When defining **key_dimgout**, the output are written in DIMG format, an IEEE output format.

Since version 3.3, support for NetCDF4 chunking and (loss-less) compression has been included. These options build on the standard NetCDF output and allow the user control over the size of the chunks via namelist settings. Chunking and compression can lead to significant reductions in file sizes for a small runtime overhead. For a fuller discussion on chunking and other performance issues the reader is referred to the NetCDF4 documentation : http ://www.unidata.ucar.edu/software/netcdf/docs/netcdf.html#Chunking

The new features are only available when the code has been linked with a NetCDF4 library (version 4.1 onwards, recommended) which has been built with HDF5 support (version 1.8.4 onwards, recommended). Datasets created with chunking and compression are not backwards compatible with NetCDF3 "classic" format but most analysis codes can be relinked simply with the new libraries and will then read both NetCDF3 and NetCDF4 files. NEMO executables linked with NetCDF4 libraries can be made to produce NetCDF3 files by setting the *ln_nc4zip* logical to false in the *namnc4* namelist :

```
!!======================================================================
!!   namnc4            netcdf4 chunking and compression settings
!!======================================================================
!-----------------------------------------------------------------------
&namnc4         !  netcdf4 chunking and compression settings          ("key_netcdf4")
                !  (benign if "key_netcdf4" is not used)
!-----------------------------------------------------------------------
   nn_nchunks_i   =   4       !  number of chunks in i-dimension
   nn_nchunks_j   =   4       !  number of chunks in j-dimension
   nn_nchunks_k   =   31      !  number of chunks in k-dimension
                              !  setting nn_nchunks_k = jpk will give a chunk size of 1 in the vertical which
                              !  is optimal for postprocessing which works exclusively with horizontal slabs
   ln_nc4zip      =   .TRUE.  !  (T) use netcdf4 chunking and compression
                              !  (F) ignore chunking information and produce netcdf3-compatible files
/
```

If **key_netcdf4** has not been defined, these namelist parameters are not read. In this case, *ln_nc4zip* is set false and dummy routines for a few NetCDF4-specific functions are defined. These functions will not be used but need to be included so that compilation is possible with NetCDF3 libraries.

When using NetCDF4 libraries, **key_netcdf4** should be defined even if the intention is to create only NetCDF3-compatible files. This is necessary to avoid duplication between the dummy routines and the actual routines present in the library. Most compilers will fail at compile time when faced with such duplication. Thus when linking with NetCDF4 libraries the user must define **key_netcdf4** and control the type of NetCDF file produced via the namelist parameter.

Chunking and compression is applied only to 4D fields and there is no advantage in chunking across more than one time dimension since previously written chunks would have to be read back and decompressed before being added to. Therefore, user control over chunk sizes is provided only for the three space dimensions. The user sets an approximate number of chunks along each spatial axis. The actual size of the chunks will depend on global domain size for mono-processors or, more likely, the local processor domain size for distributed processing. The derived values are subject to practical minimum values (to avoid wastefully small chunk sizes) and cannot be greater than the domain size in any dimension. The algorithm used is :

```
ichunksz(1) = MIN( idomain_size,MAX( (idomain_size-1)/nn_nchunks_i + 1 ,16 ) )
ichunksz(2) = MIN( jdomain_size,MAX( (jdomain_size-1)/nn_nchunks_j + 1 ,16 ) )
ichunksz(3) = MIN( kdomain_size,MAX( (kdomain_size-1)/nn_nchunks_k + 1 , 1 ) )
ichunksz(4) = 1
```

As an example, setting :

```
nn_nchunks_i=4, nn_nchunks_j=4 and nn_nchunks_k=31
```

for a standard ORCA2_LIM configuration gives chunksizes of `46x38x1` respectively in the mono-processor case (i.e. global domain of `182x149x31`). An illustration of the potential space savings that NetCDF4 chunking and compression provides is given in table 13.1 which compares the results of two short runs of the ORCA2_LIM reference configuration with a 4x2 mpi partitioning. Note the variation in the compression ratio achieved which reflects chiefly the dry to wet volume ratio of each processing region.

Since version 3.2, an I/O server has been added which provides more flexibility in the choice of the fields to be output as well as how the writing work is distributed over the processors in massively parallel computing. It is activated when **key_iomput** is defined.

