mod/modg/Linux.intel.64.intelmpi
export ESMF_INSTALL_LIBDIR=lib/libg/Linux.intel.64.intelmpi
export ESMF_INSTALL_BINDIR=bin/bing/Linux.intel.64.intelmpi
export ESMF_INSTALL_DOCDIR=doc
export PATH=$ESMF_INSTALL_PREFIX/$ESMF_INSTALL_BINDIR:$PATH
export ESMF_NETCDF=split
export ESMF_NETCDF_LIBPATH="/softs/local_intel/netcdf/4.4.4_phdf5_1.10.4/lib -L/softs/local_intel/netcdf/4.4.4_phdf5_1.10.4/lib"
export ESMF_NETCDF_INCLUDE="/softs/local_intel/netcdf/4.4.4_phdf5_1.10.4/include -I/softs/local_intel/netcdf/4.4.4_phdf5_1.10.4/include"
export ESMF_COMM=intelmpi
export PATH=$ESMF_INSTALL_PREFIX/$ESMF_INSTALL_BINDIR:$PATH
```

ESMF can be compiled with "`gmake lib`" and "`gmake install`" in `/esmf` directory.

### 3.3. XIOS

Details on XIOS installation are available at https://forge.ipsl.jussieu.fr/ioserver/wiki/documentation.

The sources from XIOS trunk can be extracted with the SVN command:

```
svn co http://forge.ipsl.jussieu.fr/ioserver/svn/XIOS/trunk XIOS_trunk
```

The compiling environment has to be defined in the `XIOS_trunk/arch` directory in files `arch-xxxx.env, arch-xxxx.fcm, arch-xxxx.path` where `xxxx` refers to the specific platform (see examples in there). XIOS compilation can then be launched with "`./make_xios --arch xxxx`".

The program used to generate the weights with XIOS is `/XIOS/oasis_testcase.f90`. For compilation, copy `oasis_testcase.f90` to the XIOS sub-directory `XIOS_trunk/src/test` and add a line "`bld::target oasis_testcase.exe`" to

XIOS_trunk/bld.cfg. At XIOS recompilation, the executable oasis_testcase.exe is then automatically generated in directory XIOS_trunk/bin. Then copy the executable oasis_testcase.exe to /XIOS/. or create a symbolic link in /XIOS/ to XIOS_trunk/bin/oasis_testcase.exe.

## 4. How to use regrid_environment

### 4.1. Compilation of model1.F90

Once the regridding libraries are installed, one needs to compile the program src/model1.F90 This program is interfaced with OASIS3-MCT and performs an internal send (oasis_put) of a field defined by the chosen function on the source grid and an internal receive (oasis_get) of that field after regridding on the target grid, either with pre-generated weights (for ESMF and XIOS) or weights calculated during model1 execution (for SCRP).

If OASIS3-MCT is installed, it should be straight forward to compile model1.F90, using the script src/oasis_test_build.sh . Following the rules of OASIS3-MCT compiling environment effective since December 2022, $OASIS_COUPLE and $OASIS_ENV are two environment variables that must be defined. $OASIS_COUPLE gives the path of the sources of the coupler (/lib, /util, /examples, etc. directories). $OASIS_ENV gives the extension of the header Makefile make.${OASIS_ENV} from the directory ${OASIS_COUPLE}/util/make_dir used to compile OASIS3-MCT itself. Note that src/Makefile also includes a CPP key CPPKEY_FANA that must correspond to the analytical function chosen. Typing "./oasis_test_build.sh" in the /src directory will compile executable model1 in that directory.

Note that this compilation is carried out automatically by the script run_regrid.sh described in the next section if the analytical function has changed compared to the CPP key CPPKEY_FANA or if executable model1 does not exist.

