NESTING TOOLS
The mesh generations

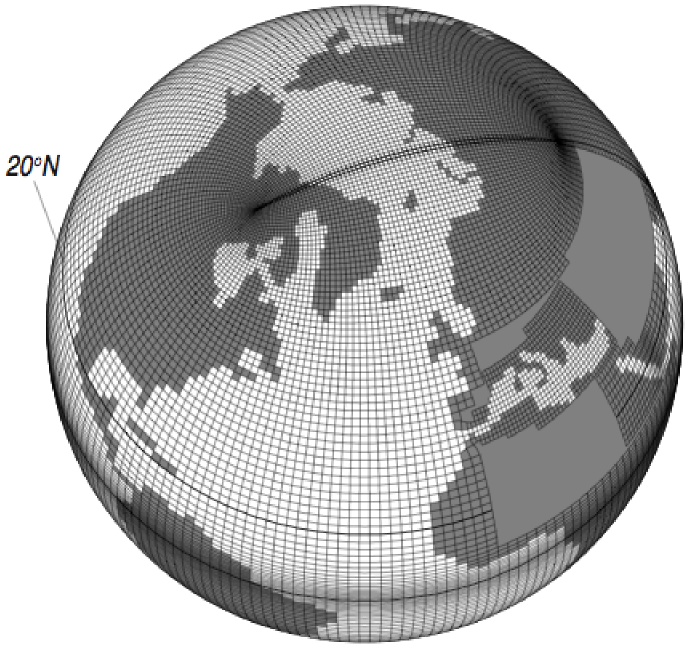


Figure 1: *meshing of the ORCA grids*
(source: NEMO website)

Nemo System Team
Brice Lemaire

July 9, 2010

Contents

Chapter 1

Introduction

The objective is to have a toolbox allowing the creation of regional configurations from curvilinear grid. The first part will explain the method in that general case. With a series of procedures Fortran, we want to be able to make zooms on a domain (increasing the resolution) or a simple extraction of a sub-domain. The domain may be regional or global and the format of input and output files will be NetCDF format.
The main difficulty is the use of ORCA grids which are global (left/right periodicity) and tripolar (North boundary) and thus, no-regular. Indeed, to delete the singularity at the geographical north pole, two north poles have been created outside of the ocean domain (over Russia and Canada, see figure 1). Although ORCA grids are curvilinears, they are orthogonal to allow the use of the finite difference method. So, in a second part, we are going to see the aspect and features of the ORCA grids.
We have chosen to write exclusively with Fortran for portability.

Chapter 2

Curvilinear Grids and Interpolation

2.1 Generalities

The curvilinear grids are no regular (see figure 2.1). To increase the resolution we need to calculate the news points by making an interpolation. Thus, to interpolate inside a curvilinear grid, the simpliest way is to place us in the matrix plan (see figure 2.2). Indeed, in this plan, the spatial step is regular and constant (Δx=Δy=Δi=$\Delta j = 1$).

Curvilinear grids representations

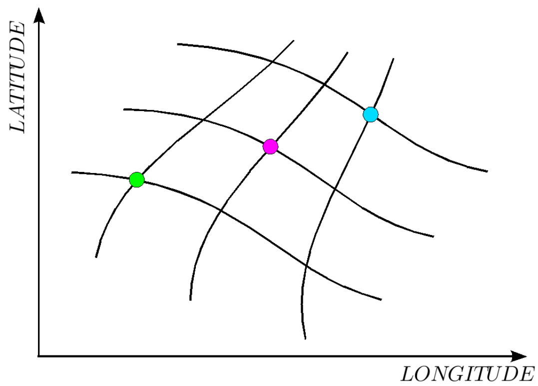


Figure 2.1: *physical plan*
longitude-latitude coordinates
$\Delta lon \neq \Delta lat \neq cst$

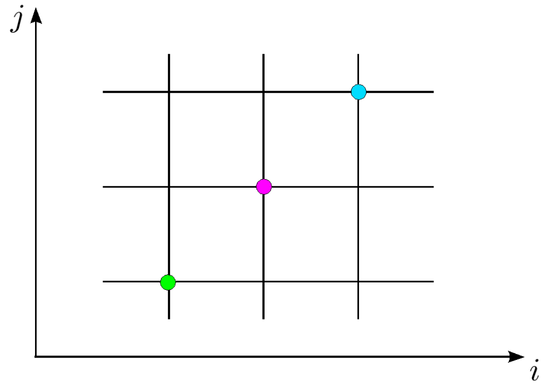


Figure 2.2: *matrix plan*
i-j indexation
$\Delta i = \Delta j = 1$

For this, we use a 3th-order polynomial interpolation (n=3) which needs 4 points. Indeed, the small variations inside the grid allow us ...
At nth-order and 2D, the polynomial interpolation can be written like this:

$$u(x_0, y_0) = \sum_{i,j=1}^{n+1} \underbrace{S_i \quad S_j}_{\substack{interpolation\\coefficients}} \overbrace{u(x_i, y_j)}^{\substack{known\\values}}$$

Where (x_0, y_0) are the coordinates of the interpolate point and (x_i, y_i) are coordinates of the known points. In the matrix plan, (x_i, y_i) are integer values equal to (i, j)-indexation.

By using Lagrangian polynomial, the 3th-order interpolation coefficients S_i, S_j are given by:

$$S_i = \prod_{\substack{k=1 \\ k \neq i}}^{4} \frac{x_0 - x_k}{x_i - x_k} \qquad\qquad S_j = \prod_{\substack{q=1 \\ q \neq j}}^{4} \frac{y_0 - y_q}{y_j - y_q}$$

2.2 1-D example

In 1D, the coefficients S_i are:

$$S_1 = \left(\frac{x_0 - x_2}{x_1 - x_2}\right)\left(\frac{x_0 - x_3}{x_1 - x_3}\right)\left(\frac{x_0 - x_4}{x_1 - x_4}\right)$$
$$S_2 = \left(\frac{x_0 - x_1}{x_2 - x_1}\right)\left(\frac{x_0 - x_3}{x_2 - x_3}\right)\left(\frac{x_0 - x_4}{x_2 - x_4}\right)$$
$$S_3 = \left(\frac{x_0 - x_1}{x_3 - x_1}\right)\left(\frac{x_0 - x_2}{x_3 - x_2}\right)\left(\frac{x_0 - x_4}{x_3 - x_4}\right)$$
$$S_4 = \left(\frac{x_0 - x_1}{x_4 - x_1}\right)\left(\frac{x_0 - x_2}{x_4 - x_2}\right)\left(\frac{x_0 - x_3}{x_4 - x_3}\right)$$

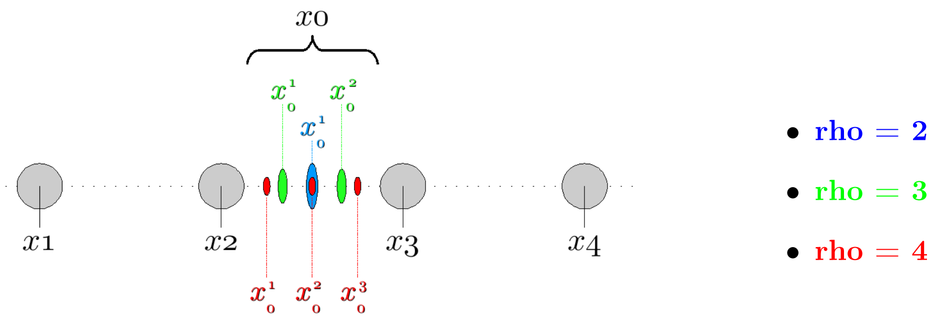


Figure 2.3: *Position of points to interpolate in function of the factor of refinement rho*

We see (figure 2.3), that the number of the new points to calculate is equal to $rho - 1$.

