ECMWF Feature article

from Newsletter Number 114 – Winter 2007/08

METEOROLOGY

A new partitioning approach for ECMWF's Integrated Forecast System

www.ecmwf.int/en/about/news-centre/media-resources

doi:10.21957/2m1i7n0v3a

This article appeared in the Meteorology section of ECMWF Newsletter No. 114 – Winter 2007/08, pp.17-23.

A new partitioning approach for ECMWF's Integrated Forecast System

George Mozdzynski

Since the mid-1990s the Integrated Forecast System (IFS) at ECMWF has used a two-dimensional scheme for partitioning grid-point space to MPI tasks where MPI is a protocol for the data exchange and synchronisation between the executing tasks of a parallel job. While this scheme has served ECMWF well there has still been some areas of concern, namely:

- Communication overheads for IFS reduced grids at the poles to support the Semi-Lagrangian scheme.
- The halo requirements needed to support the interpolation of fields between model and radiation grids. The halo is an area of a task's memory reserved for data that is owned by neighbouring tasks.

These issues have been addressed by the implementation of a new partitioning scheme called EQ_REGIONS which is characterised by an increasing number of partitions in bands from the poles to the equator. The number of bands and the number of partitions in each particular band are derived so as to provide partitions of equal area and small 'diameter'. The EQ_REGIONS algorithm used in IFS is based on the work of Paul Leopardi, School of Mathematics, University of New South Wales, Sydney, Australia.

IFS parallelisation

The Integrated Forecast System (IFS) at ECMWF consists of a large suite of software used primarily to produce a daily ten-day forecast. Key components of the IFS include the processing of observations, a 4D-Var analysis, a ten-day forecast model at T799L91 resolution, and a fifteen-day ensemble prediction system (EPS). The EPS comprises a control forecast at T399L62 resolution and a further 50 forecasts at T399L62 using perturbed initial states. These run out to ten days, when the resolution is truncated to T255L62 before completing the runs out to fifteen days.

The key parallelisation scheme for IFS was developed in the mid-1990s (*Barros et al.,* 1995; *Isaksen & Hamrud,* 1996). This scheme consists of a series of data transpositions between grid-point space, Fourier space and spectral space as shown in Figure 1. With this approach, the complete data required is redistributed at these stages of a time step so that the computations between two consecutive transpositions can be performed without any inter-process communication.

The focus of this article is the distribution of data in grid-point space where the bulk of the IFS computations take place for the physics and dynamics aspects of the model. This two-dimensional distribution of grid-point space to 256 MPI tasks is shown in Figure 2(a) using an Aitoff projection. Here each task is associated with a single partition. This projection allows us to easily see the main features of this distribution for the whole globe, with 16 north-south bands (or sets) and 16 east-west bands, where 16 is the square root of 256. For 512 tasks there is no exact square root, so the nearest factors of 512 are used, namely, 32 north-south bands and 16 east-west bands, as shown in Figure 2(b). The reason for having more north-south bands than east-west bands in this case is down to performance, as IFS applications run marginally faster this way. Finally, Figure 2(c) shows a case for 1024 MPI tasks.

The Semi-Lagrangian scheme in the IFS requires data from neighbouring tasks, the width of this halo being a function of the maximum possible wind speed and the time step. It is clear that the ideal shape for two-dimensional partitions (hereafter referred to as 2D partitions) to achieve the smallest halo area is square-like. However, the 2D partitions shown in Figure 2 are more rectangular than square-like with a width to height ratio of 2:1 or 4:1, and further they have a nasty 'cake' feature for those partitions at the poles. These polar partitions present a problem as they not only require a large halo area, but also that this halo area requires communication with all partitions converging on the relevant pole. Reducing the number of north-south bands and increasing the number east-west bands would make partitions more square-like at the equator, however this would only make matters worse at the poles with the increased communication. Some attempts to resolve the polar halo problems were made by using a different partitioning strategy for the first and last partitions (i.e. a polar cap). However, these developments only made a small difference on performance and sometimes had a negative effect. What was required was a new approach for the partitioning of grid-point space.

 doi:10.21957/2m1i7n0v3a

There are clearly many approaches to partitioning a sphere, a good example being that of *Lemaire & Weill* (2000) which uses constant area quadrangles as shown in Figure 3. However, many of these approaches share little in common with the IFS 2D partitioning; as a result they would require a major effort (many person years) to incorporate any of them into the IFS.