### 4.2. Using the running script run_regrid.sh for one specific regridding

The script run_regrid.sh, which has 7 arguments, can then be used to launch the regridding weight and error calculation with:

```
>./run_regrid.sh $sgrid $tgrid $remap $fana $n_p_t $library $ext
```

where:

- $sgrid is the source grid acronym (bggd, sse7, icos, icoh, nogt, torc),
- $tgrid is the target grid acronym (bggd, sse7, icos, icoh, nogt, torc),
- $remap is the regridding algorithm (distwgt, bili, bicu, conserv_1st_fracarea, conserv_2nd_fracarea, conserv_1st_destarea, conserv_2nd_destarea),
- $fana is the analytical function (sinusoid, vortex, gulfstream, harmonic)[2]
- $n_p_t defines the total numbers of nodes (n), the number of MPI tasks per node (p) and the number of OpenMP threads per MPI task (t),
- $library is the regridding library (SCRP, ESMF or XIOS),
- $ext is a suffix used to define the name of the working directory.

Regarding the number of MPI tasks per node (p) and the number of OpenMP threads per MPI task (t) the following should be noticed:

[2] $fana can also be "mask" when the script is called by run_createMasks.sh to create an atmospheric mask, see section 4.5.

- ESMF and XIOS are not parallelized with OpenMP; therefore (t) should always be set to 1 and (p) should be at maximum to the number of cores per node.
- For SCRP, the weight calculation is set to enrol one MPI process per node. Therefore, (p) should always be 1. For optimum performance and to avoid hyper threading, it is then recommended to set (t) to the number of cores of the node[3].

The script run_regrid.sh generates the batch job script which depends on the computer used ("arch=$OASIS_ENV"). Currently, CERFACS LENOVO cluster kraken ("=$OASIS_ENV=kraken_intel18.0.1.163_intelmpi2018.1.163") and Météo-France supercomputer belenos ("=$OASIS_ENV =belenos_intel2018.5.274_intelmpi2018.5.274") are included. **This is the part that the user has to adapt to his/her computing platform.**

If the run is successful, the working directory (defined by variable rundir in run_regrid.sh) contains the regridding weights in file starting with "rmp_", the field regridded on the target grid in the file FRECVANA.nc, and the error field in the file REGRID_ERROR.nc. The main log file of model1, model1.out_100, also contains the mean, maximum and minimum of the regridding error (see lines with respectively "mean", "Max" or "Min").

### 4.3. How to chain the tests and verify results

The scripts run_tests_SCRP, run_tests_ESMF, run_tests_XIOS allow to chain all possible combinations of source grid, target grid and regridding algorithm for each library for a specific value of $n_p_t and $ext, which need to be provided as calling arguments, i.e.

```
>./run_tests_SCRP $fana $n_p_t $ext
```

Baseline results of the expected mean error for these combinations and $fana=sinusoid are provided in the files resu_SCRP.txt_OK, resu_ESMF.txt_OK and resu_XIOS.txt_OK for comparison. The script run_verif.sh can be used to extract the mean error of the different tests and write it to a file resu_$n_p_t $library.txt in the rundir directory; the script also prints the differences between the expected mean error and the mean error obtained for a specific library and specific values of n_p_t and ext. These 3 parameters must be provided when calling the script. i.e.

```
>./run_verif $n_p_t $library $ext
```

### 4.4. How to extract the time needed for weight calculation

Finally, the time needed to calculate the weights can be extracted from log files in the working directory:

- For SCRP, it is the value of the "Elapsed time for timer remap_* overall" in debug.root.01.
- For XIOS, it is the value of computeIndexSourceMapping in xios_client_*.out files.
- For ESMF, this specific measure is not saved. However, one can have a look at the total time of the run in the job error or output file, knowing that this time includes the weight calculation and the writing of the weights to the NetCDF file.

[3] The environment variable OASIS_OMP_NUM_THREADS will be set to the (t) value. Notice that for most of the OpenMP implementations, the number of threads activated at run time is limited by the overall value set by the OMP_NUM_THREADS environment variable. If is not set, OASIS_OMP_NUM_THREADS defaults to OMP_NUM_THREADS.

### 4.5. Using the running script run_createMasks.sh

The script run_createMasks.sh, which has 4 or 5 arguments, can be used beforehand to create the atmospheric mask following the best practice described in section 2.5. This script has to be launched with:

```
>./run_createMasks.sh $ogrid $agrid $n_p_t $library [$opt]
```

where:

- $ogrid is the ocean source grid acronym (nogt, torc),
- $agrid is the atmospheric target grid acronym (bggd, sse7, icos, icoh),
- $n_p_t defines the total numbers of nodes (n), the number of MPI tasks per node (p) and the number of OpenMP threads per MPI task (t),
- $library is the regridding library (SCRP, ESMF or XIOS),
- $opt is an optional argument to plot the binary and fractional masks with Ferret if set to -plot.