By placing us in the matrix representation (see figure 2.2), the values of (x_i,y_i) and (x_k,y_k) are integer values (i,j). Thus, the distance Δx is an integer value (Δx=Δi=1). We can check the validity of the Lagrangian coefficients whose the sum must be equal to 1 (see table 2.1).

rho	x_0	x_1 / S_1	x_2 / S_2	x_3 / S_3	x_4 / S_4	$\sum S$
2	$x_2 + \frac{1}{2}$	$-\frac{1}{16}$	$\frac{9}{16}$	$\frac{9}{16}$	$-\frac{1}{16}$	1
3	$x_2 + \frac{1}{3}$	$-\frac{5}{81}$	$\frac{20}{27}$	$\frac{10}{27}$	$-\frac{4}{81}$	1
	$x_2 + \frac{2}{3}$	$-\frac{4}{81}$	$\frac{10}{27}$	$\frac{20}{27}$	$-\frac{5}{81}$	1
4	$x_2 + \frac{1}{4}$	$-\frac{7}{128}$	$\frac{105}{128}$	$\frac{35}{128}$	$-\frac{5}{128}$	1
	$x_2 + \frac{2}{4}$	$-\frac{1}{16}$	$\frac{9}{16}$	$\frac{9}{16}$	$-\frac{1}{16}$	1
	$x_2 + \frac{3}{4}$	$-\frac{5}{128}$	$\frac{35}{128}$	$\frac{105}{128}$	$-\frac{7}{128}$	1

Table 2.1: *Coefficients of interpolation for a few values of rho*

Chapter 3

ORCA grids

ORCA grids are curvilinear grids used by NEMO (Nucleus for European Modelling of the Ocean). To cover all the seas and oceans and to allow the using of the finite difference method, the north pole has been moved toward 2 points inside lands (Asian and Canada lands, see figure 1). There are two main kinds of ORCA global grids. The first one is a grid with 2 degrees of resolution and the second one with half degree. We will use one of these grids to build a fine grid, thus, it's important to know both difference and similarity between them. So, in a first part we are going to see the common points of the all ORCA grids and then, we will see the particular features of each one.

3.1 Generalities

3.1.1 Elementary cell and indexation

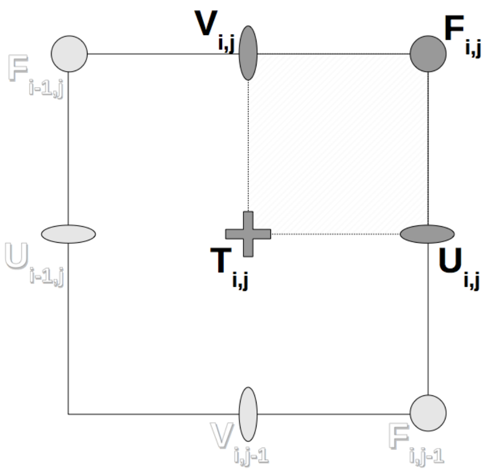


Figure 3.1: *plan of an elementary cell*

According to the "C" grid in Arakawa's classification, each cell of the ORCA grids is centered on scalar T-points (e.g. T, S, p etc...). The eastward U-points, northward V-points and north-east F-points have the same index (see figure 3.1). i-indexation begins in the Indian ocean and j-indexation begins over the antarctic land.

3.1.2 Along Equator

Equator line is located along the T and U-points. Indeed, the coriolis force is calculated at T-points and we know it must be zero along the equator. Thus, in order to respect the physics in this area, we want to set this constraint.

3.1.3 Matrix periodicity

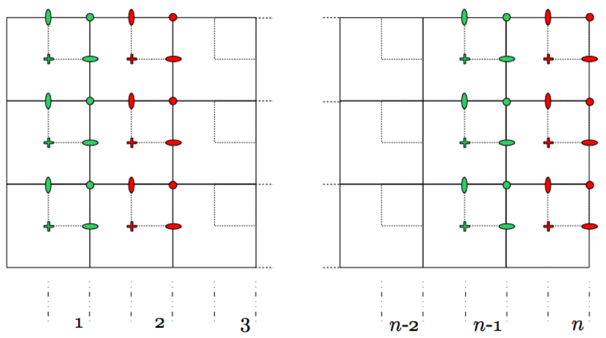


Figure 3.2: *overlap bands*

The ORCA grids are global and thus, to allow the periodicity, the first column of the matrix corresponds to the penultimate and the second column corresponds to the last one (see figure 3.2). So we have two bands that overlap and located in the Indian ocean.

$$\begin{aligned} column(1) &= column(n-1) \\ column(2) &= column(n) \end{aligned}$$

3.1.4 North boundary

As we said previously, ORCA grids possess two north poles, located over Russia and Canada (figure 1). So, there is a boundary between these points. The location of this boundary (and of the two north poles) is function of the kind of initial grid (i.e along T/U-points or along V/F-points). Indeed, to respect the constraint concerning the position of equator and depending on the resolution, the position of the two north poles won't be the same for each grid (ORCA 2 or 05). Moreover, this boundary cross the geographical north pole and so is located along two meridians. So, concerning the longitude, this boundary looks like a function of heavyside with a gap of ±180°.

3.2 Specificity of the two degrees ORCA grid (ORCA2)

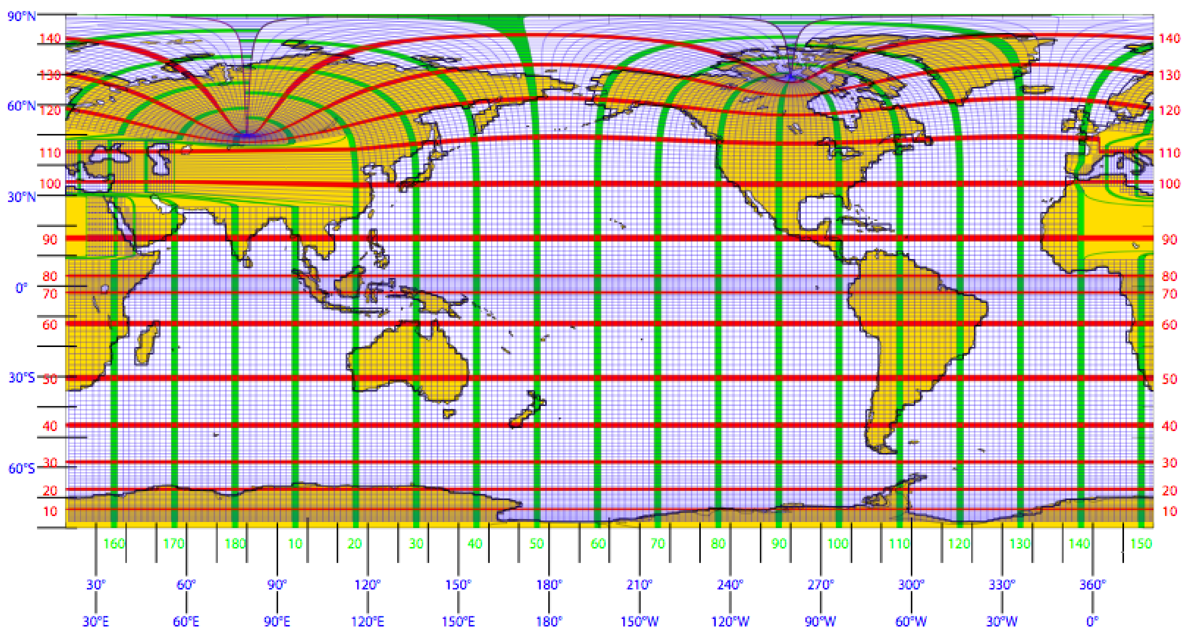


Figure 3.3: *2 degrees ORCA grid and indexation - size of matrix: 182×149 (source: NEMO website)*

We see that the i-indexation begins to 80°E accros the Indian ocean.