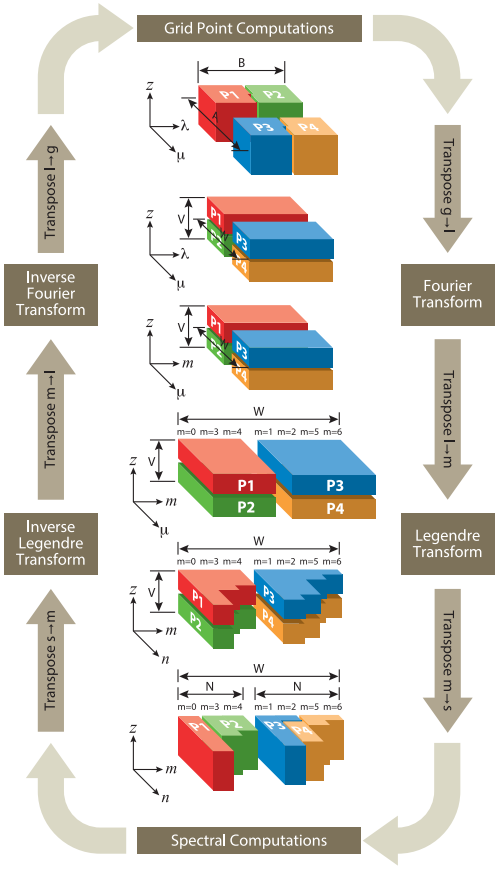

Figure 1 IFS model time step showing data transpositions.

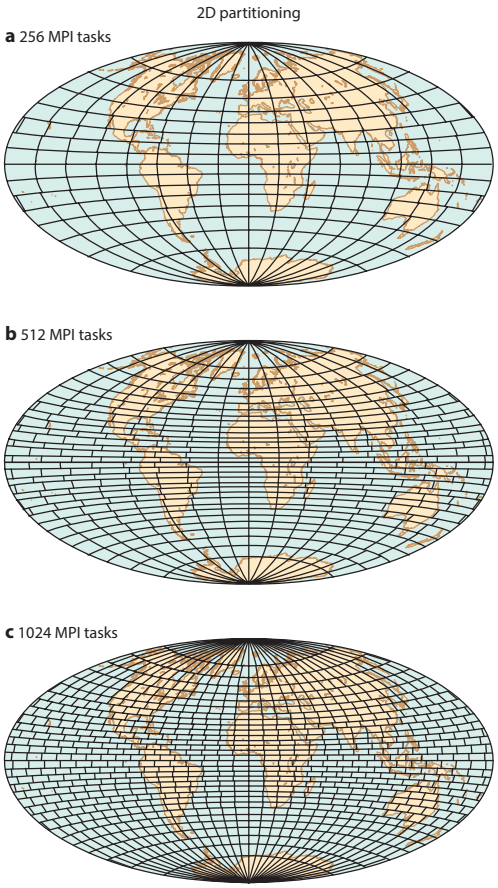

Figure 2 (2D partitioning of grid-point space to (a) 256 MPI tasks, (b) 512 MPI tasks and (c) 1024 MPI tasks for the T799 model.

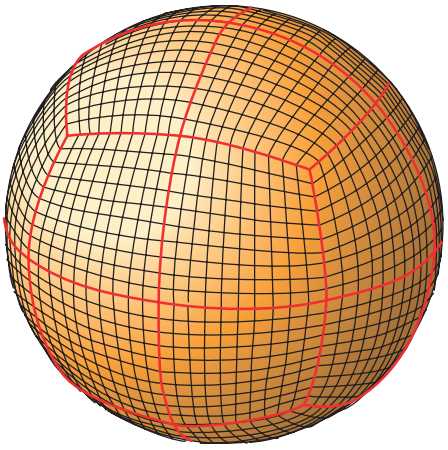

Figure 3 Partitioning the sphere with constant area quadrangles (taken from *Lemaire & Weill,* 2000).

