TECHNICAL MEMORANDUM

673

Coupling of the NEMO and IFS models in a single executable.

Kristian Mogensen, Sarah Keeley and Peter Towers

Research Department

April 2012

European Centre for Medium-Range Weather Forecasts
Europäisches Zentrum für mittelfristige Wettervorhersage
Centre européen pour les prévisions météorologiques à moyen terme

Series: ECMWF Technical Memoranda

A full list of ECMWF Publications can be found on our web site under:
http://www.ecmwf.int/publications/

Contact: library@ecmwf.int

European Centre for Medium-Range Weather Forecasts
Shinfield Park, Reading, RG2 9AX, England

Abstract

A new way of coupling the Integrated Forecasting System (IFS) atmosphere model and the Nucleus for European Modelling of the Ocean (NEMO) ocean model based on an integrated single executable system with a common time step loop is presented.

Details of the technical implementation are presented to give a fairly complete overview of how the IFS and the NEMO models interact in the single executable system.

Initial technical performance testing is showing similar or better performance than the existing OASIS3 based system and potential for coupling at high resolution. Some potential improvements to the new system is discussed.

1 Introduction

This technical memorandum describes the implementation of a single executable coupled atmosphere and ocean modelling system consisting of the ECMWF Integrated Forecasting System (IFS) atmosphere model and the Nucleus for European Modelling of the Ocean (NEMO) ocean model (Madec (2008)).

Traditionally the coupled atmosphere/ocean system used at ECMWF for the ensemble prediction system (EPS) from day 10 (Vitart *et al.*, 2008) and in seasonal forecasts from day 0 (Stockdale *et al.*, 2011; Molteni *et al.*, 2011) has been based on various versions of the OASIS coupler. A short introduction to the technical aspect of the coupled systems which have been used at ECMWF will now be given.

For the coupled systems based on the HOPE (Hamburg Ocean Primitive Equation) ocean model (Wolff *et al.*, 1997), which were used until the introduction of the NEMO model until the fourth quarter of 2011, the OASIS2 coupler (Terray *et al.* (1998)) was used; it communicates information between the atmosphere model and the ocean model via files. Launching the coupled system consisted of the following steps:

1. Starting a single threaded execution of the coupler.
2. Starting a multithreaded (via OpenMP[1]) execution of the HOPE ocean model.
3. Starting a multithreaded (via OpenMP) execution of the IFS.

The end of the execution of the coupled model happened when all three executables had finished their tasks or an error condition occured. During the coupling exchanges data were written to disk and signal files used by each component to inform the other components that data were ready to be read. This system relied on shared memory parallelization of the IFS and the HOPE models.

For high resolution modelling which needs to run on modern supercomputers with a distributed memory system the threaded execution model is only supported with a single "node". This limits the number of processors that can be used. A more general execution model based on explicit (*e.g.* via the Message Passing Interface (MPI)[2]) or implicit (*e.g.* via Co-Array Fortran[3]) message passing needs to be used. The initial version of the coupled IFS/NEMO system is based on the OASIS3 coupler (Valcke (2006)) which can use MPI for coupling. The IFS/NEMO initial system uses the MPI1 option of OASIS3 where launching of the coupler system is done as a Multiple Program Multiple Data (MPMD) set of tasks with

[1] http://openmp.org/wp/
[2] http://www.mcs.anl.gov/research/projects/mpi/
[3] http://www.co-array.org/

a single task for OASIS3 and a user specified number of tasks for both the NEMO and the IFS model. As part of the start up process of the coupled system the MPI_COMM_WORLD communicator (defining the global "addresses" of MPI tasks) is split into "private" communicators for the NEMO model and for the IFS model. All MPI communications within the component models is then done via these "private" communicators whereas exchange of data between models is done via communications with the coupler task through MPI_COMM_WORLD. Exchange of a field from the NEMO (or IFS) model and the IFS (or NEMO) consists of: a gather operation of the field to the OASIS3 task - a regridding of the field due to the different model grids; followed by a scatter operation of the regridded field to the IFS/NEMO model. Later versions of OASIS3 allow for multiple OASIS3 tasks to be used which each can handle a single full field to be exchanged, but this version has not been implemented at ECMWF.

An alternative approach to coupling IFS and NEMO is to integrate the NEMO and the IFS models into a single executable and let them share a common time stepping loop. The remainder of this technical memorandum is devoted to describe the initial implementation of such a single executable coupled IFS/NEMO system in some details and show some basic performance of the system from a purely technical point of view. The focus is on the technical aspects of the coupling and we will not discuss the science behind the coupled system except for the next introductory section.

2 Physical basis for coupling atmosphere and ocean models

Before going into the technical details we provide a brief overview of how the coupled IFS/NEMO model works. In coupled mode the atmospheric model runs for a period (the coupling interval) and surface fluxes are accumulated. These fluxes are then passed to the ocean model which uses them as the upper boundary conditions when running the ocean model. After this run the updated ocean fields can be used as the lower boundary condition of a subsequent atmosphere model run.

The above logic implies a sequential loop of atmosphere followed by ocean followed by atmosphere runs. To allow the ocean and atmosphere to run in parallel with OASIS3 the ocean is lagged with one coupling interval, so the ocean fields used by the atmosphere is one coupling interval behind the model time (illustrated in figure 1). A fully sequential atmosphere/ocean cycle is more physically correct, but that would require the atmosphere model to wait for the ocean model (and *vice versa*) which will potentially double the cost of the coupled model.

The fluxes and ocean fields exchanged as part of the coupling depend on the coupled model setup. The initial IFS/NEMO implementation does not use a sea ice model, but rather prescribes the sea ice based on sampling of previous years of observed sea ice (for details see Molteni *et al.* (2011)). Work is ongoing to implement the version two of Louvain-la-Neuve sea ice model (LIM2) included in the NEMO modelling framework into the coupled IFS/NEMO system at ECMWF. The work was initiated by Linus Magnusson ((L. Magnusson, personal communication) based on experiences from the EC-EARTH projects[4]. The list of fields exchanged in the two configurations is given in table 1. The physical coupling will be developed (especially for the active LIM2 model) in the future, so the table is more an indication of some of the fields that will be exchanged between to IFS and NEMO (including LIM2) and will be subject to change.