We also observe a deformation of the grid over a few lands. Indeed, some points located on lands and thus useless, have been moved towards the oceanic domain. This means an increasing of the resolution over the oceanic domains and a decreasing over land domains.

Moreover, a 2 degrees resolution doesn't allow the observation of some physical phenomena along equator and thus, in this area the resolution have been increased (from 2 degrees to half degree, see figure 3.3). The equator line is well located along the T-Upoints.

Concerning this grid, the 2 north bipoles, located over lands, are T-points (see figure 3.4 and 3.5) and so the boundary between them too.

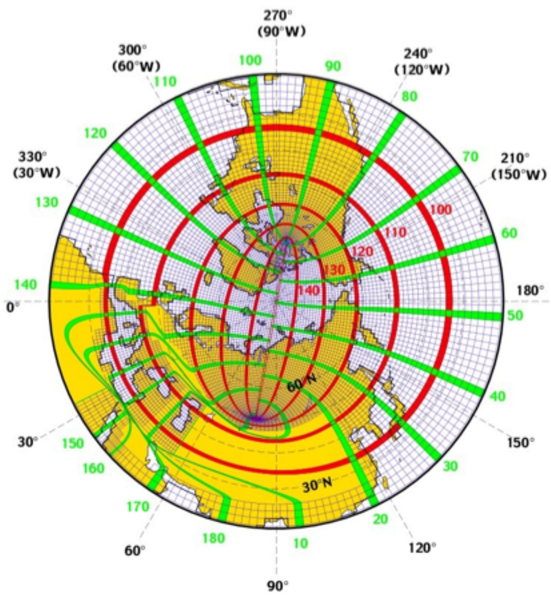


Figure 3.4: *North boundary (source: NEMO website)*

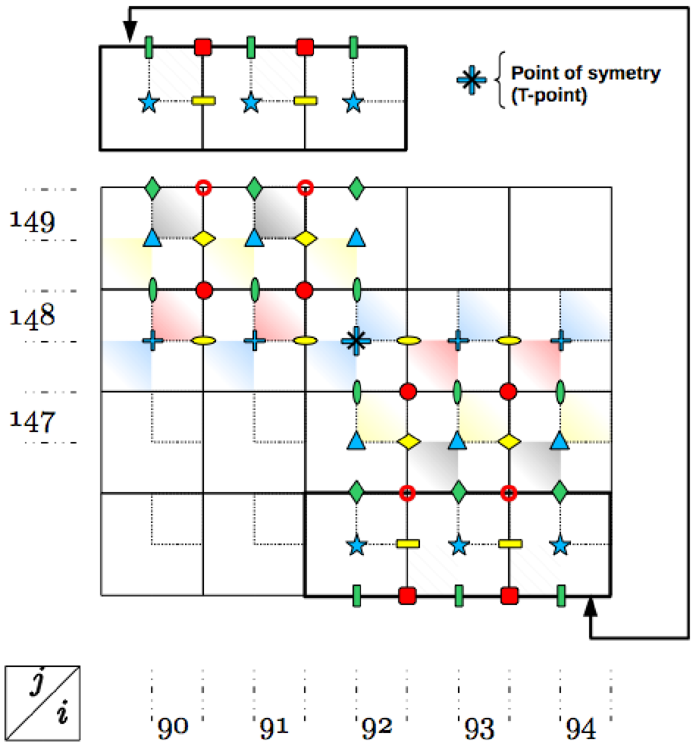


Figure 3.5: *Symmetry around the Canada bipole for ORCA2 grids*

We can see the symmetry along the north boundary (see figure 3.5). The point of symmetry is located over a T-point. In this case, we see the Canada bipole. We clearly see the symmetry concerns of the three last rows (j=147:149) plus the V/F-points of the row 146. Moreover, although some points have the same index at the left of the symmetry point, at the right these same points have no longer the same index.

$T_{91,149} = T_{93,147} \rightarrow (i, j)$
$U_{91,149} = U_{92,147} \rightarrow (i-1, j)$
$V_{91,149} = V_{93,146} \rightarrow (i, j-1)$
$F_{91,149} = F_{92,146} \rightarrow (i-1, j-1)$

3.3 Specificity of the half degree ORCA grid (ORCA05)

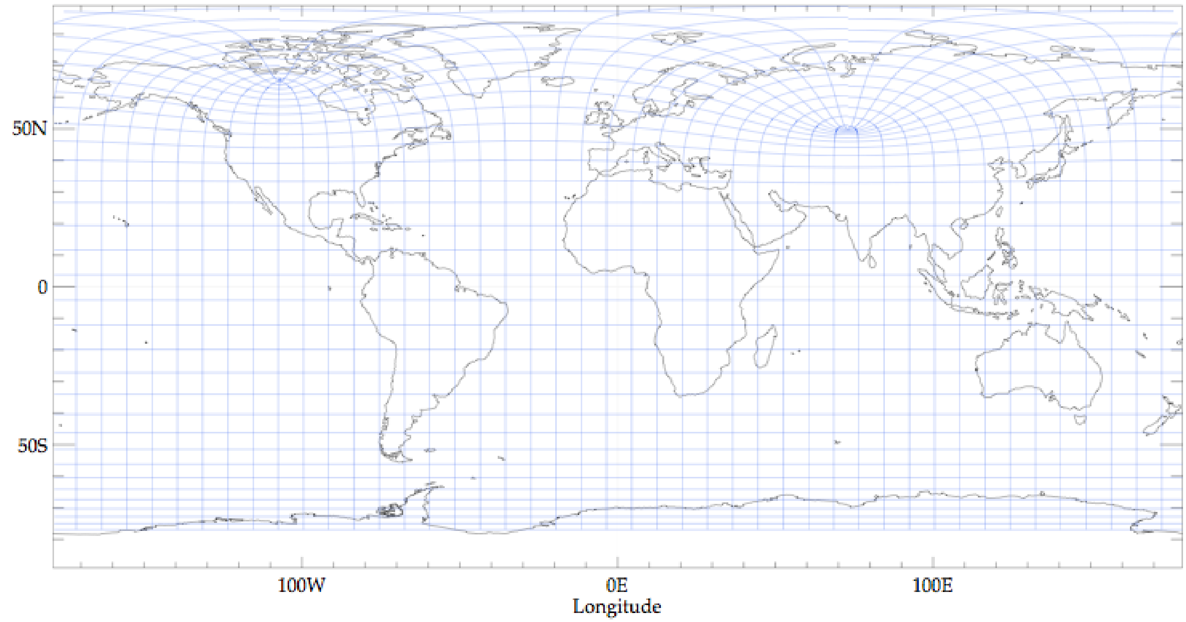


Figure 3.6: *ORCA05 grid (F-points)*
xstride=16 / ystride=16

The ORCA05 grid are no discontinuity from land points to oceanic points (see figure 3.6). Moreover, the resolution is quite fine along the equator, thus, it's constant. Equator is located along T and U-points.