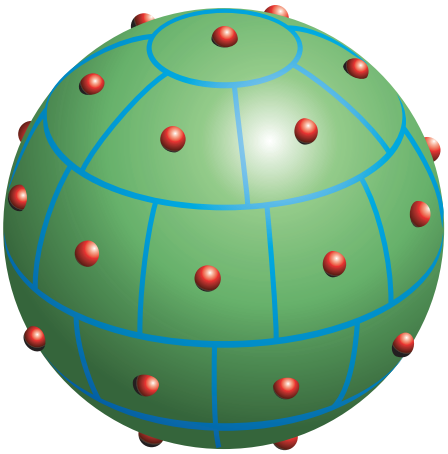

Figure 4 Partitioning of a sphere into 33 regions using the EQ_REGIONS MatLab package.

doi:10.21957/2m1i7n0v3a

EQ_REGIONS

The EQ_REGIONS partitioning scheme (*Leopardi,* 2006) appeared attractive from the beginning, as can be seen in Figure 4 for a sphere with 33 partitions. This practical algorithm came with a MatLab package titled *Recursive Zonal Equal Area Sphere Partitioning Toolbox* which can be found at http://eqsp.sourceforge.net/. As a comparison IFS 2D partitioning would need 11 bands north-south and 3 bands east-west for this number of partitions. The reason why EQ_REGIONS was attractive is due to the similarity between the IFS 2D bands and what EQ_REGIONS calls collars – the partitioning of a sphere resulting in two polar caps plus a number of collars with increasing partition counts as we approach the equator.

The description of the algorithm and mathematical proof are described in great detail in *Leopardi* (2006). This algorithm results in partitions of equal area and small 'diameter'. However, this would not be sufficient for an IFS implementation, as the density of grid-points on the globe varies with the latitude, the greatest density being at the poles and the least density at the equator. This imbalance is 13% for a T799 model with 512 partitions when using the EQ_REGIONS algorithm to provide the bounds information (start/end latitude, start/end longitude) for each partition as shown in Figure 5.

The solution to this imbalance issue was to use the EQ_REGIONS algorithm to only provide the band information, i.e. the number of north-south bands and the number of partitions per band. Then the IFS partitioning code would use this information in a similar way to that used for 2D partitioning, resulting in an equal number of grid-points per partition. With this approach there was only one new data structure (an single-dimensioned array) used to store the number of partitions in each band.

The results for this balanced EQ_REGIONS implementation can be seen in Figure 6(a) for 256 partitions, Figure 6(b) for 512 partitions and Figure 6(c) for 1024 partitions. The characteristic features of this partitioning approach are square-like partitions for most of the globe and polar caps, together with a significant improvement in the convergence at the poles.

From a code point of view the differences between the old 2D partitioning and the new EQ_REGIONS partitioning are relatively simple as shown in Box A.

Coding differences between 2D and EQ_REGIONS partitioning **A**

For the 2D partitioning there were loops such as:

```
DO JB=1,NPRGPEW
   DO JA=1,NPRGPNS
     ...
   ENDDO
 ENDDO
```

where `NPRGPEW` and `NPRGPNS` are the number of east-west and north-south bands (or sets).

For EQ_REGIONS partitioning loops were simply transformed into:

```
DO JA=1,N_REGIONS_NS
   DO JB=1,N_REGIONS(JA)
     ...
   ENDDO
 ENDDO
```

where `N_REGIONS_NS` is the number of north-south EQ_ REGIONS bands. `N_REGIONS(:)` is an array containing the number of partitions for each band.

In total some 100 IFS routines were modified with such transformations. It should be noted that the above loop transformation supports both 2D and EQ_REGIONS partitioning, i.e. to use 2D partitioning, a simple namelist variable would be set `LEQ_REGIONS=F`, which would result in the following initialisation:

```
N_REGIONS_NS=NPRGPNS
N_REGIONS(:)=NPRGPEW
```


doi:10.21957/2m1i7n0v3a

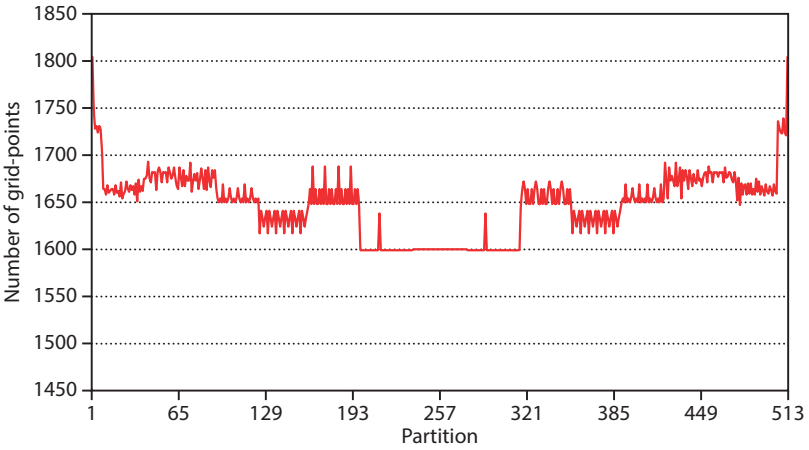

Figure 5 Number of grid-points per partition for T799 resolution using EQ_REGIONS 'area' partitioning for 512 tasks.