This script launches run_regrid.sh script with the above arguments and with the conserv_1st_destarea regridding algorithm using the ocean mask to define the coupling field ($fana="mask"). The binary and fractional masks of the atmospheric grid are obtained in the NetCDF file OASIS/$library_createdMasks/masks_$ogrid_$library.nc . The screenshots of any plots are also saved in this directory.

## 5. Implementation details

For sake of completeness, the different tasks launched by the script run_regrid.sh including the ones achieved by model1 are described here in more details.

### 5.1. Generation of the namcouple configuration file

After preliminary checks, the script run_regrid.sh creates the OASIS3-MCT configuration file, namcouple, needed to configure the internal exchange in model1, according to the source and target grids and regridding chosen. For ESMF and XIOS, the weights are generated before model1 execution and a generic MAPPING operation involving the weight file generated is used. For SCRP, the regridding weights are calculated during model1 execution and the namcouple is generated to fully configure the regridding chosen by the user (see regridding options in OASIS3-MCT User Guide, section 4.3).

### 5.2. Copy of required files to working directory

The script run_regrid.sh copies or links all required files from the /OASIS directory to the working directory, including the mask file masks.nc. In order to limit the size of the git repository, the grid file grids.nc is extracted from https://mercure.cerfacs.fr/oasis3-mct/examples/regrid_environment/OASIS/grids.nc with the "curl" command.

The script run_regrid.sh also generates a file name_grids.dat containing information about the source and target grids, which is read at runtime by model1, and copies this file and model1 executable to the working directory.

For ESMF, the python scripts OasisGridsToESMF.py needed to transform the grids from the OASIS3-MCT format (in grids.nc and masks.nc) to one of the two ESMF formats are

also copied to the working directory[4]. The script `ESMFWeightsToOasis.sh`, which transforms the weight file generated by ESMF to the OASIS3-MCT weight file format, is also copied.

For XIOS, the executable `oasis_testcase.exe` which execution will generate the weights (see section 3.3) is copied to the working directory. A file `param.def` defining the number of processes to run `oasis_testcase.exe` is also generated and copied to the working directory. Finally, the XML files configuring XIOS, `iodef.xml` and `context_toy.xml`, are adapted and also copied.

## 5.3. Job generation adapted to the regridding libraries

As already mentioned above, the script `run_regrid.sh` generates the batch job script which depends on the computer used ("`arch=$OASIS_ENV`"). Currently, CERFACS LENOVO cluster kraken ("`kraken_intel18.0.1.163_intelmpi2018.1.163`") and Météo-France supercomputer belenos ("`belenos_intel2018.5.274_intelmpi2018.5.274`") are included. **This is the part that the user has to adapt to his/her computing platform.**

### 5.3.1. SCRP

For SCRP, the job will simply execute `model1`, which performs both the regridding weight and error calculation.

### 5.3.2. ESMF

For ESMF, the job first transforms the grids described in the OASIS3-MCT format into one of ESMF format (with python scripts `OasisGridsToESMF.py`, see above). Then it executes, in parallel on the total number of processes available, the command `ESMF_RegridWeightGen` which performs the regridding weight calculation. The options used for the different regriddings are:

- distwgt ("neareststod" in ESMF):
  `--src_loc center --dst_loc center --ignore_degenerate`
- bili ("bilinear" in ESMF):
  `--extrap_method neareststod --src_loc center --dst_loc center --ignore_degenerate`
- bicu ("patch" in ESMF):
  `--extrap_method neareststod --src_loc center --dst_loc center --ignore_degenerate`
- conserv_1st_fracarea ("conserve" in ESMF):
  `--ignore_unmapped --norm_type fracarea --ignore_degenerate`
- conserv_2nd_fracarea ("conserve2nd" in ESMF):
  `--ignore_unmapped --norm_type fracarea --ignore_degenerate`

[4] Two formats can be used to describe a grid in ESMF: the so-called "SCRIP" format or the "unstructured" format. The "SCRIP" format describes the grid providing the latitude and the longitude of the centre and corners of each cell. The unstructured format describes the grid as an ensemble of elements and provides the element connectivity associating for each element a certain number of nodes in the list of nodes for which the latitude and longitude are provided. The grids torc and nogt are structured and can in principle be described in ESMF with the SCRIP format. However, as detailed in section 7.4 of (Valcke et al. 2021), the conservative remapping for a field expressed on a nogt grid shows wrong values in the region of the North fold of the grid. But this problem disappears when the nogt grid is expressed as an unstructured grid. This is why the script `OasisGridsToESMF.py` expresses the nogt grid in the unstructured ESMF format, for conserv1st and conserv2nd when nogt is the source grid.