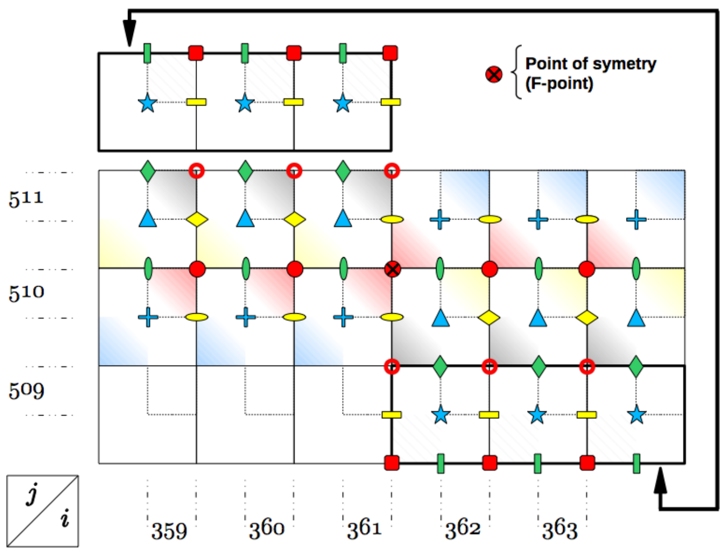


Figure 3.7: *Symmetry around the Canada bipole for half degree grids (ORCA05)*

We can see the symmetry along the north boundary (see figure 3.7). The point of symmetry is located over a F-point. We see the symmetry concerns the two last rows (j=510:511) plus the V/F-points of the row 509.

$T_{361,511} = T_{362,510} \rightarrow (i, j)$
$U_{361,511} = U_{361,510} \rightarrow (i-1, j)$
$V_{361,511} = V_{362,509} \rightarrow (i, j-1)$
$F_{361,511} = F_{361,509} \rightarrow (i-1, j-1)$

3.4 Summary

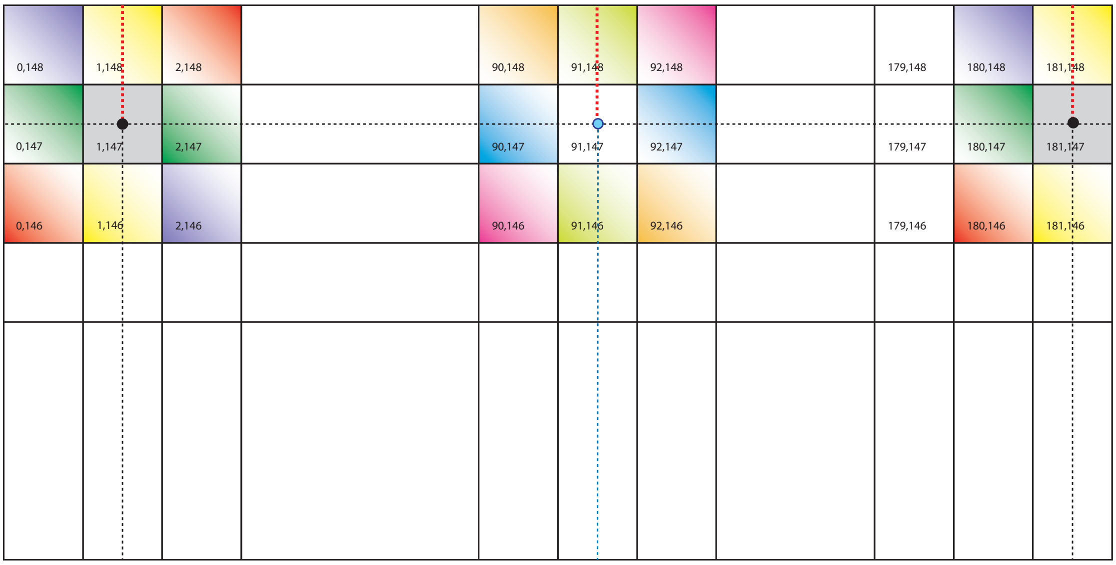


Figure 3.8: *Global shape of ORCA 2 matrix indexation begins to 0*

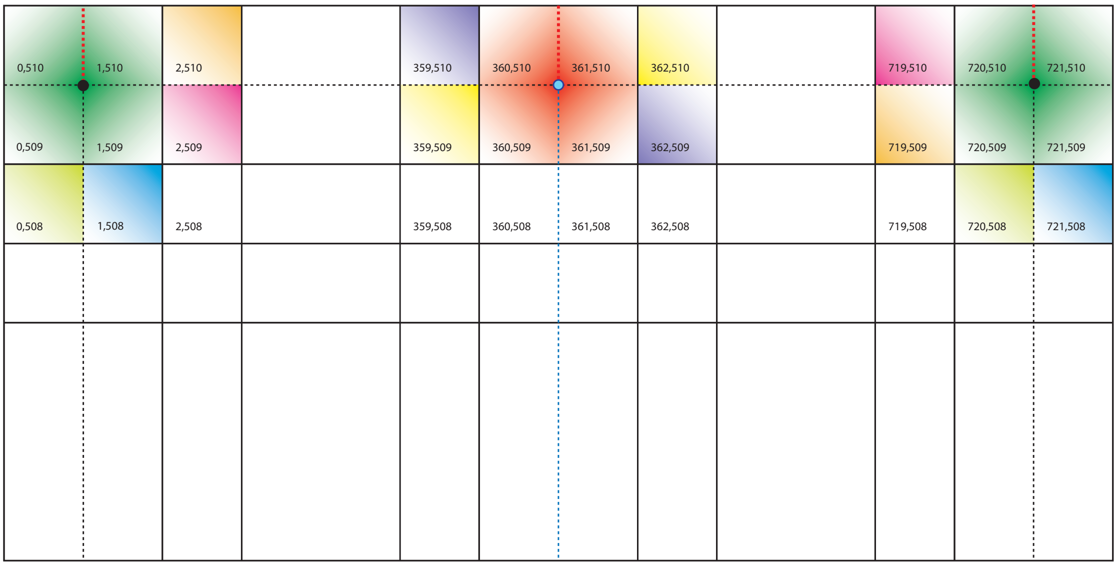


Figure 3.9: *Global shape of ORCA 05 matrix indexation begins to 0*

Grid	**size matrix**		**domain**		**location Asian bipole(s)**			**location Canada bipole**		
	nxsize	*nysize*	*min*	*max*	point	*i*	*j*	point	*i*	*j*
ORCA2	182	149	78°E 78.19°S	80°E 90°N	T	2 - 182 80°E	148 50°N	T	92 100°W	148 70°N
ORCA05	722	511	72.75°E 77.01°S	73.25°E 90°N	F	1 - 721 73°E	510 50°N	F	361 107°W	510 66°N

Table 3.1: *ORCA matrix parameters*

Chapter 4

Create the fine grid

To create a fine grid we need several parameters such as the boundaries (*nn_imin*, *nn_imax*, *nn_jmin*, *nn_jmax*) and the refinement factors (*nn_rhox* and *nn_rhoy*). We also need to know the initial grid (*cn_parent_coordinate_file*) and possibly the position of the point of symmetry (*cn_position_pivot*). These parameters are given in a namelist as shown below.

Extract 1. *namelist*

```
&coarse_grid_files
    cn_parent_coordinate_file = 'coordinates_ORCA_R2.nc'
    cn_position_pivot = 'T-grid'
/

&nesting
    nn_imin = 56
    nn_imax = 650
    nn_jmin = 150
    nn_jmax = 367
    nn_rhox  = 2
    nn_rhoy = 3
/
```

The main fortran program is called *create_coordinates.f90.*

4.1 Methods

The method to create the fine grid require four steps and can be describe like this:

1. **Define sub-domain** (*domain.f90*)
   First, we have to define the domain in the initial grid to know the cases. Instead, for example if the domain possess the overlap bands we have to suppress two of them.