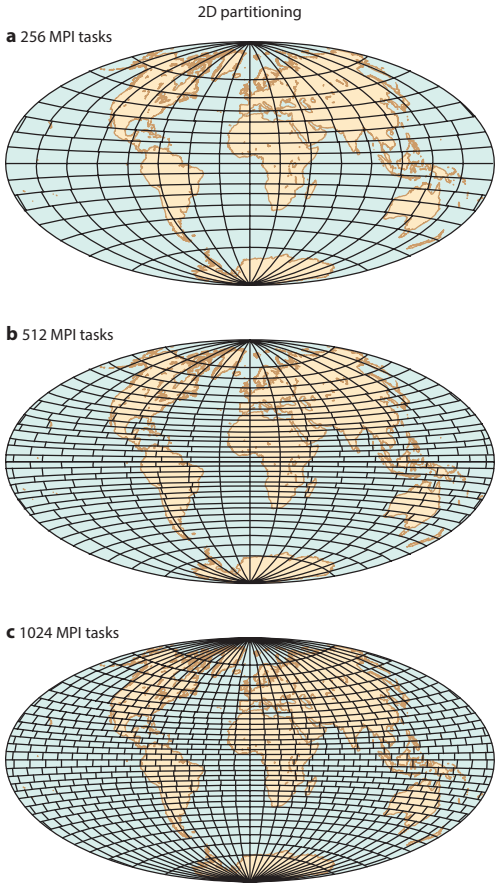

Figure 6 EQ_REGIONS partitioning of grid-point space to (a) 256 MPI tasks, (b) 512 MPI tasks and (c) 1024 MPI tasks for the T799 model.

Radiation Grid

Radiation computations in the IFS (as in other models) are very expensive and to reduce their relative cost we both reduce the frequency of such computations (i.e. every hour for a T799 model) and run these computations on a coarser grid than the model grid. This coarse grid was initially a sampled model grid (*Dent & Mozdzynski,* 1996) but more recently a reduced grid (linear) was used which gave a small improvement in meteorological skill. As an example, for a T799 model grid (843,490 grid points) the corresponding radiation grid would be T399 (213,988 grid points), with an approximate ratio of 4:1. Of course to use this reduced radiation grid we must interpolate data from the model grid required for the radiation computations and after such computations interpolate data back to the model grid from the radiation grid, in both cases using 12 point bi-dimensional interpolations.

Two approaches are possible for such interpolation. The first approach would be to send each field to a separate task using a global gather operation, perform the interpolation and then return the data by using a global scatter operation. This approach has the disadvantage that global operations are

relatively expensive on today's parallel computers, particularly when the number of fields (variables × levels) to interpolate is less than the number of tasks being used. The second approach, the one used in IFS, was to use a halo for the data required from neighbouring tasks and perform the interpolations locally. This approach was assumed to be better as all tasks are used for the interpolation and only local communication is used. The second approach also had an advantage from a code maintenance viewpoint as the routines needed for supporting the halos already existed for the Semi-Lagrangian scheme and could easily be used for the radiation interpolation.

However, there was a downside to this second approach. The halos required for the radiation interpolations were much larger than one would expect, particularly for large numbers of tasks. It was only after the geographic position of some partitions that required large halos were plotted that the problem was understood. Figures 7 and 8 show the model and radiation partitions (using 2D partitioning) for task 201 (Africa partition) and task 11 (Polar partition) of 512 for a T799 model and T399 radiation grid. One would have expected partitions for two different grids to be at the same geographic position, as they use the same partitioning code. The reason this is not the case stems from the definition of these two grids, one is clearly not a projection of the other. Differences between these two grids are due to the requirements for equal spacing of grid points in each grid (with the exception of polar latitudes). In addition there is a need for the number of points on the same line of latitude to be divisible by factors 2, 3, and 5 – a requirement for the Fourier transforms (see Figure 1).

