PoCO
Post-processing coupled with OASIS

Eric Maisonnave

TR/CMGC/13/70

“Lo bueno, si breve, dos veces bueno.
Lo malo, si poco, no tán malo”
B. Gracián

Table of Contents

Abstract

Taking benefit of OASIS3-MCT interpolation facilities, an new output system called PoCO (Post-Processing Coupled with Oasis) has been design to replace the existing ARPEGE output library (FULLPOS). Several independent executables were coupled with ARPEGE to receive diagnostic variables that can be interpolated on different grids and written in Netcdf format, asynchronously to ARPEGE calculations. Tests performed with T359 (50Km) high resolution configuration show weak time dependency to output variable number and negligible absolute response time compared to FULLPOS ones.

Rationale

During the last few years, supercomputers processing capacities grew faster than disk storage bandwidth, with the unavoidable consequence to bound those computational capacities. It was not necessary to wait for Exascale to reach the point where computations had to wait for output completion before resuming. At CERFACS, this limit was exceeded during a seasonal forecast experiment, on the occasion of a PRACE project access [1]: the amount of output data has to be drastically reduced, to a few global high resolution (50Km) horizontal atmospheric variables.

An analysis of this bottleneck issue showed that (i) coupled simulation restitution time was increasing due to model output writing, (ii) storage was saturated by huge amount of data that (iii) considerably slowed down post-processing and (iv) was impossible to bring back to CERFACS local disk through standard internet connection.

Better said, a huge amount of data is difficult to handle (disk read/write + internet transfer). So: let's reduce the data !

Until now, our standard strategy consisted in writing the maximum amount of model diagnostics to the disk, assuming that numerous scientists will choose in this repository a subset of variables needed for their specific studies. Considering that (i) this maximum now gathers too few variables and (ii) computing time is becoming cheaper[1] than storage, we assume that, from now on, the user will have to determine beforehand which variables with which characteristics (resolution, frequency, geographical extension) is necessary to his (her) study. And possibly will replay the same simulation, outputting different variables for each different studies.

Following this idea of data reduction, it is then possible to conceive that (at least a part of) the analysis necessary for the study can be done *during* the simulation and not in a post-processing phase. This idea is very important to help us reducing disk access: if a model can broadcast its diagnostic variables during the simulation to a post-processing executable, most of disk writing is avoided. This post-processing tool will necessarily produce much less data than the model itself.

Our last assumption is that, when using a high resolution model, output at model full resolution is not necessary for every variable and/or for every grid points. An interpolator is then needed between model and our new post-processing executable.

Recently, software such as XIOS [2], CDI [3], CFIO[4] or ADIOS [5] have been developed to address IO slowing down issues, allowing to a-synchronize model computations and output. Some of then include interpolation facilities, but it seemed easier for us to rely on a tool already available in practically every major coupled system in Europe: OASIS3-MCT. Moreover, we preferred to keep as simple as possible our first post-processing tool. It mainly consists in a single short Fortran program. Subsequently, it could be replaced (or assisted in parallel) by a more sophisticated post-processing engine[2].

Obviously, this solution implies more anticipated engineering to identify relevant data for a given study and instantiate Fortran post-processing solutions. And it is clearly not recommended for uncoupled models.

1 By "cheaper" we mean that it seems easier to have access to a massive number of processors instead of a decent amount of data storage or an efficient file system to go from the one to the other.

2 For example: a parallel version of CDO or NCL.

PoCO program functioning description

We call “PoCO”[3] our post-processing solution. It consists on a Fortran executable, coupled to a model through OASIS. In this document, only one component of the described climate model has been modified (ARPEGE), considering that NEMO was already using XIOS for asynchronous output. This configuration strongly simplifies the previous workflow, including ARPEGE post-processing phase within the main parallel program.

Workflow

In figure 1, we present how the entire ARPEGE workflow has been transformed and what improvement has been gained.