2. **Create mixed grid** (*mixed_grid.f90*)
   Here, we create a grid to store all the known values for the same component (longitude, latitude, scale factors etc...) of the different grids (T,U,V & F). For this, we define the size of this grid and we write the known value by leaving the place for the values to determine by interpolation.

3. **Make Interpolation** ($cfg\_tools.f90$)
Then, inside the grid mixed, we can make a 3th-order accurate interpolation and write the new values.

4. **Create fine grid**($cfg\_tools.f90$)
At the end, we have to break the mixed grid toward four grids, one for each component T,U,V & F.

4.2 Define domain

This first step consist to define the size of the domain inside the initial grid (*nxcoag* and *nycoag*). Indeed, this domain may be regional or global. In the global case, the size of the domain is the same than the size of the initial grid (*nxsize* and *nysize*). On the other hand, in the regional case, we need to add one band of ghost cell all around the sub-domain defined by user to allow the interpolation along the borders. Moreover, in the regional case we are going to suppress all boundaries (no periodicity) and so we will obtain a matrix shape different without overlap bands. Thus, firstly, we have to determine if we are in the global or regional case ($create\_coordinates.f90$).

Extract 2. *Kind of sub-domain (create_ coordinates.f90)*

```
IF(nn_imin.EQ.nn_imax.AND.nn_jmin.EQ.nn_jmax) THEN
    nglobal = .TRUE.
    WRITE(*,*) 'Size of domain: GLOBAL'
ELSE
    nglobal = .FALSE.
    WRITE(*,*) 'Size of domain: REGIONAL'
ENDIF
```

For example, in global case (*nglobal* = $.TRUE.$), we will have:

Extract 3. *Definition of the domain in the global case (domain.f90)*

```
IF(nglobal) THEN
    !
    WRITE(*,*) ''
    WRITE(*,*) ' ** global ** '
    WRITE(*,*) ''
    !
    nn_imin = 1
    nn_imax = nsizex
    nn_jmin = 1
    nn_jmax = nsizey
    nxcoag = nsizex
    nycoag = nsizey
    !
...
```

In the simple case without the northern boundary ($nn\_jmin < nn\_jmax$), we will have:

Extract 4. *Definition of the domain when we don't cross the northern boundary (domain.f90)*

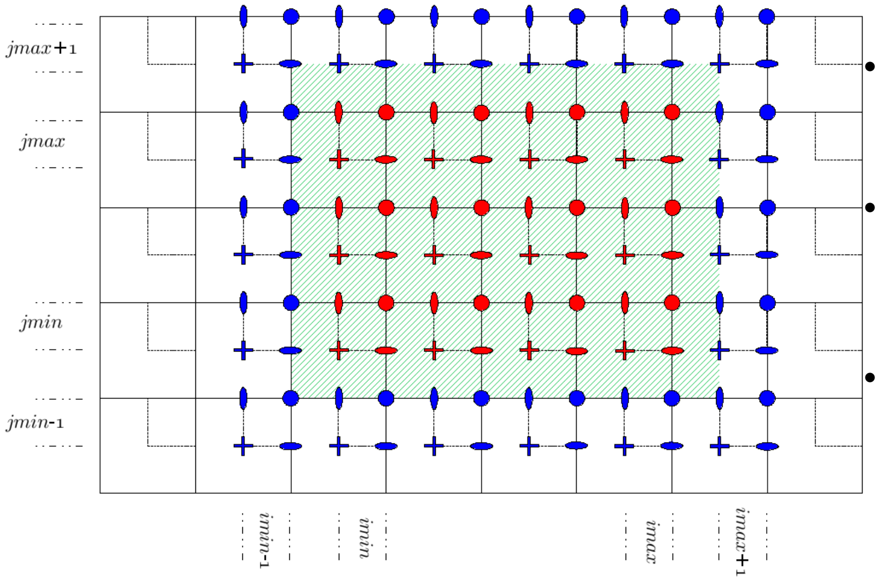
```
SUBROUTINE define_domain
  !
  WRITE(*,*) ''
  WRITE(*,*) ' ### SUBROUTINE define_domain ### '
  WRITE(*,*) ''
  !
  ! *** WITHOUT NORTH BOUNDARY ***
  IF((nn_jmax.LT.(nsizey-1)).AND.(nn_jmax.GT.nn_jmin)) THEN
    !
    WRITE(*,*) ' ****************************** '
    WRITE(*,*) ' *** WITHOUT NORTH BOUNDARY *** '
    WRITE(*,*) ' ****************************** '
    !
    ! *** WITH EAST/WEST BOUNDARY ***
    IF(nn_imin.GT.nn_imax) THEN
      nxcoag = (nsizex - (nn_imin-1) + 1) + ((nn_imax+1) - 2)
    ! *** ALL AROUND THE EARTH ***
    ELSEIF(nn_imin.EQ.nn_imax) THEN
      nxcoag = nsizex
    ELSE
      nxcoag = (nn_imax+1) - (nn_imin-1) + 1
    ENDIF
    !
    !(+/-1) we need ghost cells to make interpolation
    nycoag = (nn_jmax+1) - (nn_jmin-1) + 1
    !
ENDIF
!
...
```

We see that there are several scenarios depending on whether we cross the borders of the matrix ($nn\_imin \geq nn\_imax$) or not.

4.3 Creation of the mixed grid

4.3.1 Define and writting

We have to take into account the refinement coefficient to define the size of the mixed grid. To allow the interpolation inside this same grid along the boundaries we add one band all around (see figure 4.1). We want to obtain a single grid whose the size of the components is ($nxgmix \times nygmix$). The input grid is of kind *coordinates* and the mixed grid is of kind *mixed_coodinates* (see extract 6).



- **Red** → Domain defined by user (*nn_imin*, *nn_imax*, *nn_jmin and nn_jmax*). (4×3)
- **Blue** → Add of one ghost cell band to make interpolation along the borders. $nxcoag = 6, nycoag = 5$ $nxgmix, nygmix$.
- **Green** → Area to interpolate new points $(nxfine \times nyfine)$

Figure 4.1: *Building regional mixed grid*

Extract 5. *Definition of the size of the mixed grid (mixed_grid.f90)*

```
!************************************
!!!Calculate size of mixed grid (ixgmix x iygmix)
!************************************
IF(.NOT.nglobal)THEN
  ixgmix = (nxcoag) * 2
  ixgmix = ixgmix + (nn_rhox-1)*(ixgmix-1)
  !
  iygmix = (nycoag) * 2
  iygmix = iygmix + (nn_rhoy-1)*(iygmix-1)
  !
ELSEIF(nglobal) THEN
  ixgmix = (nxcoag) * 2
  ixgmix = ixgmix + (nn_rhox-1)*(ixgmix)
  !
  iygmix = (nycoag) * 2
  iygmix = iygmix + (nn_rhoy-1)*(iygmix)
ENDIF
!
nxgmix = ixgmix
nygmix = iygmix
!
...
```