[4] http::/ecearth.knmi.nl

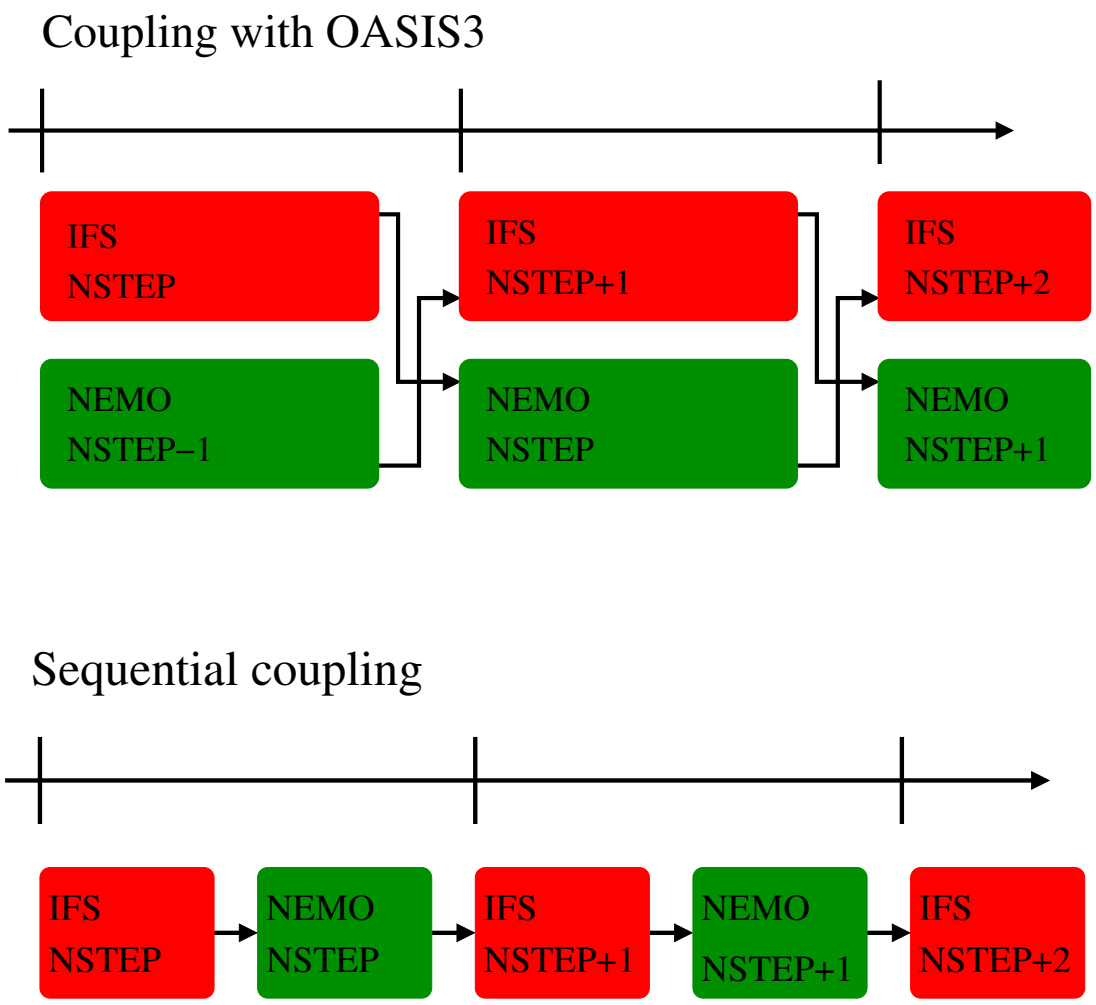

Figure 1: Coupling time axis with OASIS3 coupling (top) and sequential coupling (bottom).

	IFS/NEMO (without LIM2)	IFS/NEMO (with LIM2)
IFS to NEMO	Solar radiation	Ocean solar radiation
		Ice solar radiation
	Non-solar radiation	Ocean non-solar radiation
		Ice non-solar radiation
	Evaporation minus precipitation	Total evaporation
		Evaporation over ice
		Liquid precipitation
		Solid Precipitation
		Runoff
	Zonal wind stress	Ocean zonal stress
		Ice zonal stress
	Meridional wind stress	Ocean meridonal stress
		Ice meridonal stress
		Ice dQ/dT
NEMO to IFS	Sea surface temperature	Sea surface temperature
	Sea ice concentration	Sea ice concentration
		Sea ice temperature
		Sea ice albedo
		Sea ice thickness
		Snow thickness over sea ice
	Zonal surface current	Zonal surface current
	Meridional surface current	Meridional surface current

Table 1: Fields to be exchanged during coupling between IFS and NEMO with and without LIM2 activated.

3 Principles for the implementation of the single executable coupling of NEMO and IFS.

The IFS model already contains a two-way coupling to the WAM wave model integrated into its time stepping loop. Another alternative approach would be to code an external time stepping loop (or possible adopt the ESMF framework[5]) to call IFS and NEMO in sequence. However, the fact that the IFS model contains of the order of 2.6 million lines of code whereas the NEMO model is around 0.12 million lines of code makes it much harder to make major modifications to the IFS code compared to the NEMO code, so this option was considered impractical.

It was therefore decided to implement the same approach as for the IFS/WAM coupled system to the coupled IFS/NEMO system. In order for this to work three major developments were needed:

1. Enable the NEMO model to be called from an external driver model with a simple application programming interface consisting of:
   (a) Initialization of the NEMO model.
   (b) Advance the NEMO model one time step.
   (c) Finalize the NEMO model.
2. Development of a set of regridding routines that can interpolate data on the IFS grid to the NEMO grid (and *vice versa*), which can be called by the IFS code in full MPI parallel mode.
3. Modify the IFS model to allow it to call NEMO with forcing data from the IFS model and retrieve field data from NEMO (*e.g.* sea surface temperature and ice fields).

It was decided to allow the regridding routines access to global data structures of NEMO. Access to data (including grid information) from the IFS model in the regridding routines were allowed only as dummy arguments. By making this separation it should be possible to apply the same coupling of NEMO with a different atmospheric model.

This was also used to faciliate the development, debugging and testing of the approach since it allowed the implementation of a "dummy" atmosphere which reads the data needed by NEMO from files and simulate the time stepping of a "real" atmospheric model.

4 Parallel regridding issues

Distributed memory parallel regridding from a source grid to a target grid is a fairly complex task since it involves making sure that data required for the interpolation is communicated to where they are needed. In general the interpolation for a source grid of size n to a target grid can be expressed as:

$$t_j = \sum_{i=1}^{n} w_{j,i} s_i \tag{1}$$

where t_j are the points on the target grid, s_i are the points on the source grids and $w_{j,i}$ are the weights of each source point i contributing to the interpolated target point t_j. The weights depends on the method

[5] http://www.earthsystemmodeling.org/

of interpolation (*e.g.* bilinear, nearest neighbour, distance weighted, conservative ...). The interpolation process can be thought of as a two step process consisting of determining the weights $w_{j,i}$ and applying them using the above equation. Assuming that the grids (including any masks) are constant in time the weights do not change and can be computed once for a given interpolation scheme and set of target/source grids. It is therefore natural to separate the process in practical implementations.

In Fortran 90 equation (1) can be expressed as:

```
zdst(:)=0.0
DO n=1,num_links
  zdst(dst_address(n)) = zdst(dst_address(n)) + &
     & remap_matrix(1,n) * zsrc(src_address(n))
END DO
```

following the notation of the SCRIP documentation (Jones, 1999) with: `zdst` denoting the target grid points; `zsrc` denoting the source grid points; `remap_matrix` denoting the interpolation weights; `dst_address` denoting the addresses of the target grid memory; `src_address` denoting the addresses of the source grid memory and `num_links` is the number points contributing to the interpolation.

For the computation of weights for the single executable coupled IFS/NEMO system we use an offline process before the coupled model integration. Weights can then be read as part of the setup process of the coupled model integration. We adopt the SCRIP conventions of storing interpolation weights in netCDF format. An example of the netCDF header of such a netCDF file containing interpolation weights is given in figure 2.

Initially only very simple interpolation schemes, such as distance weighted, have been implemented. However, since the weights are read from files any interpolation software that can produce SCRIP compatible weights files (or produce something which can be converted to SCRIP compatible files) can be used.

For a single memory address space (serial or shared memory parallel) executable the meaning of `dst_address` and `src_address` is just memory addresses, but for a distributed memory parallel executable both `dst_address` and `src_address` can be considered as pointers containing both memory addresses and task numbers. In message passing interface (MPI) parallel codes, with certain decompositions of both the `zsrc` and `zdst` variables, the above interpolation can be implemented using communication of the `zsrc` data from all tasks that contributed to interpolation of a specific element `zdst(i)` to the task which holds the element. Assuming that the decomposition is static during execution it is possible to compute what data needs to be sent to which MPI task number as part of the initialization phase.