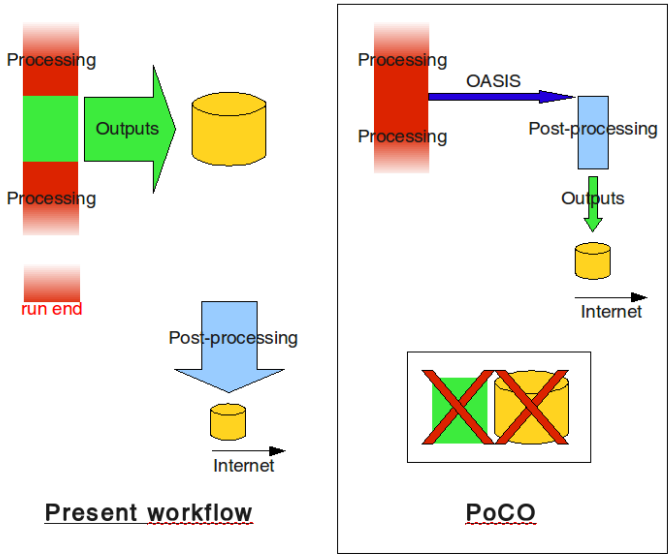

Fig 1: PoCO workflow description and advantages

On the left part of the graphic, we show our workflow as it goes before PoCO improvements.

During ARPEGE simulation, an output routine (FULLPOS, green box) is periodically called between calculation phases (red boxes). Calculations stop during output (sequential call). Diagnostic variables are written (green arrow) on disk (orange cylinder section). When simulation ends, a post-processing tool (blue arrow) reads simulation output files, process format conversion (FA to Netcdf) on selected variables and write Netcdf files. At experiment end, additional analysis can be done to reduce even more data amount (like EOF or yearly means). Results are brought back and stored to local disk via Internet.

On the right part of the graphic, PoCO solution is described.

Model calculations (red boxes) only stop to send (blue arrow) output variables through OASIS (standard non blocking MPI_ISEND *after* MPI_WAITALL) like any other coupling field. Those data are not written in files but interpolated (if required) and received (MPI_IRECV) by a post-processing executable (blue box). Converted data are written in files (green arrow) and workflow follows as previously.

Main consequences are (i) the removal of output elapsed time during simulation[4] and (ii) the removal of comprehensive model diagnostic writing in files.

3 for **Po**st-processing **C**oupled to **O**ASIS

4 considering that it is replaced by oasis_put subroutines that only last for non blocking MPI_WAITALL + MPI_ISEND durations

This last aspect is particularly important in our case because all FULLPOS variable were output with the same frequency, in the same vertical levels (when 3D fields) and in the same model horizontal grid. Consequently, the greatest accuracy was set for all output variables. With OASIS, it is possible to prescribe those 3 parameters field by field.

Parallelism of simulation and post-processing[5] is another advantage: results are immediately available when simulation ends. And no additional undersized jobs are required on supercomputers for post-processing.

Data storage is strongly reduced[6] and data are only written once (and never read[7]) on disks. Transfer through the Net is facilitated.

The old post-processing part of our workflow is removed, but the workflow main part (the coupled run) has to be adapted: an OASIS interface must be set in the model (here: ARPEGE), and the PoCO post-processing tool must be developed.

An example of model interface (ARPEGE)

As a first step, the existing FULLPOS library must be disabled.

Actually, only a portion of it has been bypassed: the data gathering on output nodes and the effective writing on disk. FULLPOS also collects all diagnostic variables on easily identifiable structures, according to user specification made through model namelist. It reorders variables on geographically coherent arrays and perform vertical interpolation when needed. Given that those operations represent less than 10% of the total FULLPOS duration[8], we preferred to re-use these FULLPOS routines and do OASIS calls within one of them[9].

A variant could be to catch variables as soon as they are calculated, without geographical re-ordering, following the order set for calculations[10]. With no obvious gain, the main inconvenient of this method is that a different partitioning must be declared during OASIS setting phase[11]. The high geographical non coherence of this partitioning supposes more communications with the coupled post-processing tool, then worse performances.

Our output variables have the same status as any other coupling variable. They must be declared during the OASIS setting phase. To identify them, its list is read in an OASIS global variable[12] (itself read in the OASIS parameter file, “namcouple”, see figure 2).