Extract 6. *Content of different formats used (types.f90)*

```
TYPE coordinates
  REAL*8, DIMENSION(:,:), POINTER  :: nav_lon,nav_lat              => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: glamv, glamu, glamt, glamf   => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: gphit, gphiu, gphiv, gphif   => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: e1t, e1u, e1v, e1f           => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: e2t, e2u, e2v, e2f           => NULL()
  INTEGER, DIMENSION(:) , POINTER  :: time_steps                   => NULL()
END TYPE coordinates
!
TYPE mixed_coordinates
  REAL*8, DIMENSION(:,:), POINTER  :: nav_lon,nav_lat              => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: glam                         => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: gphi                         => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: e1                           => NULL()
  REAL*8, DIMENSION(:,:), POINTER  :: e2                           => NULL()
  INTEGER, DIMENSION(:) , POINTER  :: time_steps                   => NULL()
END TYPE mixed_coordinates
!
...
```

- $glam\mathbf{t}(nxcoag) + glam\mathbf{u}(nxcoag) + glam\mathbf{v}(nxcoag) + glam\mathbf{f}(nxcoag) \Longrightarrow glam(nxgmix)$
- $gphi\mathbf{t}(nycoag) + gphi\mathbf{u}(nycoag) + gphi\mathbf{v}(nycoag) + gphi\mathbf{f}(nycoag) \Longrightarrow gphi(nygmix)$
- $e1\mathbf{t}(nxcoag) + e1\mathbf{u}(nxcoag) + e1\mathbf{v}(nxcoag) + e1\mathbf{f}(nxcoag) \Longrightarrow e1(nxgmix)$
- $e2\mathbf{t}(nycoag) + e2\mathbf{u}(nycoag) + e2\mathbf{v}(nycoag) + e2\mathbf{f}(nycoag) \Longrightarrow e2(nygmix)$

In agreement with figure 4.1, we can define several variables. For example, in the simplest case we will have:

- $nxcoag = (nn\_imax{+}1)$ - $(nn\_imin$-$1) + 1$ — number of coarse grid **cells** along x-axis
- $nycoag = (nn\_jmax{+}1)$ - $(nn\_jmin$-$1) + 1$ — number of coarse grid **cells** along y-axis
- $nxgmix = 2 \times nxcoag$ — number of **known points** along x-axis (T+U and V+F)
- $nygmix = 2 \times nycoag$ — number of **known points** along y-axis (T+V and U+F)
- $nxfine = nxgmix + [(rho\text{-}1) \times (nxgmix\text{-}1)]$ — we add the **points to interpolate** along x-axis
- $nyfine = nygmix + [(rho\text{-}1) \times (nygmix\text{-}1)]$ — we add the **points to interpolate** along y-axis

The simplest case (see figure 4.1) is to build a fine grid inside an area which have no boundary as above. But, somes cases needs special treatment. Indeed if $nn\_imax < nn\_imax$, we will cross the boundaries left and right of the matrix. If $nn\_jmin > nn\_jmax$, we will cross the northern boundary.

4.3.2 Simple extraction

If we want a simple extraction, without changing the resolution, we must set the coefficients of refinement equal to one in the nesting parameters ($nn\_rhox = nn\_rhoy = 1$). There is no interpolation.

4.3.3 With overlap bands

In this case ($nn\_imin \geq nn\_imax$), we have to suppress the two recovering bands when we define the domain, and thus, we have (see extract 4):

- $nxcoag = [nsizex$ - $(nn\_imin\text{-}1){+}1] + [(nn\_imax{+}1)\text{-}2]$ — $nsizex$ is the x-size of the coarse grid
- $nycoag = [nsizey$ - $(nn\_jmin\text{-}1){+}1] + [(nn\_jmax{+}1)\text{-}2]$ — $nsizey$ is the y-size of the coarse grid

4.3.4 Crossing Northern boundary

In this case ($nn\_jmin \geq nn\_jmax$), method is to create the new matrix with two parts taken from each side of the northern border. For this, there are several scenarios:

- with one bipole (see figure 3.5)
- with two bipoles
- without bipole: between wich bipole? (from Canada to Russia or from Russia to Canada)

4.3.5 Global

This case is interesting when we want to increase the global resolution. For that, user needs to indicate $nn\_imin = nn\_imax$ and $nn\_jmin = nn\_jmax$ in the namelist (see extract 1). If $nn\_rho > 1$ we obtain:

- $nxcoag = nsizex$
- $nycoag = nsizey$
- $nxgmix = (nxcoag{\times}2) + [(nn\_rhox\text{-}1){\times}((nxcoag{\times}2)\text{-}1)]$
- $nygmix = (nycoag{\times}2) + [(nn\_rhoy\text{-}1){\times}((nycoag{\times}2)\text{-}1)]$
- $nxfine = nxgmix$ - $\dfrac{4 \times (nn\_rhox - 1))}{2}$
- $nyfine = \dfrac{nygmix - (2 \times nn\_rhoy)}{2}$ - nn_rhoy

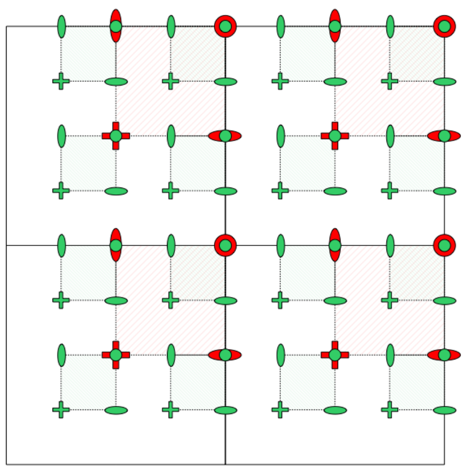
4.3.6 Summary

There is no major problem to build a mixed grid since we retrieve the values already existing. The main difficulty is to allow all scenarios.

4.4 Interpolation

To calculate all the news points we have to interpolate. For this, we use a 3th-order polynomial interpolation (see chapter 2). As we want to interpolate within a 2D matrix, we first interpolate along longitude and then along latitude. So, before each loop, we calculate the coefficients of interpolation and store them in a 1D array. These coefficients are the same along longitude and the same along latitude. Thus, they are calculated only one time for each.

4.4.1 Even interpolation



red → points of coarse grid

green → points of fine grid

Figure 4.2: *refinement coefficient equal 2*

With an even refinement factor,as we can see on the figure 4.2, all the red points (T,U,V and F coarse points) becomes green F points. Indeed, when the refinement coefficient (rho) is an even coefficient, the T,U,V, and F points of the coarse grid becomes F points in the fine grid. Here we see the interest of this method which consist to use all the known values and especially not to recalculate them.

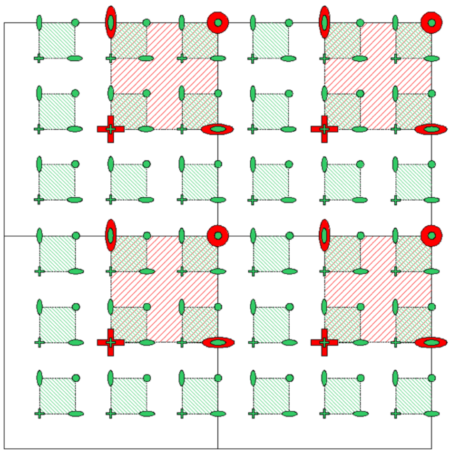
$T \rightarrow F$
$U \rightarrow F$
$V \rightarrow F$
$F \rightarrow F$

We have to be careful with this because the equator won't be align with the T/U-points but along the V/F-points. So, if the sub-domain is global or includes the equator area, we need to shift the grid along the j-indexation.