These options have the following effect:

- `--src_loc center --dst_loc center` : allow non-conservative regriddings on the cell centre locations of an unstructured grid defined with the ESMF unstructured file format;
- `--extrap_method neareststod` : each target point that does not receive a value with the original algorithm uses the closest unmasked source point to define its value; this option was activated for non-conservative algorithms to reproduce the SCRP default behaviour;
- `--ignore_unmapped` : do not do anything special for target point that does not receive a value with the original algorithm; this option was activated for conservative algorithms to reproduce the default SCRP behaviour;
- `--ignore_degenerate` to ignore degenerate cells in either the source or the destination grid; this can be useful for the torc and nogt grids which may have masked cells (i.e. not used in the regridding) collapsing into a point or line.

Then the script `ESMFWeightsToOasis.sh` is called to transform the weight file generated by ESMF to the OASIS3-MCT weight file format.

Finally `model1` is executed to calculate the regriding error.

#### 5.3.3. XIOS

For XIOS, the regridding weights are first generated with the execution of `oasis_testcase.exe`, in parallel on the total number of processes available.

Then the weights are transformed into the OASIS3-MCT format with the script `/XIOS/XiosWeightsToOasis.py`

Finally, `model1` is executed to calculate the regriding error.

### 5.4. Details about model1.F90

The program `src/model1.F90` calculates the regridding error using either pre-generated weights (for ESMF and XIOS) or weights calculated during its execution (for SCRP). This program is interfaced with OASIS3-MCT and performs an internal send (`oasis_put`) of the source field defined by the chosen function on the source grid and an internal receive (`oasis_get`) of that field after regridding on the target grid. The source and target grid characteristics are specified in the file `name_grids.dat` generated by `run_regrid.sh`. based on the user choices.

The program performs OASIS3-MCT initialization calls and defines a source grid partition and a target grid partition, which depends on the grid type. If the grid is logically rectangular ("LR"), it is split in rectangle partitions having the dimension of the global grid in x and the dimension of the global grid divided by the number of MPI processes available in y; if the grid is unstructured ("U") or Gaussian Reduced ("D"), the first dimension of the grid, which covers in fact the whole grid (the 2nd dimension is always 1 for those grids), is split in `npes` segments, where `npes` is the number of MPI processes available (see `def_parallel_decomposition.F90`). The program then reads the grid and mask definition in files `OASIS/grids.nc` and `OASIS/masks.nc`.

The program declares the field associated with the source and target partitions calling `oasis_def_var` and closes the initialisation phase calling `oasis_enddef` . Then it defines the values of the field with either the sinusoid, harmonic, vortex, or gulfstream analytical function (see Section 2 above) depending on the value of the CPP key value (`Fsinusoid`, `Fharmonic`, …) activated in the `Makefile` (see `CPPKEY_FANA`).

If SCRP is used and if bicu is chosen for an LR grid, the gradients in the three directions, needed as extra fields in the `oasis_put` call, are calculated. If conserv2nd is chosen for an LR grid, the gradients in the longitudinal and latitudinal directions, needed as extra fields in the `oasis_put` call, are calculated. The source field is then sent with an `oasis_put` call and received on the target grid with an `oasis_get` call.