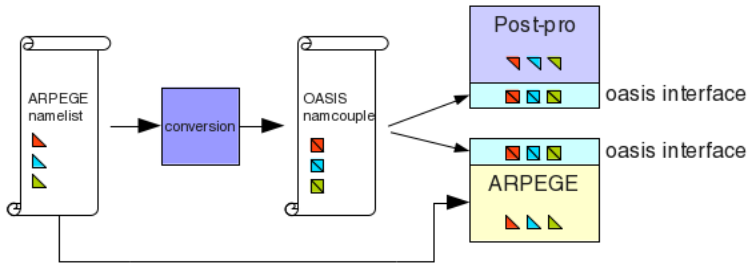

Fig 2: Output fields declaration and reading by models

5 which should now be called co-processing
6 now, data are few (“poco” in spanish)
7 we mean: “never read during the production phase”. We hope that, at some point, someone reads them
8 FULLPOS spends 12 seconds to process 570 2D fields (50Km resolution), from which 11s for gathering and output (on BULLX thin nodes, TGCC, CEA)
9 “WRHFP” routine
10 Arrays dimensions are changed according to NPROMA vector length. One length is optimum for one configuration on one machine (depending on processor cache length).
11 “oasis_def_partition” routine
12 “namsrcfld” variable

All OASIS send operations are done in a row within FULLPOS, following output model frequency[13].

Taking benefit of OASIS non intrusiveness, code modifications are few[14]. Modifications mostly lie in checking namcouple/ARPEGE namelist coherence.

OASIS namcouple

As usual, ARPEGE output field names are defined in ARPEGE namelist.

But now, a dedicated pre-processing shell script converts this information into namcouple-style coupling field definition. This information is added to the namcouple file (see figure 2).

Users can activate this automatic mode[15] or choose to modify their namcouple before the main executable launching, changing output frequency, extension and/or resolution according to their specific needs.

OASIS3-MCT also offers the possibility to send several times to the post-processing tool a field that was output only once by ARPEGE. For example, this seems useful if a variable needs to be stored on the global grid at low resolution and, at the same time, at higher resolution on a regional area (figure 3).

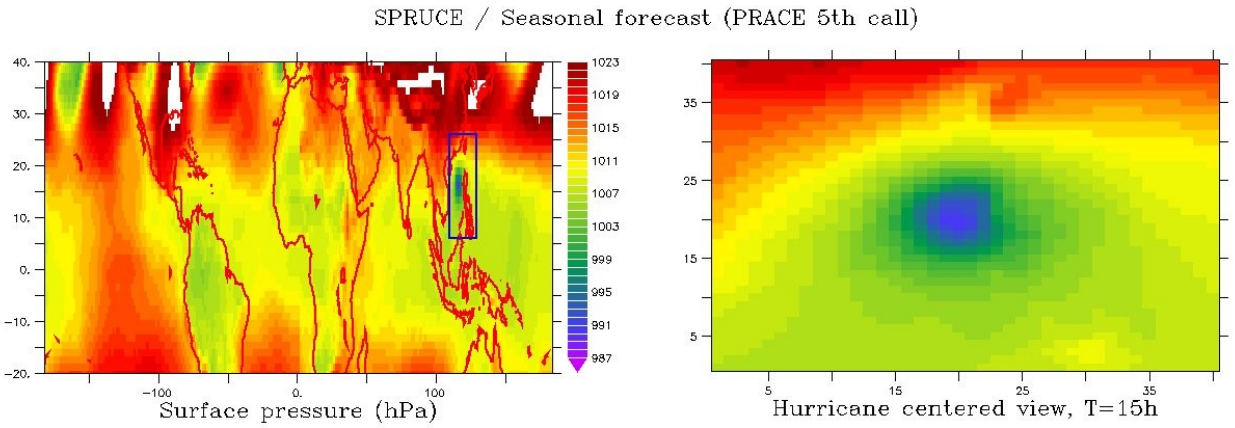

Fig 3: Same field (Surface Pressure) stored in T127 grid at global scale, and in 50Km regular grid centred over a typhoon eye

The different destination grids are defined by the user. Grid point values of its latitude, longitude, land-sea mask and areas are put in the OASIS auxiliary files[16].