In the present approach we use a `mpi_alltoallv` call to exchange data for interpolation. The message passing communication pattern is computed at the beginning of the run. During this phase local indicies for each unique source point needed on the target side for each task is packed into a buffer which can be used for packing of data to be sent during the interpolation. Similarly the local addresses (equivalent of `src_address`) on the receive side are computed to perform the interpolation based on the output of the `mpi_alltoallv` call. Care has been taken to ensure that the order of summation is invariant for different number of tasks in order to make the results binary identical for runs with different number of tasks.

Once the information about sending buffer packing and local source addresses is available the parallel

interpolation becomes straighforward as can be seen from the following code snippet:

```
! Pack the sending buffer
DO i=1,pinfo%nsendtot
   zsend(i)=zsrc(pinfo%send_address(i))
ENDDO
! Do the message passing
CALL mpi_alltoallv(&
   & zsend,pinfo%nsend(0:nproc-1),&
   & pinfo%nsdisp(0:nproc-1),mpi_double_precision, &
   & zrecv,pinfo%nrecv(0:nproc-1), &
   & pinfo%nrdisp(0:nproc-1),mpi_double_precision, &
   & mpi_comm,istatus)
! Do the interpolation
zdst(:)=0.0
DO i=1,pinfo%num_links
   zdst(pinfo%dst_address(i)) = zdst(pinfo%dst_address(i)) + &
     & pinfo%remap_matrix(1,i)*zrecv(pinfo%src_address(i))
END DO
```

In the above code the `pinfo` variable is a structure that holds all the needed information for interpolation. Multiple different `pinfo` variables can be used to define different interpolation between different grids.

5 Implementation of an inline coupling interface in the NEMO model

In the present implementation the parallel regridding routines discussed in the previous section are model independent, but the actual use of these routines is done in an inline coupling interface, which is coded with access to NEMO Fortran-90 module data making this interface NEMO specific.

The NEMO model already has a well defined routine that advances the model one time step, which makes it fairly straighforward to be able to call it from an external model.

New initialization and finalization routines had to be coded (based on existing ones) mostly to deal with MPI library being initialized and finalized in the atmospheric model before it calls the NEMO model. A new option in the surface boundary condition (SBC) code was introduced in order for NEMO not to read fluxes from files, but allow the atmosphere model to directly update the NEMO variables via the coupling interface.

After initialization of NEMO the coupling interface needs to be initialized. In the present code we assume that the atmosphere model runs on an unstaggered grid so all atmosphere variables are defined at the same latitudes and longitudes for a given point. Since NEMO runs on a Arakawa C-grid[6] we need to setup interpolation from atmosphere points to NEMO T, U and V points. The surface currents are passed back to the atmosphere model but, just as in the current OASIS3 based NEMO coupling interface, the surface currents are destaggered from U, V points to T points and rotated from grid I, J directions to east, north directions before they are interpolated. This eliminates the need for the interpolation of NEMO U and V points to atmosphere points so that we only need to interpolate from NEMO T points to atmosphere

[6]For LIM2 the fluxes are not exhanged on the native B-grid of LIM2 but on the C-grid.

points. Therefore, a total of four different interpolation weights needs to be provided as files in the SCRIP format described above.

The atmosphere grid information is provided both directly by arguments to the coupling initialization routine or indirectly via the SCRIP weights files. As a safety measure the masks of the NEMO model and the atmosphere model are checked against the grid data in the SCRIP weight data files to ensure that they are consistent. However, there is no assumption in the coupling interface that the atmosphere is running on reduced Gaussian grid and there is no use of any modules from the IFS either, so in principle any atmospheric model could be used with this interface as long as it can provide data on an unstaggered grid. Extension to staggered grid should be straightforward if this becomes necessary.

The full implemented interface as seen from the atmosphere model's point of view is given in figure 3. For simplicity it was decided to put the logic of controlling the LIM2 sea ice model in the atmospheric model code and have the atmosphere model call the relevant routines (*_*lim2*_* or not) based on the atmospheric sea ice switch.

Extension to more variables in the coupling routines are trivial since it just involves adding more arguments to the interface call and making the appropriate links to the corresponding internal NEMO data.

As of release 3.3.1 of NEMO the model uses dynamic memory allocation and the same executable can run with any domain decomposition for a given grid. Previously the NEMO model used static memory and each domain decomposition needed it's own executable. However even with version 3.3.1 of NEMO there is some grid dependent compilation controlled by preprocessing. For the coupled system this means that we can create a library which contains both the NEMO model and the coupling interface which can be used for any decomposition of the coupled model system. However this library will be for a specific NEMO grid only and can not be used for any NEMO grid. If we want to get around this limitation significant changes to the NEMO model are needed, so this was not considered in the initial implementation.

Before inclusion into the full IFS model the above interface was tested with a "dummy" atmosphere model that reads atmospheric data in GRIB format (*e.g.* ERA interim data) and applys the procedure described in section 3 to simulate a coupled system. This "dummy" atmosphere model reads all input GRIB data on the "master" MPI task but is able to call NEMO in full MPI parallel mode. It was not meant to be a useful tool for any application, but it did assist in the debugging and testing of the coupling interface. This tool only runs without the LIM2 model, since the coupling to LIM2 in forced mode is done with the bulk formulae forcing (Madec, 2008) and not via fluxes, thus making the forced model system quite different from the coupled model system. With prescribed sea ice the forced NEMO system can use the same fluxes as the coupled model making it easy to simulate the coupled system with a dummy atmosphere.

6 Modifications to the IFS model

In the current OASIS3 based IFS-NEMO coupled system the initialization of the MPI library is done within the OASIS3 code. In the single executable coupled system the IFS initializes MPI directly and passes the MPI communicator to NEMO for the initialization of NEMO using either **nemogcmcoup_init** or **nemogcmcoup_lim2_init** (in the case of an active sea ice model) in figure 3.

Initialization of the coupled model is quite similar to the existing OASIS3 interface in the sense that the grid information from the IFS is passed to the coupling interface via the **nemogcmcoup_coupinit** call (in **cnt3**) just before the main time stepping loop (in **cnt4**).

```
netcdf remap_Ttogauss {
dimensions:
        src_grid_size = 105704 ;
        dst_grid_size = 213988 ;
        src_grid_corners = 4 ;
        dst_grid_corners = 4 ;
        src_grid_rank = 2 ;
        dst_grid_rank = 1 ;
        num_links = 607928 ;
        num_wgts = 1 ;
variables:
        int src_grid_dims(src_grid_rank) ;
        int dst_grid_dims(dst_grid_rank) ;
        double src_grid_center_lat(src_grid_size) ;
                src_grid_center_lat:units = "degrees" ;
        double dst_grid_center_lat(dst_grid_size) ;
                dst_grid_center_lat:units = "degrees" ;
        double src_grid_center_lon(src_grid_size) ;
                src_grid_center_lon:units = "degrees" ;
        double dst_grid_center_lon(dst_grid_size) ;
                dst_grid_center_lon:units = "degrees" ;
        double src_grid_corner_lat(src_grid_size, src_grid_corners) ;
                src_grid_corner_lat:units = "degrees" ;
        double src_grid_corner_lon(src_grid_size, src_grid_corners) ;
                src_grid_corner_lon:units = "degrees" ;
        double dst_grid_corner_lat(dst_grid_size, dst_grid_corners) ;
                dst_grid_corner_lat:units = "degrees" ;
        double dst_grid_corner_lon(dst_grid_size, dst_grid_corners) ;
                dst_grid_corner_lon:units = "degrees" ;
        int src_grid_imask(src_grid_size) ;
                src_grid_imask:units = "unitless" ;
        int dst_grid_imask(dst_grid_size) ;
                dst_grid_imask:units = "unitless" ;
        double src_grid_area(src_grid_size) ;
                src_grid_area:units = "" ;
        double dst_grid_area(dst_grid_size) ;
                dst_grid_area:units = "" ;
        double src_grid_frac(src_grid_size) ;
                src_grid_frac:units = "" ;
        double dst_grid_frac(dst_grid_size) ;
                dst_grid_frac:units = "" ;
        int src_address(num_links) ;
        int dst_address(num_links) ;
        double remap_matrix(num_links, num_wgts) ;

// global attributes:
                :title = "" ;
                :normalization = "" ;
                :map_method = "" ;
                :history = "" ;
                :conventions = "" ;
                :dest_grid = "3991_2_mask.nc" ;
                :source_grid = "orca1_z42_mask_T_L1.nc" ;
}
```

Figure 2: Example of SCRIP like netCDF files containing interpolation weights.