Extract 7. *We determine if we need to shift the grid to have equator line over the T-Upoints (create_coordinates.f90)*

```
!We want to fix equator along T and U-points
imodulo =  MOD(nn_rhoy,2)
!
IF(.NOT.nglobal
     .AND.(scoagrd%gphit(nn_imin,nn_jmin)*scoagrd%gphit(nn_imin,nn_jmax)).LT.0
     .AND.imodulo.EQ.0) THEN
  nequator = 1
ELSEIF(nglobal.AND.imodulo.EQ.0) THEN
  nequator = 1
ELSE
  nequator = 0
ENDIF
!
...
```

4.4.2 Odd interpolation



red → points of coarse grid

green → points of fine grid

Figure 4.3: *refinement coefficient equal 3*

With an odd refinement coefficient, as wee can see on the figure 4.3, the red points (T,U,V and F coarse points) stay the same points in the fine grid.

$T \rightarrow T$
$U \rightarrow U$
$V \rightarrow V$
$F \rightarrow F$

4.4.3 East/West discontinuity

In the input file, the range values for the longitude are [-180°:180°]. So, we have a discontinuity at ±180. When we interpolate on both side of this boundary, we must increase all negative values of 360. After the interpolation we substract 360 at the values larger than 180°.

Extract 8. *We check the ±180 discontinuity before interpolation (cfg_ tools.f90)*

```
!!!INTERPOLATION ALONG LONGITUDE-AXIS
WRITE(*,*) 'Interpolation along longitude'
!
DO jj=nn_rhoy,nygmix,nn_rhoy
  DO ji=nn_rhox,nxgmix,nn_rhox
    DO jk=1,nn_rhox-1
    !
    ! First, we check the +-180 discontinuity.
    ! In this case, we increase the negative values of 360.
    IF(ABS(smixgrd%glam(ji,jj)-smixgrd%glam(ji+1*nn_rhox,jj)).GT.180.0
    .AND.smixgrd%glam(ji,jj).LT.0.) THEN
      smixgrd%glam(ji,jj) = smixgrd%glam(ji,jj) + 360.
    ENDIF
    !
    IF(ABS(smixgrd%glam(ji,jj)-smixgrd%glam(ji+1*nn_rhox,jj)).GT.180.0
    .AND.smixgrd%glam(ji+1*nn_rhox,jj).LT.0.) THEN
      smixgrd%glam(ji+1*nn_rhox,jj) = smixgrd%glam(ji+1*nn_rhox,jj) + 360.
    ENDIF
    !
    IF(ABS(smixgrd%glam(ji+1*nn_rhox,jj)-smixgrd%glam(ji+2*nn_rhox,jj)).GT.180.0
    .AND.smixgrd%glam(ji+1*nn_rhox,jj).LT.0.) THEN
      smixgrd%glam(ji+1*nn_rhox,jj) = smixgrd%glam(ji+1*nn_rhox,jj) + 360.
    ENDIF
    !
    IF(ABS(smixgrd%glam(ji+1*nn_rhox,jj)-smixgrd%glam(ji+2*nn_rhox,jj)).GT.180.0
    .AND.smixgrd%glam(ji+2*nn_rhox,jj).LT.0.) THEN
      smixgrd%glam(ji+2*nn_rhox,jj) = smixgrd%glam(ji+2*nn_rhox,jj) + 360.
    ENDIF
    !
    IF(ABS(smixgrd%glam(ji+2*nn_rhox,jj)-smixgrd%glam(ji+3*nn_rhox,jj)).GT.180.0
    .AND.smixgrd%glam(ji+2*nn_rhox,jj).LT.0.) THEN
      smixgrd%glam(ji+2*nn_rhox,jj) = smixgrd%glam(ji+2*nn_rhox,jj) + 360.
    ENDIF
    !
    IF(ABS(smixgrd%glam(ji+2*nn_rhox,jj)-smixgrd%glam(ji+3*nn_rhox,jj)).GT.180.0
    .AND.smixgrd%glam(ji+3*nn_rhox,jj).LT.0.) THEN
      smixgrd%glam(ji+3*nn_rhox,jj) = smixgrd%glam(ji+3*nn_rhox,jj) + 360.
    ENDIF
!
...
!
WHERE(smixgrd%glam.GT.180)
  smixgrd%glam = smixgrd%glam - 360.0
ENDWHERE
!
```

4.4.4 The geographical North pole singularity

Nearby the geographical north pole, the variations of the longitude are too large compared with the grids resolution. We have to make a polar stereographic projection before making interpolation. We define λ_0 and ϕ_0 as the origine of the projection. For making the projection around the north pole, we take λ_0=0° and ϕ_0=90∘ which allow us to simplify these equations.

First, we make the projection (*projection.f*90) over a cartesian map $(\lambda, \phi) \rightarrow (x, y)$:

Extract 9. *Nearby the geographical north pole we make a stereographic projection (cfg_tools.f90)*

```
! Nearby the geographical north pole,
! the variation of the longitudes is too important.
! We need to make a polar stereographic projection before interpolation.
IF(.NOT.llnorth_pole
  .AND.ABS(smixgrd%glam(ji,jj)-smixgrd%glam(ji+3*nn_rhox,jj)).GE.80.0)THEN
  CALL stereo_projection(ji,jj,jk,llnorth_pole,1)
  jproj = 1
ENDIF
!
...
```

$$k = \frac{2R}{1+\sin\phi_0\sin\phi+\cos\phi_0\cos\phi\cos(\lambda-\lambda_0)} \quad\longrightarrow\quad k = \frac{2R}{1+\sin\phi}$$

$$\begin{aligned} x &= k\cos\phi\sin(\lambda-\lambda_0) &\longrightarrow\quad x &= k\cos\phi\sin(\lambda) \\ y &= k\left[\cos\phi_0\sin\phi-\sin\phi_0\cos\phi\cos(\lambda-\lambda_0)\right] &\longrightarrow\quad y &= k\left[\cos\phi_0\sin\phi-\sin\phi_0\cos\phi\cos(\lambda)\right] \end{aligned}$$

$$\begin{aligned} \rho &= \sqrt{x^2+y^2} \\ c &= 2\arctan\left(\frac{\rho}{2R}\right) \end{aligned}$$

When interpolation has been carried out, we come back into the original map $(x, y) \rightarrow (\lambda, \phi)$:

Extract 10. *After interpolation we make inverse projection*

```
! We make the polar stereographic projection reverse if needs.
IF(jproj.EQ.1)THEN
  CALL stereo_projection_inv(ji,jj,jk,llnorth_pole,1)
ENDIF
!
...
```

$$\begin{aligned} \lambda &= \lambda_0+\arctan\left(\frac{x\sin c}{\rho\cos\phi_0\cos c-y\sin\phi_0\sin c}\right) &\longrightarrow\quad \lambda &= \lambda_0+\arctan\left(\frac{x}{-y}\right) \\ \phi &= \arcsin\left(\cos c\sin\phi_0+\frac{y\sin c\cos\phi_0}{\rho}\right) &\longrightarrow\quad \phi &= \arcsin\left(\cos c\right) \end{aligned}$$

Along the northern boundary

As we said previously, the evolution of the longitudes along the northern boundary looks like a heavyside function nearby the geographical north pole. Thus, concerning the longitude we can't make interpolation along this line at this point.

Extract 11.

```
! If we are along north boundary,
! the variation of longitude looks like a heaviside fonction
! at the geographical north pole.
! Thus, we can't make an interpolation.
!
IF(ABS(smixgrd%glam(ji,jj) - smixgrd%glam(ji+3*nn_rhox,jj)).EQ.180.0)THEN
  llnorth_pole = .TRUE.
ENDIF
!
...
```

4.5 Creation of the fine grid

Now, we want to create the fine grid by scattering the mixed grid. Before we have to detremine the size of the fine grid ($nxfine$ and $nyfine$).