PoCO takes benefit of all existing OASIS3-MCT developments made for parallel interpolation. Which is not few[17]:

- Several kind of interpolation can be calculated by OASIS (with SCRIP software) for various meshes and partitioning.
- When it is not enough, user defined weights and addresses pre-calculated files can be used.
- Interpolation can be made either in ARPEGE (source model) or in post-processing tool (destination model). Considering that the post-processing tool runs faster, interpolation must be done by it. But when interpolation is particularly fast (output on model grid, for example) and the amount of data

13 NFRPOS, user defined in ARPEGE namelist, must be set at the highest frequency of all output variables. Data are only send by OASIS at appropriate frequency for each, according to user's specifications in namcouple
14 "poco", in spanish
15 all variable are then output at maximum NFRPOS frequency, as previously.
16 and named as usual with 4 letters following this convention: the first 2 must be "Po" (upper/lower case counts) and the last two must be a figure on two digits
17 Yes, you get it: it's false modesty

particularly big, it could be interesting to let ARPEGE doing it, to give more time to the post-processing tool (for writing).

- Some variables can be masked (vegetation on land points only).

If more sophisticated operations, that are not available in OASIS, are necessary (variance, EOF ...) they can be done within the post-processing tool[18].

The coupled post-processing tool

It consists in a few hundred of Fortran code lines. Its basic purpose is to receive variables as processed by OASIS and to rewrite them in Netcdf format. Simplified algorithm is described in figure 4.

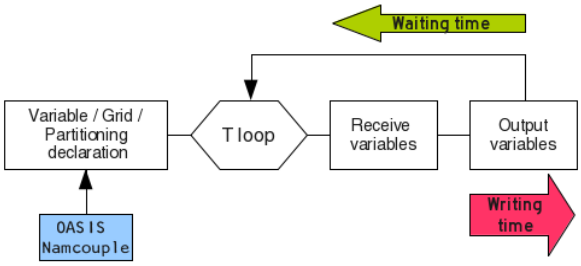

Fig 4: Post-processing tool algorithm and timings

On a first step, incoming variables are declared during the OASIS setting phase. As for ARPEGE, coupled field names are read in the corresponding OASIS global variables (set by namcouple). Similarly, coupling frequencies and mesh sizes are deduced from namcouple definitions. The post-processing tool is not parametrizable with namelist: the needed information lie in the namcouple (those informations are summarized in figure 2).

A time loop is defined, with a time step equal to the highest coupling frequency of the output variables.

Each variable is received on the whole mesh. To avoid a bottleneck in memory at this stage, a pseudo-parallelism[19] by field is implemented. The post-processing tool is launched in several MPI process, each of them receiving a subset of the ARPEGE output variables. Benefit is double: OASIS MPI communications are not focused on a single process and Netcdf output are parallel (per-field parallelism). Post-processing tool parallelism (and mapping on allocated resources) is simply set by user, configuring the “mpirun” command.

More parallelism (not necessary at the moment) would consist in a regular geographical partitioning, configurable with OASIS.

Within the time loop, all the variables coupled at a given time step are received in a row and stored in a temporary array. Variables of this array are then written at the right temporal position in a Netcdf file (one file per coupled variable). Those two steps are necessary to avoid that disk writing slowed down reception of coupling fields.

In addition, a timing is implemented to evaluate durations of Netcdf writing, and waiting for the first coupling field after the last Netcdf writing.

Operations performed by our post-processing tool are simple. Potentialities are mainly bounded by the time needed by ARPEGE to perform its calculations between two output variable sending.

As example, a rudimentary routine calculating hurricanes eye position has been implemented. It allows to reduce the variable geographic horizontal extension to the hurricane surrounding. This area is moving

18 Contradicting the famous Gascon adage: “dab lou hilh deu diable d’Oasis, que’m podi tot har”
19 see A. Caubel developments for OASIS3 pseudo parallel mode [6]

following the hurricane eye[20].

Variables coming from different modules of the coupled system fitted to PoCO could be combined. The same, if different instances of a coupled model could be launched in a single “mpirun” command[21]. Here, the possible result would be to write in disk variables calculated only on ensembles (like seasonal forecast scores) instead of each individual results of the different members.

Performances

A set of one-month long climate simulations is performed on CEA-TGCC Bullx supercomputer [7]. ARPEGE and NEMO model components are discretized respectively in T359 (50Km, 181,724 grid points) and ORCA025 (25km) mesh grids. ARPEGE partitioning is spread on 504 cores of Sandy-Bridge 2x8-cores (thin) nodes, NEMO on 496. ARPEGE is instrumented with PoCO, while NEMO is using XIOS for its outputs. 11 coupling fields are exchanged every 3 hours. ARPEGE output period is always 3 hours.