Initialization routines: Routines which needs to be called before the time step loop.

nemogcmcoup_init: Initialize the NEMO model and open NEMO output files (prescribed sea ice case).

nemogcmcoup_lim2_init: Initialize the NEMO model and open NEMO output files (LIM2 case).

nemogcmcoup_coupinit: Initialize the coupling interface. Needs to be called after **nemogcmcoup_init** since it needs access to the NEMO grid information.

Time step routines and coupling: Routines to be called as part of the time stepping loop of the host model to provide fluxes to NEMO, advance NEMO and retrieve data from NEMO:

nemogcmcoup_update: Update NEMO fluxes (prescribed sea ice case).

nemogcmcoup_lim2_update: Update NEMO fluxes (LIM2 case).

nemogcmcoup_step: Advance the NEMO model one time step.

nemogcmcoup_get: Retrival of ocean data from the NEMO model (prescribed sea ice case).

nemogcmcoup_lim2_get: Retrival of ocean data from the NEMO model (LIM2 case).

Termination routines: Routines to cleanly exit NEMO.

nemogcmcoup_final: Close NEMO files and terminate NEMO execution.

Figure 3: Interface routines for inline coupling to the NEMO model. The fluxes exchanged in the update/get routines are listed in table 1.

During IFS time stepping (**cnt4**) the following actions happen:

1. The IFS code runs n time steps (specified by the existing **NFRCO** namelist parameters). The fluxes are accumulated using the existing routines used by the OASIS3 interfaces.
2. The fluxes are normalized with n multiplied by the time step to get the data the NEMO model needs and passed to NEMO via the **nemogcmcoup_update** or **nemogcmcoup_lim2_update** routines.
3. The NEMO model runs for m time steps by calling **nemogcmcoup_step** m times. The parameter m is determined by the coupling frequency (n) and the time steps of the NEMO model and the IFS as follows:
$$m = n\frac{\Delta t_{\mathbf{IFS}}}{\Delta t_{\mathbf{NEMO}}} \tag{2}$$
with the condition that m has to be an integer (if not the code aborts). In the above equation $\Delta t_{\mathbf{IFS}}$ and $\Delta t_{\mathbf{NEMO}}$ is the time step of the IFS and NEMO models respectively.
4. The existing routine (**updclie**) for updating the sea surface temperature and ice fields in the IFS calls the **nemogcmcoup_get** or **nemogcmcoup_lim2_get** routine to get updated fields from the NEMO model.

An important difference of this new implementation compared to the OASIS3 setup is that when using OASIS3 the SST and ice fields are lagged with one coupling step (as already mentioned in section 2), but in the single executable coupled setup they are not. This is due to the fact that in the single executable coupled setup the IFS and NEMO runs sequentially whereas in the OASIS3 setup they run in parallel.

Since most of the coupling logic and control of NEMO is done within the coupling layer the amount of changes to the IFS model is actually quite small. The main changes is in the time stepping loop, but even those are fairly moderate in size.

Since the changes are quite small it is easy to code a set of "dummy" routines which can take the place of the routines mentioned in figure 3 to avoid the need to link with the full NEMO library if no coupling to NEMO is desired. This flexibility makes it possible to include all the routines needed for coupling with NEMO in the "main" IFS library without any dependencies on either coupling or NEMO code since it can be decided at link time if the executable should be able to run coupled with the NEMO model or not. Even if the full NEMO library is included in the linking it is still possible to switch off the NEMO model when actually running the executable.

All modifications to the IFS were initially done in a branch of CY37R2, but has since then been merged into later cycles. An implementation of the single executable coupled interface without the LIM2 modifications and based on v3.3.1 of NEMO has been included as a passive change in CY38R1 of the IFS model. Branches of the coupled system with support for LIM2 exists for CY37R3 and CY38R1 based on version 3.4 of NEMO.

7 Performance results and outstanding issues

To test the performance a standalone simplified scripting system outside the normal prepIFS/SMS scripting system was used. In this setup the IFS model does not use the field database (FDB) software for output, so it is expected that the performance of IFS is not optimum. However only a limited amount of output were produced in these runs. In the following we will only present results without LIM2 being active.

7.1 Comparision with the OASIS3 setup

In current applications typical resolutions to be used at ECMWF for the coupled IFS-NEMO system is T255L91 (seasonal forecasting system 4) and T319L62 (EPS leg B and monthly forecasting) with the coupled system running on a single node.

To compare the technical performance of the current IFS-OASIS3-NEMO coupled system to the proposed single executable system a set of comparisons were made with IFS running at resolutions of T255L62 and T399L62, which, although not exactly the operational resolutions, is sufficiently close that they can provide useful information.

Table 2 presents different run times for different configurations for one to three nodes of both the current IFS-OASIS3-NEMO system and the proposed system. The same IFS model cycle (CY37R2) has been used for all runs, but since the local modifications for the OASIS3 coupling have not yet been implemented in NEMO version 3.3.1 the NEMO version used for the IFS-OASIS3-NEMO was version 3.0 rather than version 3.3.1. Experiences shows that the NEMO version 3.3.1 is more expensive than version 3.0, but it is not believed that the conclusions made here are sensitive to this difference. Also in the table the uncoupled IFS timings are given as reference. All timings are wall clock time including I/O (to files rather than FDB). An advantage with the single executables (which will be discussed in more detail below) is that it is straightforward to use OpenMP shared memory threading within the IFS model even when running the coupled model.

The conclusion is that the fastest single executable run for a given number of nodes runs as fast as the OASIS3 based run for T255L62 runs and slightly faster for the higher resolution T399L62 runs. Since there has not yet been any attempt to optimize the single executable system this is quite encouraging.

It is also possible that the load balance of some of the OASIS3 runs could be better leading to slightly shorter run times for these runs. It is however not believed that this will change the conclusion that the single executable is similar or better in run time performance compared to the operational IFS-OASIS3-NEMO system.

7.2 Performance and scalability of the IFS and NEMO models in uncoupled mode

If either of component models does not scale with number of processors then the scalability of single executable approach will be limited. It is therefore important to establish the scalability of the model compenents in uncoupled mode before discussing the performance of the coupled system in details.

The IFS model is known to scale quite well, so we expect that the NEMO model is the limiting factor. At the moment ECMWF only uses a fairly low resolution version of the NEMO model (ORCA1 with 42 levels). This configuration uses a 362 by 272 gridpoints grid with a resolution of approximately 1.0 degree while in the equatorial region the resolution increases, in latitude only, to 0.3 degrees. The physics of ocean models are in general much less computationally costly than atmosphere models, so the vertical physics of an ocean model, which is embarrassingly parallel, accounts for much less of the total run time compared to the physics of an atmosphere model.