It is now interesting to see the effect of using EQ_REGIONS partitioning and how it addresses the above problems. To make a fair comparison, tasks 220 and 4 were chosen that by comparison had a relatively large halo but also were located in comparable geographic positions (Africa and Polar) to tasks 201 and 11 for 2D partitioning. This can be seen in Figures 9 and 10 respectively.

The conclusion of this comparison is that EQ_REGIONS partitioning requires smaller halos than 2D partitioning for the interpolation when using the same number of tasks. This translates into less data being communicated and therefore improved performance. To take this further Figure 11 shows graphically the halo area (total halo grid points including a partition's own grid points) for all tasks for the radiation grid interpolation. The top two lines show the halo area required for interpolating from the model to the radiation grid and the bottom two lines for interpolating from the radiation to the model grid. In both cases it can clearly be seen that EQ_REGIONS partitioning results in smaller halos when compared with 2D partitioning.

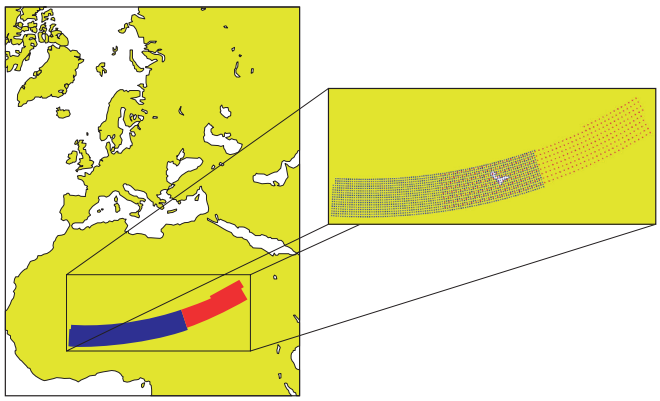

Figure 7 2D partitioning – model (blue) and radiation (red) partitions for task 201 (Africa partition) of 512 tasks for a T799 model/T399 radiation grid.

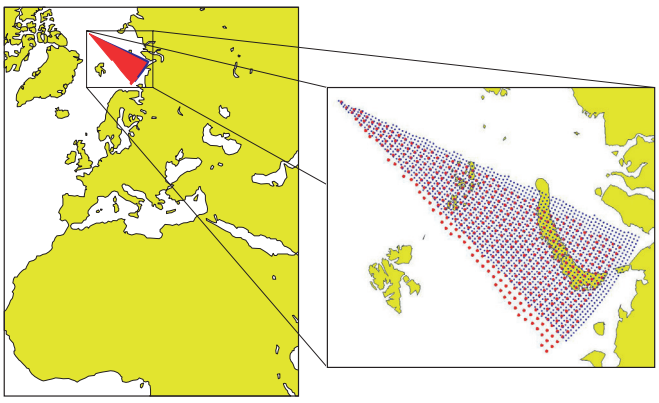

Figure 8 2D partitioning – model (blue) and radiation (red) partitions for task 11 (Polar partition) of 512 tasks for a T799 model/T399 radiation grid.

doi:10.21957/2m1i7n0v3a

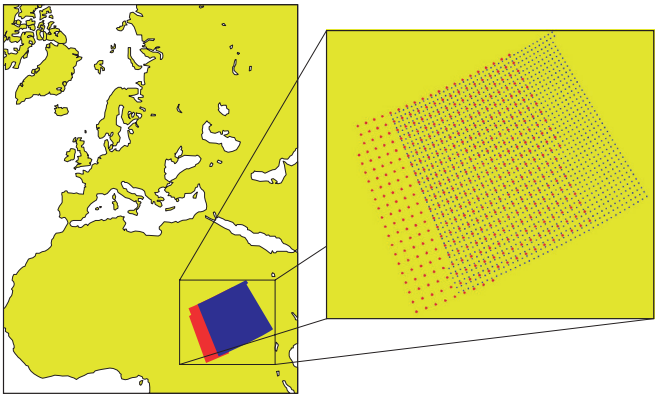

Figure 9 EQ_REGIONS partitioning – model (blue) and radiation (red) partitions for task 220 (Africa partition) of 512 tasks for a T799 model/T399 radiation grid.