Three different experimental setup are presented, with different ARPEGE output systems:

- the original FULLPOS subroutine, producing one file per output period, ARPEGE format (single precision integer), at model resolution (vector of 181,724 grid points, Gaussian grid)
- PoCO system, with post-processing tool coupled with OASIS launched on a single core, producing one file per output field, Netcdf format (single precision real), at T127 resolution (vector of 24,572 grid points, Gaussian grid)[22]
- same than previous, with post-processing tool on 24 cores in parallel[23].

Figure 5 shows how much lasts a 1-month long simulation for these 3 systems, with a variable number (6 to 767) of output fields[24]. Let's precise that, for regular purposes, no more than a few tenths of variables are usually output: our experiment clearly outreaches the present needs. Calculation time is approximately equal to 25 minutes.

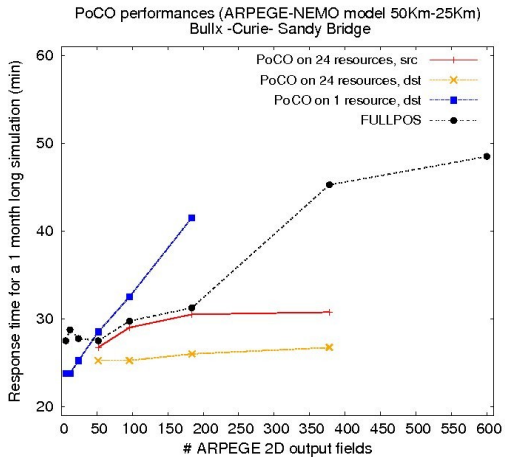

Fig 5: Duration of a one month-long simulation with FULLPOS or PoCO output system

Variability on restitution time is quite high, mainly because it depends on how a huge number of variable are written on disk through a file system (Lustre) which performances can varying a lot depending on machine load. Nevertheless, differences between the 3 graphics are big enough to conclude that PoCO (parallel) is more efficient than without parallelism or than the previous output system.

A profiling on 2 of the 3 main operations performed by the post-processing tool reveals (figure 6) that most of the time is spent in waiting. Netcdf writing is quick (less than 2 min). OASIS receiving time is hard to determine, mainly because it is done in parallel on various nodes. According to previous measures [8], we

20 … enabling a possible dynamical coupling with a regional model

21 we mention that, even though MCT is not a re-entrant library, an OASIS3-MCT coupled model has been designed with one MCT-parallelism-based component [9]. To run several OASIS3-MCT in parallel should not be a problem

22 Measurements for this system fails with up to 200 output fields (for one single post-processing tool)

23 The 26 post-processing tools are split on two different nodes (16+10). Splitting on more nodes slightly increase performances

24 Concerning the 767 output fields measure, fields are send twice to the post-processor: once on the global T127 Gaussian grid, once on a high resolution 276x122 regional grid. Notice that the total size of this global/regional output set is 4 times smaller than the size of the corresponding FULLPOS production.

expect that it should not exceed interpolation duration.

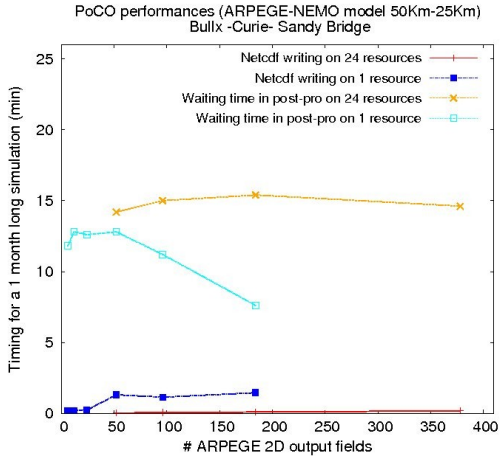

Fig 6: Duration of Netcdf writing and coupled fields waiting in post-processing tool

This interpolation duration depends on where interpolations are performed. OASIS allows to perform them either on source - ARPEGE - or destination - post-processing tool(s) – models[25]. We can choose to process these interpolations in parallel in the source model, but one variable after the other, or in the whole target grid, but split per fields on 24 processes of the target model. This option is the most efficient, mainly because it can be done by the post-processing tool in parallel (asynchronously) with the other ARPEGE computations. In Figure 7, we present the total duration (1-month long simulation) of OASIS interpolations in both case.