Figure 4 shows the wall clock time (top plot) and speed-up (bottom plot) of the IFS for T255L62 and T399L62 configurations and of NEMO for the ORCA1L42 configuration for a model integration of one month as function of number of CPU cores. These runs were done on the IBM Power6 machine at ECMWF (Storer and Weger, 2008) without simultaneous multithreading (SMT) and without use of multiple OpenMP threads, so the maximum number of nodes used is 8 since there is 32 cores per node.

Resolutions	Nodes	Coupling	Wall clock (sec)	Remarks
T255L62	1	NONE	3632	
T255L62	1	NONE	2372	2 threads/core
T255L62	1	NONE	2423	2 MPI tasks/core
T255L62/ORCA1L42	1	INLINE	4116	
T255L62/ORCA1L42	1	INLINE	2927	2 threads/core
T255L62/ORCA1L42	1	INLINE	2866	2 MPI tasks/core
T255L62/ORCA1L42	1	OASIS3	2961	N_{IFS}=53, N_{NEMO}=6
T399L62	1	NONE	11852	
T399L62	1	NONE	7596	2 threads/core
T399L62	1	NONE	8133	2 MPI tasks/core
T399L62/ORCA1L42	1	INLINE	11563	
T399L62/ORCA1L42	1	INLINE	8160	2 threads/core
T399L62/ORCA1L42	1	INLINE	MEMKILL	2 MPI tasks/core
T399L62/ORCA1L42	1	OASIS3	9640	N_{IFS}=53, N_{NEMO}=6
T255L62	2	NONE	1894	
T255L62	2	NONE	1280	2 threads/core
T255L62	2	NONE	1340	2 MPI tasks/core
T255L62/ORCA1L42	2	INLINE	2201	
T255L62/ORCA1L42	2	INLINE	1618	2 threads/core
T255L62/ORCA1L42	2	INLINE	1762	2 MPI tasks/core
T255L62/ORCA1L42	2	OASIS3	1616	N_{IFS}=106, N_{NEMO}=12
T399L62	2	NONE	6164	
T399L62	2	NONE	4000	2 threads/core
T399L62	2	NONE	4172	2 MPI tasks/core
T399L62/ORCA1L42	2	INLINE	6593	
T399L62/ORCA1L42	2	INLINE	4383	2 threads/core
T399L62/ORCA1L42	2	INLINE	4722	2 MPI tasks/core
T399L62/ORCA1L42	2	OASIS3	5050	N_{IFS}=106, N_{NEMO}=12
T255L62	3	NONE	1308	
T255L62	3	NONE	895	2 threads/core
T255L62	3	NONE	999	2 MPI tasks/core
T255L62/ORCA1L42	3	INLINE	1590	
T255L62/ORCA1L42	3	INLINE	1195	2 threads/core
T255L62/ORCA1L42	3	INLINE	1418	2 MPI tasks/core
T255L62/ORCA1L42	3	OASIS3	1198	N_{IFS}=159, N_{NEMO}=18
T399L62	3	NONE	4200	
T399L62	3	NONE	2761	2 threads/core
T399L62	3	NONE	2945	2 MPI tasks/core
T399L62/ORCA1L42	3	INLINE	4531	
T399L62/ORCA1L42	3	INLINE	3092	2 threads/core
T399L62/ORCA1L42	3	INLINE	3565	2 MPI tasks/core
T399L62/ORCA1L42	3	OASIS3	3625	N_{IFS}=159, N_{NEMO}=18

Table 2: Run time for 744 hours of integration on a single node (top), two nodes (middle) and 3 nodes (bottom) for different coupled IFS/NEMO systems. MEMKILL means that this run needed more memory than available (50000MB) on a single node so it crashed.

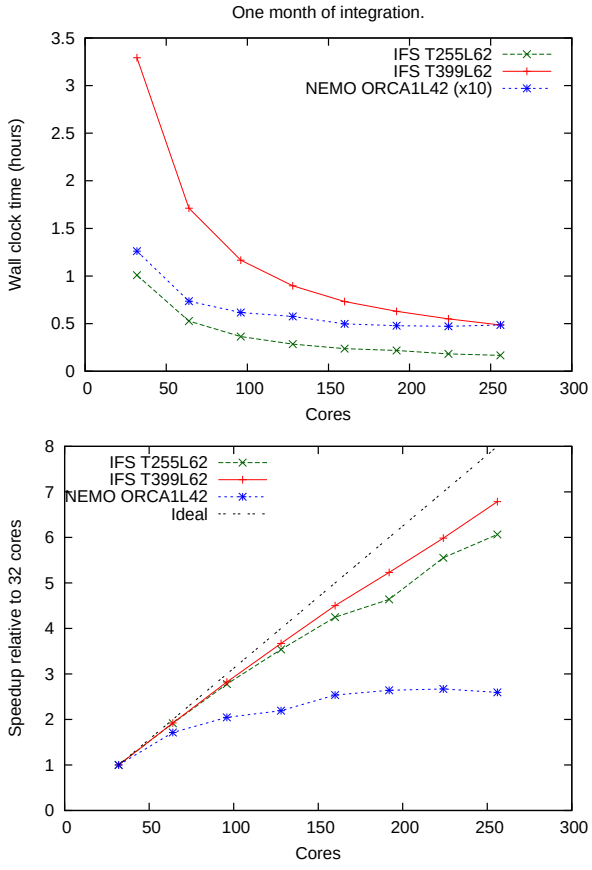

Figure 4: Wall clock time and speedup for the IFS model for a one month model integration for two different resolutions compared with NEMO for the ORCA1 configuration. Be aware that the NEMO wall clock time has been scaled with a factor of 10.

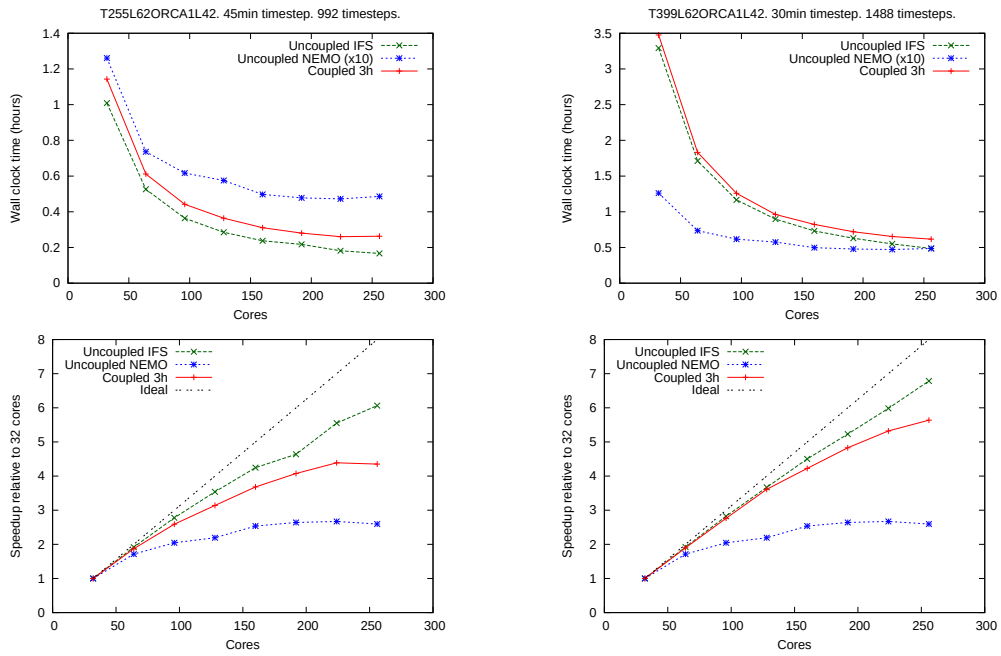

Figure 5: Wall clock time and speedup for the coupled IFS-NEMO model for a one month model integration. The IFS resolution was T255L62 (left) and T399L62 (right). I both cases the NEMO resolution were ORCA1L42.