When **key_iomput** is activated with **key_netcdf4** chunking and compression parameters for fields produced via *iom_put* calls are set via an equivalent and identically named namelist to *namnc4* in *xmlio_server.def*. Typically this namelist serves the mean files whilst the *namnc4* in the main namelist file continues to serve the restart files. This duplication is unfortunate but appropriate since, if using io_servers, the domain sizes of the individual files produced by the io_server processes will be different to those produced by the invidual processing regions and different chunking choices may be desired.

13.8.2 Tracer/Dynamics Trends (key_trdmld, key_trdtra, key_trddyn, key_trdmld_trc)

```
!-----------------------------------------------------------------------
&namtrd        !   diagnostics on dynamics and/or tracer trends         ("key_trddyn" and/or "key_trdtra")
!              !       or mixed-layer trends or barotropic vorticity    ('key_trdmld' or  "key_trdvor")
!-----------------------------------------------------------------------
   nn_trd      = 365       !  time step frequency dynamics and tracers trends
   nn_ctls     =   0       !  control surface type in mixed-layer trends (0,1 or n<jpk)
   rn_ucf      =   1.      !  unit conversion factor (=1 -> /seconds ; =86400. -> /day)
   cn_trdrst_in      = "restart_mld"   ! suffix of ocean restart name (input)
   cn_trdrst_out     = "restart_mld"   ! suffix of ocean restart name (output)
   ln_trdmld_restart = .false.         !  restart for ML diagnostics
   ln_trdmld_instant = .false.         !  flag to diagnose trends of instantantaneous or mean ML T/S
/
```

When **key_trddyn** and/or **key_trddyn** CPP variables are defined, each trend of the dynamics and/or temperature and salinity time evolution equations is stored in three-dimensional arrays just after their computation ($i.e.$ at the end of each $dyn \cdots .F90$ and/or $tra \cdots .F90$ routines). These trends are then used in *trdmod.F90* (see TRD directory) every *nn_trd* time-steps.

What is done depends on the CPP keys defined :

key_trddyn, key_trdtra : a check of the basin averaged properties of the momentum and/or tracer equations is performed ;

key_trdvor : a vertical summation of the moment tendencies is performed, then the curl is computed to obtain the barotropic vorticity tendencies which are output ;

Filename	NetCDF3 filesize (KB)	NetCDF4 filesize (KB)	Reduction %
ORCA2_restart_0000.nc	16420	8860	47%
ORCA2_restart_0001.nc	16064	11456	29%
ORCA2_restart_0002.nc	16064	9744	40%
ORCA2_restart_0003.nc	16420	9404	43%
ORCA2_restart_0004.nc	16200	5844	64%
ORCA2_restart_0005.nc	15848	8172	49%
ORCA2_restart_0006.nc	15848	8012	50%
ORCA2_restart_0007.nc	16200	5148	69%
ORCA2_2d_grid_T_0000.nc	2200	1504	32%
ORCA2_2d_grid_T_0001.nc	2200	1748	21%
ORCA2_2d_grid_T_0002.nc	2200	1592	28%
ORCA2_2d_grid_T_0003.nc	2200	1540	30%
ORCA2_2d_grid_T_0004.nc	2200	1204	46%
ORCA2_2d_grid_T_0005.nc	2200	1444	35%
ORCA2_2d_grid_T_0006.nc	2200	1428	36%
ORCA2_2d_grid_T_0007.nc	2200	1148	48%
...	...	...	..
ORCA2_2d_grid_W_0000.nc	4416	2240	50%
ORCA2_2d_grid_W_0001.nc	4416	2924	34%
ORCA2_2d_grid_W_0002.nc	4416	2512	44%
ORCA2_2d_grid_W_0003.nc	4416	2368	47%
ORCA2_2d_grid_W_0004.nc	4416	1432	68%
ORCA2_2d_grid_W_0005.nc	4416	1972	56%
ORCA2_2d_grid_W_0006.nc	4416	2028	55%
ORCA2_2d_grid_W_0007.nc	4416	1368	70%

TABLE 13.1 – Filesize comparison between NetCDF3 and NetCDF4 with chunking and compression