The received field is transformed to give the value of 10000 to masked target point and the value of 1.e20 to non-masked points that did not receive any interpolated value (if any), in order to be able to easily detect them. The error field is then calculated as the difference between the values of the received field and the values of the analytical function expressed on the target grid points, divided by the analytical field (and multiplied by 100 to have it in %). On masked points, the error is set to 0; on non-masked points that did not receive any interpolated value, the error is set to -1.e20.

The received field and the error are gathered on the master process and written in files `FRECVANA.nc` and `REGRID_ERROR.nc` respectively. The error global minimum and maximum on non-masked points are calculated and written out to the log file `model1.out_100` .

Finally, the program calls `oasis_terminate` to finalize the run.

## 6. Using `regrid_environment` for other couples of grids

`regrid_environment` can be adapted to calculate the weights for your own grids and regriddings, going through the following steps:

- include the definition of your own grids in the OASIS3-MCT grid data files `grids.nc` and `masks.nc`. See section 5.2 above for details on the masks and section 5.1 of the User Guide for details on OASIS3-MCT grid data files. To create these files with either with F90 or NCL, you can also adapt sources in directories `/create_grids_masks_with_F90` and `/create_grids_masks_with_NCL`.
- Modify the script `run_regrid.sh`, adapting the checks on the grids (see "`Check grids`"), on the regridding algorithm (see "`Check remap`") and adding the grid characteristics (see "`Source grid characteristics`" and "`Target grid characteristics`"). Make sure that the proper `masks.nc` file is copied to the working directory (see section 2.5).
- Adapt the script `namcouple_create.sh` to generate a `namcouple` file corresponding to your grids and regridding algorithm.
- For XIOS, you will have to add your grid in `oasis_testcase.f90` in subroutine `init_model_params` and `init_domain` .

You should then be able to use *regrid_environment* to calculate the regridding weights and the regridding error for your own grids and algorithm, and therefore evaluate the quality of your regriddings. In practice, defining what a "good" regridding precisely means is impossible. But at least, *regrid_environment* should allow you to identify bugs (if any) and to compare the quality of SCRIP, ESMF and XIOS for your own grids and regriddings .

Of course, you can always ask the OASIS3-MCT developers (oasishelp@cerfacs.fr) for advice regarding your specific results!

REFERENCES

Collins, N., Theurich, G., DeLuca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang, W., Hill, C. and da Silva, A. 2005: Design and Implementation of Components in the Earth System Modeling Framework, Int. J. High Perfor. Comput. Apps, 19 (3), 341–350.

Jones, P. 1999: Conservative remapping: First-and second-order conservative remapping, Mon. Weather Rev., 127, 2204–2210.

Jonville, G. and Valcke, S. 2019: Analysis of SCRIP conservative remapping in OASIS3-MCT – Part B, Technical Report TR/CMGC/19-155, Cerfacs, France. https://oasis.cerfacs.fr/wp-content/uploads/sites/114/2021/08/GLOBC_TR_Jonville-SCRIP_CONSERV_TRNORM_partB_2019.pdf

Piacentini, A., Maisonnave, E., Jonville, G., Coquart, L. and Valcke, S. 2018: A parallel SCRIP interpolation library for OASIS, Technical Report TR/CMGC/18-34, Cerfacs, France. https://oasis.cerfacs.fr/wp-content/uploads/sites/114/2021/08/GLOBC_WN_Piacentini_Parallel_SCRIP_cmgc_18_34_2018.pdf

Valcke, S. and Piacentini, A. 2019: Analysis of SCRIP conservative remapping in OASIS3-MCT – Part A, Technical Report TR/CMGC/19-129, Cerfacs, France. https://oasis.cerfacs.fr/wp-content/uploads/sites/114/2021/08/GLOBC_TR_Valcke-SCRIP_CONSERV_TRNORM_partA_2019.pdf

Valcke, S., Piacentini, A. and Jonville, G. 2021: Benchmarking of regridding libraries used in Earth System Modelling: SCRIP, YAC, ESMF and XIOS,Technical Report TR/CMGC/21-145, Cerfacs, France. https://oasis.cerfacs.fr/wp-content/uploads/sites/114/2021/11/GLOBC-TR_Valcke_Report_regridding_analysis_final_2021.pdf