Two things are evident from the figure; the NEMO model is very fast compared to the IFS model at the resolutions considered but it does not scale very well. The number of gridpoints for a single 3D field of the ORCA1 configuration is 4135488 which is compariable to the T255L62 configuration (5507956) but much less than the T399L62 configuration (13267256).

The NEMO timings shown in figure 4 were done using a 2D horizontal decomposition of the ORCA1 grid. It is also possible to use a 1D decomposition, but this makes the scalability of NEMO even worse. In the following all results are for a 2D decomposition of the ORCA1 grid.

7.3 Performance and scalability of the single executable coupled IFS/NEMO system

Figure 5 shows the wall clock time (top plots) and speed-up (bottom plots) as function of number of CPU cores for two different IFS resolutions. Just like figure 4 these runs were done on the IBM Power6 machine at ECMWF without SMT and without use of multiple OpenMP threads.

It is not surprising that the coupled IFS/NEMO model does not scale as well as the uncoupled IFS model but better than the uncoupled NEMO model. Looking at the logfiles of the coupled runs it is seen that the initialization phase of the coupling is fairly expensive (up to 170 seconds for high processor counts) at the moment. During this initialization phase the communication patterns during the coupled integration are established and it seems like this part of the code needs to be optimized more. The reason why the T399L62 runs scale better than the T255L62 runs is simply that the amount of work per task relative to time spent in communication increases with increased resolution (also known as weak scaling). Since

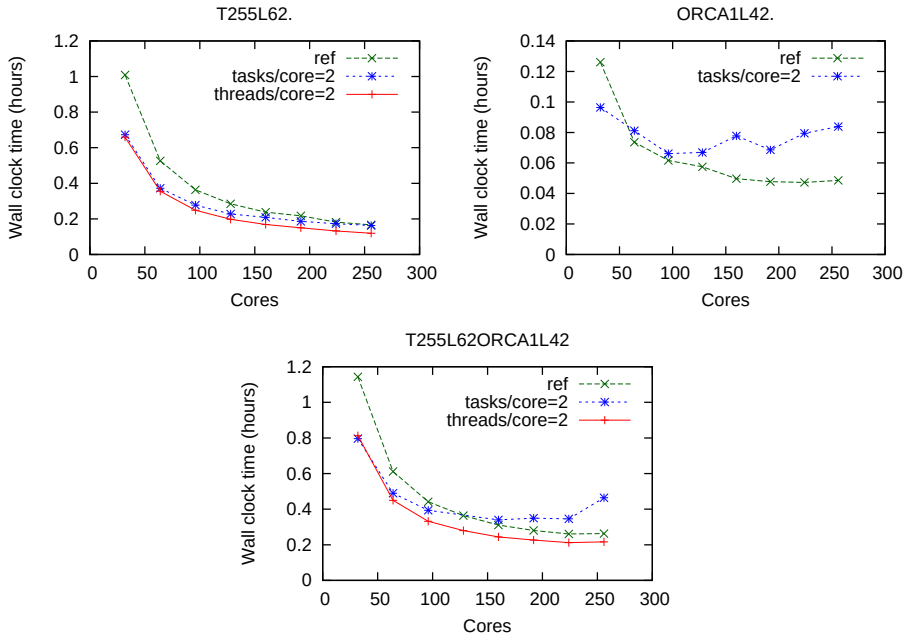

Figure 6: Wall clock time for the uncoupled IFS at T255L62 (top left), the uncoupled NEMO at ORCA1L42 and the coupled model system at the same resolutuions for either no symmetric multiprocessing (SMT), two MPI tasks per core or two OpenMP threads per core as function of total number of cores.

the cost of the NEMO model in the coupled runs is the same for both resolutions the net effect is that the coupled T399L62 runs scales better than the coupled T255L62 runs.

Another aspect to consider is that the regridding of data implies that all MPI tasks needs to be synchronized at this point in time. This means that the regridding will always introduce a parallel overhead and the only thing which can be done is to try to minimize this overhead.

An important property of the single executable system is that it produces bit identical results with different number of MPI tasks for a given configuration. This property helps debugging and optimization, since one is able to verify the scientific performance for a given number of tasks and then run with a different number of tasks knowing that any difference in the results originates from coding errors. For completeness it should be stressed that this is also true for the OASIS3 based coupled system.

In these runs a very simple nearest neighbour interpolation scheme was used, with computed weights based on inverse distances between source point and targets points. For real applications this scheme should be replaced with something more appropriate, but for initial technical testing this scheme was deemed to be sufficient.

7.4 Using symmetric multi threading (SMT) and OpenMP

The runs presented in figure 5 do not make optimal use of the IBM nodes since they do not use the simultaneous multithreading (SMT) feature of the IBM Power6 processor, which allows two threads to

run on the same processor as virtual processors. Experience at ECMWF shows that running with shared memory parallelism (OpenMP) on the SMT works well for the IFS model, but the present version of NEMO does not have support for OpenMP parallelism.

To try to use SMT two sets of runs were made. One set with running two MPI tasks per physical core and another set with two OpenMP threads per core. In the latter case the NEMO part of the execution runs with a single thread. These set of runs were only done for the T255 IFS resolution in order to save computer time.

The wall clock time and scalability for the runs with two MPI tasks per core are shown in figure 6. Both uncoupled IFS/NEMO runs and coupled runs are shown. It is clear that the NEMO model does not work well running with 2 MPI tasks per core. Even though the IFS model seems to work reasonably well the coupled system suffers from the performance of NEMO.

It is quite possible that running with 64 MPI tasks on a 32 core node is too much and running with somewhere between 32 and 64 would be better, but this has not been verified.

The wall clock time and scalability for the runs with two OpenMP threads per core are shown also in figure 6. In this case the NEMO results are the same as in figure 5 since it is single threaded. The IFS clearly works very well in this configuration with the lowest run times for the same number of cores (compare the top plots of figures 5 and 6). Even though the scalability of the coupled system suffers from NEMO not having OpenMP parallelism running with 2 OpenMP threads per core is still fastest way of running the coupled system, which is really what is the most important factor. This result also indicates that there is some scope for improving the coupled system by introducing a second level of parallelism in NEMO with OpenMP.

7.5 Beyond current operational resolutions

One of the main reasons for implementing the single executable coupling was that it was believed that it would improve the performance at high resolution. A suitable test case was chosen to be the current resolution of the first leg (which currently runs uncoupled) of the EPS system for the atmospheric model (T639 with 62 levels) and a 0.25 degree ORCA configuration (ORCA025) with 75 levels. Future configurations of the EPS system will have higher vertical resolution for the atmospheric part, but the results should still be able to provide us with some guidelines for the cost of doing coupled modelling at high resolutions.

The wall clock time (top plot) and speedup (bottom plot) from a 1 month runs as function of number of cores for this system is shown on figure 7 for the uncoupled IFS T639L62 system, the NEMO ORCA025L75 system and the coupled system with two different coupling frequencies of either 20 min (or every time step) or 3 hours (or every 9th time step). In all runs on the figure the output was kept to a minimum. The results are quite similar to the low resolution runs. For high processor counts the IFS model scales very well, but the NEMO model fails to scale much beyond 600 cores. For the coupling every time step the *all to all* communication between tasks hurts the performance, but for 3 hour coupling the scalability of the coupled system is still reasonable up to around 800 processors.