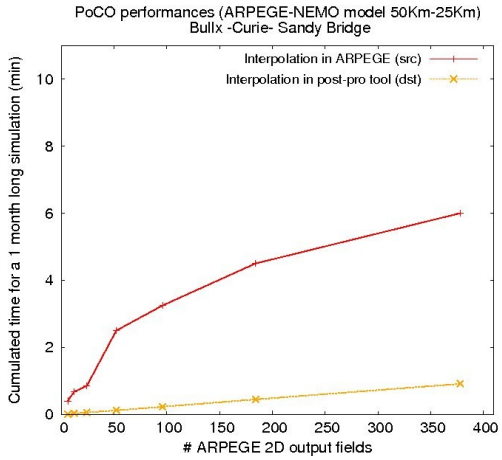

Fig 7: Duration of OASIS interpolations (1 per variable, source grid T359, target grid T127)

No interference with OASIS coupling has been observed (NEMO is always the fastest model).

Performances can be further enhanced, improving post-processing tool or OASIS functioning.

A parallel Netcdf library can be easily included in our tool to speed up writing phase when it will become necessary: OASIS will be able to perform interpolations with a fully parallel version of our post-processing tool as soon as its decomposition will be properly declared.

When output field number grows, namcouple reading can become a serious bottleneck. Legacy ASCII file reading would be advantageously replaced by a more efficient system. Gathering output variables in a single namcouple line, to couple several variables at the same time (see for example 3D variable coupling in [10]) would be a possible bypass.

In conclusion, PoCO system included in our high resolution model has proved its efficiency. We expect even better performances with higher resolution configurations. Due to current limitations, real tests cannot be lead at the moment with ARPEGE. An implementation in a T799 Ec-Earth configuration would help us to strengthen our conclusion.

25 Thanks to a simple option of MAPPING transformation which can be set on the namcouple file (src or dst).

We acknowledge that the results in this paper have been achieved using the PRACE Research Infrastructure resource Curie based in France at TGCC

References

[1] Maisonnave, E., Cassou, C., Coquart, L., Déqué, M., Ferry, N., Guérémy, J.-F., Piédelièvre, J.-P., Terray, L. and Valcke, S., 2012: PRACE Project Access for Seasonal Prediction with a high ResolUtion Climate modEl (SPRUCE) ,Working Note, **WN/CMGC/12/29**, SUC au CERFACS, URA CERFACS/CNRS No1875, France
[2] http://www.irisa.fr/orap/Forums/Forum29/PRESENTATIONS/IPSL-ORAP.pdf
[3] https://code.zmaw.de/attachments/4206/cdi_fman.pdf
[4] Huang, X., Wang, W., Fu, H., Yang, G., Wang, B., and Zhang, C.: A fast input/output library for high resolution climate models, Geosci. Model Dev. Discuss., 6, 4775-4807, doi:10.5194/gmdd-6-4775-2013, 2013.
[5] http://users.nccs.gov/~pnorbert/ADIOS-UsersManual-1.5.0.pdf
[6] https://verc.enes.org/oasis/oasis-dedicated-user-support-1/documentation/oasis-3-user-guide
[7] http://www-hpc.cea.fr/en/complexe/tgcc-curie.htm
[8] Maisonnave, E: 2013: A regional coupling with OASIS3-MCT ,Working note, **WN/CMGC/13/34**, SUC au CERFACS, URA CERFACS/CNRS No1875, France
[9] Valcke S., 2013: The OASIS3 coupler: a European climate modelling community software, Geosci. Model Dev., **6**, 373-388, doi:10.5194/gmd-6-373-20-13
[10] Maisonnave, E., Hill, R., Jamil, O. and Valcke, S., 2013: OASIS Dedicated User Support 2012, Annual report ,Technical Report, **TR/CMGC/13/18**, SUC au CERFACS, URA CERFACS/CNRS No1875, France