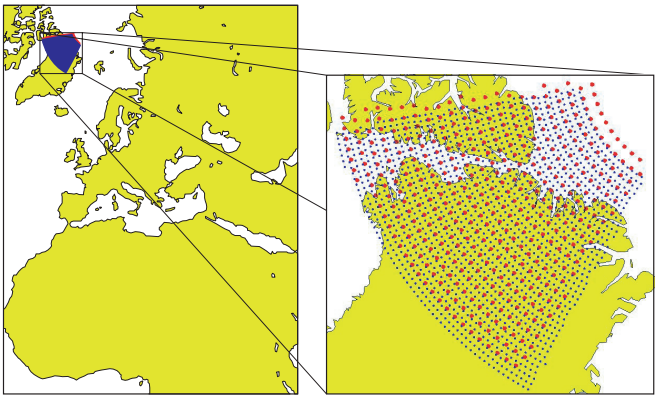

Figure 10 EQ_REGIONS partitioning – model (blue) and radiation (red) partitions for task 4 (Polar partition) of 512 tasks for a T799 model/T399 radiation grid.

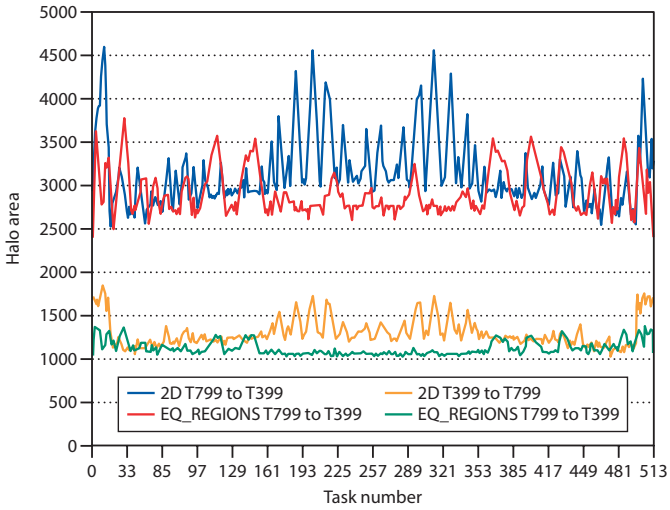

Figure 11 Comparing EQ_REGIONS and 2D partitioning requirement for Halo Area (total halo grid points including a partition's own grid points) for radiation grid interpolation for a T799 model/T399 radiation grid.

4D-Var

Besides the model and radiation grids presented earlier, there is another grid used in 4D-Var analyses – it is part of the J_b wavelet scheme (*Fisher,* 2003; *Fisher,* 2004) within the minimisation steps of a 4D-Var cycle. The grid used in this scheme is a 'full grid', where all latitudes contain the same number of points. For the J_b wavelet scheme many such grids of increasing resolution are used (the wavelet scales) from some minimum truncation (default being T21) up to the resolution of the minimisation step. As an example, for a T255 minimisation step, wavelet grids are used at T255, T213, T159, T127, T95, T63, T42, T30, and T21. This can present a problem when scaling to large numbers of tasks as the lowest trunction T21 only has 2048 grid points for a full grid (32 latitudes each with 64 points). Of course the performance of the J_b wavelet code is dominated by the higher wavelet scales so one should not pay too much attention to the low resolutions. Unfortunately, when testing EQ_REGIONS partitioning it was exactly the lowest resolution that presented a problem; the problem being a constraint of IFS partitioning code that required that a line of latitude could not be split more than once between bands (or A-sets). For 2D partitioning this was not found to be a problem.

doi:10.21957/2m1i7n0v3a

After some investigation and discussion with the author of the J_b wavelet scheme a solution was found. The solution was simply to use reduced grids (preferably linear grids) instead of full grids. It was decided to implement this unconditionally as it was realised that the overall performance of a 4D-Var cycle improved by about 2% independent of whether 2D partitioning or EQ_REGIONS partitioning was used. The reason for this performance improvement was obvious, about 30% fewer grid points were used compared with full grids in the above scheme. The wavelet scales used were preset in some large data files, so it was not practical in this implementation to change these. Further, some of these scales were not linear grids (e.g. T213 is a quadratic grid). In the near future it is planned to reset the wavelet scales so that all have corresponding linear grids. This is expected to further improve the overall performance of 4D-Var by an additional 1%.