It has already been established (section 7.4) that the coupled model benefits in terms of run time on the IBM Power6 by using SMT and allowing the IFS to use 2 OpenMP threads per core. A similar study was done with the high resolution test case with the results shown in table 3 for a fixed number number of cores, but different number of threads. It is clear that the IFS works well with OpenMP, but that using more than 2 OpenMP threads for the coupled system is detrimental for the performance due to the lack

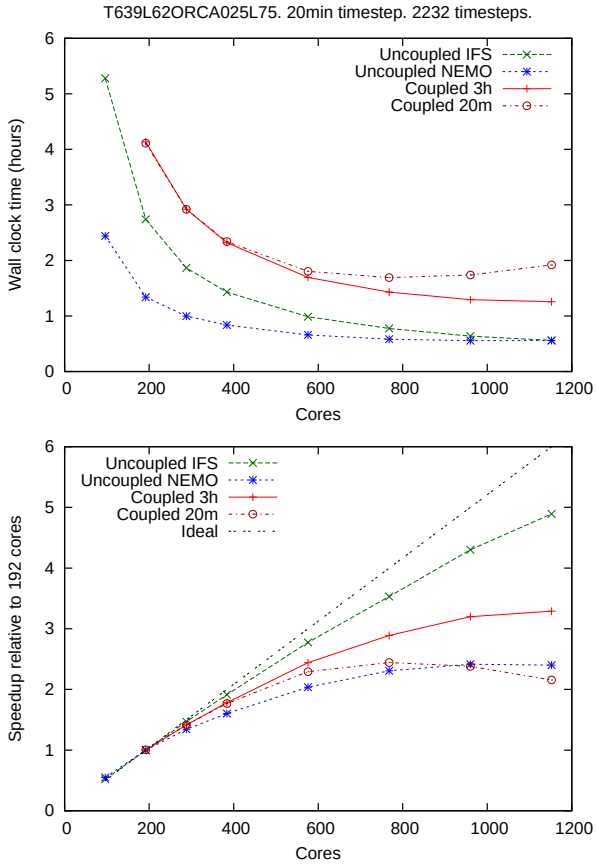

Figure 7: Wall clock time and speedup for the coupled IFS-NEMO model for a one month model integration. The IFS resolution was T639L62 and the NEMO resolution was ORCA025L75.

Configuration	Cores	Tasks	Threads	SMT	Wall clock (sec)
T639L62	576	576	1	No	3560
T639L62	576	576	2	Yes	2600
T639L62	576	288	4	Yes	2576
T639L62	576	144	8	Yes	2618
T639L62/ORCA025L75	576	576	1	No	6148
T639L62/ORCA025L75	576	576	2	Yes	5312
T639L62/ORCA025L75	576	288	4	Yes	6437
T639L62/ORCA025L75	576	144	8	Yes	8790

Table 3: Wall clock for the IFS only (first four rows) and the IFS/NEMO single executable coupled system (last four rows) for different number of threads and SMT configurations running on the IBM Power6.

Configuration	Cores	Tasks	Threads	Time/day (sec)
T639L62/ORCA025L75	120	120	1	805
T639L62/ORCA025L75	240	240	1	432
T639L62/ORCA025L75	480	480	1	297
T639L62/ORCA025L75	720	720	1	232
T639L62/ORCA025L75	120	60	2	782
T639L62/ORCA025L75	240	120	2	418
T639L62/ORCA025L75	480	240	2	235
T639L62/ORCA025L75	720	360	2	181
T639L62/ORCA025L75	960	480	2	160

Table 4: Time per day for the IFS/NEMO single executable coupled system for different of number tasks and threads on Intel Westmere.

of OpenMP in the NEMO model.

We also had the opportunity to run the coupled model on the Janus cluster at the Colorado University[7]. This cluster has two 2.8-GHz Intel Westmere processors per node with 6 cores per processor. The results are shown in table 4 for the coupled system only. It is interesting that even though this machine do not have SMT it is still benificial to use two OpenMP threads per tasks meaning that effectively the amount of tasks (both totally and per node) running NEMO is halved. We can speculate that the reason for this is that NEMO could be memory bandwidth limited, so halving the number of NEMO tasks per node doubles the available memory bandwidth. An alternative explaination could be that the communication decreases leading to better scalability, which compensates for the lack of available CPU's to NEMO. We clearly need to better understand the memory access patterns of NEMO in the coupled system (*e.g* by looking at cache misses).

7.6 Communication issues in the single executable coupled system

To optimize the communication as much as possible we ideally want the domain decomposition of the atmospheric and ocean grids to overlap as much as possible to maximize the local memory to memory copying and limit the amount of data needed to be communicated. Unfortunately the domain composition of the IFS and the NEMO models makes this quite difficult as can be seen from figure 8, which shows the domain number (ranging from 1 to total number of MPI tasks of 16) for the IFS at T255L62 (top figure) and the NEMO model at ORCA1L42 (bottom figure) for all ocean points of the two grids. It is clear that the domain decompositions of IFS and NEMO are not ideal for minimizing the amount of communications during the regridding, since there is very little overlap between the grid points of the different tasks on the two grids.

To quantify the communication patterns the communication statistics for a single regridding between sets of source and target grids for different number of tasks is given in table 5. In the table the column *max p2p* is the maximum number of point to point exchanges between any tasks, the column *max length* is the maximum message length (in elements) of any of point to point exchanges, the column *mean p2p* is the mean number of point to point exchanges on all tasks and the *mean length* is the mean number of elements in all the point to point exchanges.

From the table it is clear that the total number of exchanges increases (about the same number of exhanges

[7] https://www2.cisl.ucar.edu/resources/janus-cluster

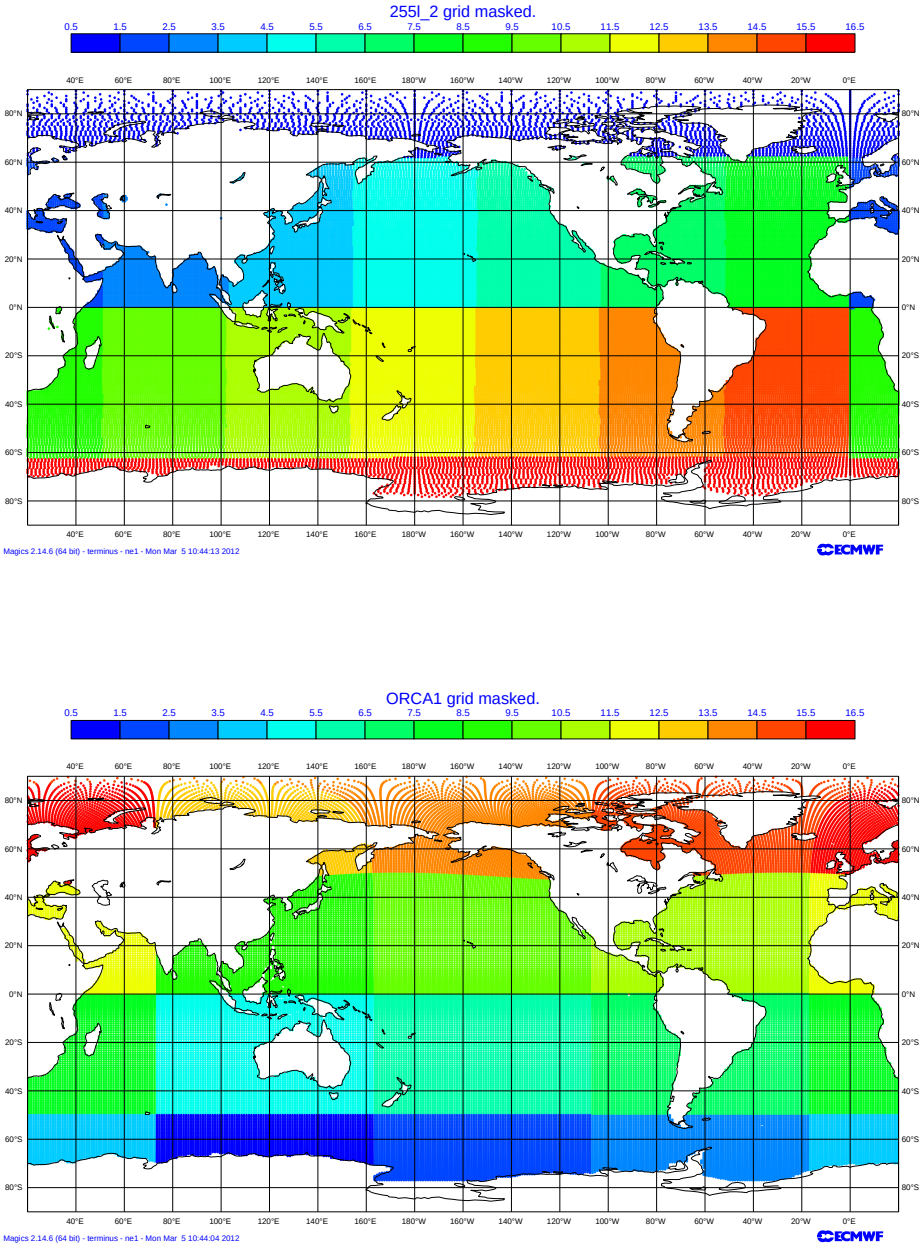

Figure 8: Ocean grid points of the IFS T255L62 Gaussian grid (top) and the NEMO ORCA1 grid (bottom). The colour bar indicates which task number (starting from 1) individual grid point belongs to for a total number of tasks of 16.

Source	Target	Total tasks	Max p2p	Max length	Mean p2p	Mean length
255L62	ORCA1L42	16	6	4492	4.4	911
ORCA1L42	255L62	16	6	3558	4.2	977
255L62	ORCA1L42	32	8	2685	4.0	508
ORCA1L42	255L62	32	6	2085	3.8	553
639L62	ORCA025L75	192	6	2261	3.3	643
ORCA025L75	639L62	192	9	5170	3.4	1204
639L62	ORCA025L75	384	9	1412	3.2	338
ORCA025L75	639L62	384	10	3120	3.2	639

Table 5: Communications statistics for a single regridding between sets of source and target grids for different number of tasks. The max p2p is the maximum number of exchanges between on any tasks, the max length is the maximum number of elements in any exchange, the mean p2p is the mean number of exchanges on all tasks and the mean length is the mean number of elements in all the exchanges.

per task, but the maximum number increases) as the processor count increases. If the communication cost is dominated by latency this means communications overhead will increase as function of number of tasks meaning that it will eventually limit the scalability. Similarly, although doubling the number of tasks decreases the message length it does not half it so, if bandwidth is the limitation, then the overhead in communication will increase with number of processors as well.

As discussed in section 4 then in actual implementation a collective communication is used rather that point to point, but that does not change the conclusion that there is a fundamental limitation in the scalability of the coupled system due to the exchange of data for the regridding.

8 Conclusions and outlook

The technical implementation and performance of the initial implemenation of a single executable IFS/NEMO coupled system has been discussed. The system has been implemented starting from IFS CY37R2 and NEMO version 3.3.1 and carried forward into later cycles, but most of the developments have been done in the interface layer which couples the models together and few changes are made to the models themself.

In the single executable coupled system all information (grid and fluxes) from the atmospheric model is either passed as arguments to the coupling interface or read from the interpolation weights data files so it is possible to apply the approach to a different atmospheric model; the code, therefore, could be shared with other institutions. Also the parallel regridding used by the coupling interface is very generic in nature and can be used in other applications. We are already considering if this regridding can be used for simplifying the NEMOVAR assimilation system.

Technically the initial implementation works reasonably well and it could, in principle, replace the current OASIS3 based coupled system since the run times are similar or better. Naturally this assumes that the scientific performance is as good as the existing coupled model for all systems, so this has to be verified first. Initial tests done with the EPS system indicates a similar scientific performance as the existing coupled IFS-OASIS3-NEMO model (not shown), but more work is needed for the scientific evaluation of the system.

Beyond the performance benefit seen there are other advantages for the approach compared to the existing

OASIS3 based coupled model. The execution on of the coupled model is simpler since we do not need to launch a multiple program multiple data (MPMD) collection of different executables. We also believe that it will be easier to implement a fully coupled ocean/atmosphere assimilation system with the new system.

However, a major disadvantage is that we would not be able to directly use coupling developments based on OASIS couplers made by external partners. However, it is believed that scientific developments in coupled modelling done by the community can be relatively easily implemented in a single executable coupled system even if the coupling interface used is different to the one used for the developments.

The single executable system has been demonstrated to work reasonably well for a fairly high resolution coupled system consisting of a T639L62 atmospheric resolution coupled to an ORCA025L75 ocean resolution, so we are confident that this system can form the basis for the coupled model at ECMWF in the future. To put performance numbers for the high resolution runs in perspective we can estimate how much computer power will be required to do a monthly forecast with 51 ensemble members on one of the current IBM Power6 clusters at ECMWF. Assuming that the run time is 90 minutes (from table 3) on 18 nodes combined with the fact that there is 286 nodes per cluster this means that in principle it would be possible to run this system in about 6 hours occupying close to one full cluster during that period.

We have identified a potential improvement of the single executable coupled system if the domain composition of the grids of the atmosphere and the ocean were more compatible. However this might be difficult to do in practice and involve large number of changes in one of the models. It is also clear that the use of OpenMP in NEMO needs to be explored to improve the performance.

References

Jones PW. 1999. First- and second-order conservative remapping schemes for grids in spherical coordinates. *Mon. Wea. Rev.* **127**: 2204–2210.

Madec G. 2008. Nemo ocean engine. Note du Pole de modélisation 27, Institut Pierre-Simon Laplace (IPSL), Paris, France.

Molteni F, Stockdale T, Balmaseda M, Balsamo G, Buizza R, Ferranti L, Magnusson L, Mogensen K, Palmer T, Vitart F. 2011. The new ECMWF seasonal forecast system (System 4). Tech. Memo. 656, ECMWF.

Stockdale T, Anderson DLT, Balmaseda MA, Doblas-Reyes F, Ferranti L, Mogensen K, Palmer T, Molteni F, Vitart F. 2011. ECMWF seasonal forecast System 3 and its prediction of sea surface temperature. *Climate Dyn.* **37**: 455–471. 10.1007/s00382-010-0947-3.

Storer N, Weger I. 2008. ECMWF's Replacement High Performance Computing Facility 2009–2013. *ECMWF Newsletter.* **115**: 44–49.

Terray L, Valcke S, Piacentini A. 1998. Oasis 2.2 ocean atmosphere sea ice soil, user's guide and reference manual. Technical Report TR/CMGC/98-05, CERFACS, Toulouse, France.

Valcke S. 2006. Oasis3 user guide (prism_2-5). PRISM Support Initiative Report 3, 64pp.

Vitart F, Buizza R, Balmaseda MA, Balsamo G, Bidlot JR, Bonet A, Fuentes M, Hofstadler A, Molteni F, Palmer T. 2008. The new VAREPS-monthly forecasting system: a first step towards seamless prediction. *Q. J. R. Meteorol. Soc.* **134**: 1789–1799.

Wolff J, E MR, S L. 1997. The Hamburg Ocean Primitive Equation model. Technical Report 13, Deutsches Klimarechenzentrum, Hamburg, Germany. 98.