Performance

In Table 1 there is a comparison of the performance of a T799 forecast model and 4D-Var analysis when using 2D and EQ_REGIONS partitioning. The advantage in using EQ_REGIONS was measured at 3.9% for the forecast model and 2.7% for the 4D-Var analysis. For 4D-Var the 2% improvement gained by using reduced grids for the J_b wavelet grids discussed in the previous section is not included in the latter 2.7%.

While the overall performance advantage for using EQ_REGIONS may only be a few percent, one should bear in mind that IFS applications have had many years of code optimisation and finding even a few percent is increasingly rare. 'Rich pickings' are often found in new code, but less so with code that has been around a long time.

By inspecting some low-level timers in the IFS, it can be seen that some areas of code are now running faster due to a combination of reduced halo sizes used for the Semi-Lagrangian scheme and radiation grid interpolation, and the associated reduction in memory use (always good for performance). However, it can also be seen that there is an increase in the communications used for the transposition of data between grid-point space and Fourier space (see Figure 1). This increased communication is due to the fact that the distribution of latitudes in Fourier space is a very close match to the north-south bands when using 2D partitioning. The transposition to Fourier space from grid-point space requires complete latitudes as the first dimension, with model levels being distributed as the second dimension. For systems with many CPUs per node (such as ECMWF's IBM p5-575 clusters) a relatively large part of this communication can be performed within each node (when using 2D partitioning). We know that intra-node communication via memory is faster than inter-node communication via the Federation switch. Therefore this seems a reasonable explanation why 2D partitioning is better for this particular communication phase. This scenario naturally changes when truly large numbers of tasks (thousands) or thinner nodes are used. In both these cases, most of the data in such transpositions (if not all transpositions) will need to be communicated via the systems switch, with further gains to EQ_REGIONS over 2D partitioning. Time will tell.

EQ_REGIONS partitioning was introduced in IFS cycle Cy31r2 (November 2006) as the default partitioning scheme for grid-point space. This new partitioning approach together with ongoing developments should allow IFS to continue to scale well on todays and future high performance parallel computers.

Application	Tasks × Threads	2D partitioning (seconds)	EQ_REGIONS partitioning (seconds)	2D/ EQ_REGIONS speed-up
Model	512 × 2	3648	3512	1.039
4D-Var	96 × 8	3563	3468	1.027

Table 1 Comparing the performance of 2D partitioning and EQ_REGIONS partitioning for a T799L91 IFS model and 4D-Var analysis.

doi:10.21957/2m1i7n0v3a

Further Reading

Barros, S., D. Dent, L. Isaksen, G. Robinson, G. Mozdzynski & **F. Wollenweber,** 1995: The IFS model: a parallel production weather code. *Parallel Computing,* **21,** 1621–1638.

Dent, D. & **G. Mozdzynski,** 1996: ECMWF operational forecasting on a distributed memory platform: Forecast model. In *Proc. of the Seventh ECMWF Workshop on the Use of Parallel Processors in Meteorology,* G.-R. Hoffman and N. Kreitz editors, World Scientific, 144–154.

Fisher, M., 2003: Background error covariance modeling. In *Proc. of ECMWF Seminar on Recent Developments in Data Assimilation for Atmosphere and Ocean,* September 2003, http://www.ecmwf.int/publications/library/ecpublications/.pdf/seminar/ 2003/sem2003_fisher.pdf.

Fisher, M., 2004: Generalized frames on the sphere, with application to the background error covariance modeling. *In Proc. of ECMWF Seminar on Recent Developments in Numerical Methods for Atmospheric and Ocean Modelling,* September 2004, http://www.ecmwf.int/publications/library/ecpublications/-pdf/ seminar/ 2004/sem2004.fisher.pdf.

Isaksen, L. & **M. Hamrud,** 1996: ECMWF operational forecasting on a distributed memory platform: Analysis. In *Proc. of the Seventh ECMWF Workshop on the Use of Parallel Processors in Meteorology,* G.-R. Hoffman and N. Kreitz editors, World Scientific, 22–35.

Lemaire, C. & **J. Weill,** 2000: *Partitioning the sphere with constant area quadrangles.* 12th Canadian Conference on Computational Geometry, 16–19 August 2000, citeseer.ist.psu.edu/lemaireOOpartitioning.html.

Leopardi, P., 2006: A partition of the unit sphere into regions of equal area and small diameter. *Electronic Transactions on Numerical Analysis,* **25,** 309–327.

doi:10.21957/2m1i7n0v3a