Extract 12. *Define the size of the fine grid (create_ coodinates.f90)*

```
!*************************************
!!! CALCULATE FINE GRID DIMENSIONS
!*************************************
IF(.NOT.nglobal) THEN
  IF(nn_rhox.EQ.1) THEN
    nxfine = nxcoag
  ELSE
    nxfine = (nxcoag-2)*nn_rhox + 1
  ENDIF
  !
  IF(nn_rhoy.EQ.1) THEN
    nyfine = nycoag
  ELSE
    nyfine = (nycoag-2)*nn_rhoy + 1
  ENDIF
  !
ELSEIF(nglobal) THEN
  IF(nn_rhox.GT.1) THEN
    nxfine = (nxgmix - 4*(nn_rhox-1))/2
  ELSEIF(nn_rhox.EQ.1) THEN
    nxfine = nsizex
ENDIF
!
IF(nn_rhoy.GT.1) THEN
```

```
  nyfine = (nygmix - (2*nn_rhoy))/2 - nn_rhoy + nequator
ELSEIF(nn_rhoy.EQ.1) THEN
  nyfine = nsizey
ENDIF
!
...
```

- $glam(nxgmix) \Longrightarrow glam\mathbf{t}(nxfine) + glam\mathbf{u}(nxfine) + glam\mathbf{v}(nxfine) + glam\mathbf{f}(nxfine)$
- $gphi(nygmix) \Longrightarrow gphi\mathbf{t}(nyfine) + gphi\mathbf{u}(nyfine) + gphi\mathbf{v}(nyfine) + gphi\mathbf{f}(nyfine)$
- $e1(nxgmix) \Longrightarrow e1\mathbf{t}(nxfine) + e1\mathbf{u}(nxfine) + e1\mathbf{v}(nxfine) + e1\mathbf{f}(nxfine)$
- $e2(nygmix) \Longrightarrow e2\mathbf{t}(nyfine) + e2\mathbf{u}(nycoag) + e2\mathbf{v}(nycoag) + e2\mathbf{f}(nyfine)$

Chapter 5

Tests and validation

5.1 Regional

Example 1: Equator and overlap bands

```
&coarse_grid_files
    cn_parent_coordinate_file = 'coordinates_ORCA_R2.nc'
    cn_position_pivot = 'T-grid'
/

&nesting
    nn_imin = 178
    nn_imax = 4
    nn_jmin = 72
    nn_jmax = 87
    nn_rhox  = 3
    nn_rhoy = 2
/
```

```
openps
filecoord='/usr/home/blelod/.../coordinates_ORCA_R2.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
glamv=read_ncdf('glamv',file=filecoord,/nostruct)
...
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
gphiv=read_ncdf('gphiv',file=filecoord,/nostruct)
gphif=read_ncdf('gphif',file=filecoord,/nostruct)
domdef,69,89,-3,13
plt,glamt,/realcont,/nocontour,/nofill,/nocolorbar,thick=1,/noportrait
vargrid='T'
tracegrille,/rmout,color=75,thick=2,xstride=1,ystride=1

filecoord='/usr/home/blelod/.../1_coordinates_ORCA_R2_rox3_roy2.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
```

```
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
glamv=read_ncdf('glamv',file=filecoord,/nostruct)
...
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
gphiv=read_ncdf('gphiv',file=filecoord,/nostruct)
gphif=read_ncdf('gphif',file=filecoord,/nostruct)
domdef,69,89,-3,13
plt,glamt,/realcont,/nocontour,/nofill,/nocolorbar,thick=1

vargrid='T'
tracegrille,/rmout,color=40,thick=2,xstride=1,ystride=1

vargrid='F'
tracegrille,/rmout,color=110,thick=2,xstride=1,ystride=1

closeps
printps
```

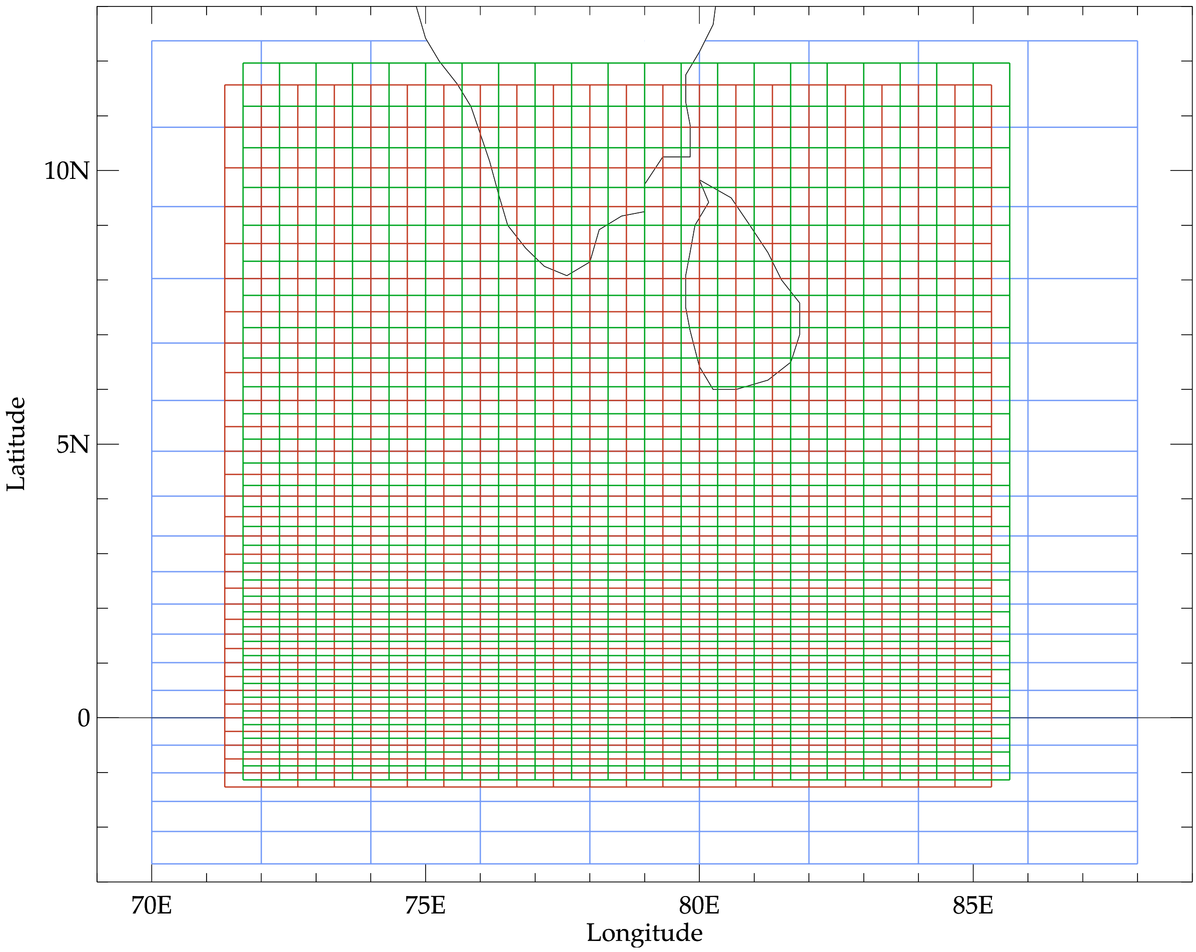


Figure 5.1: *left/right periodicity along equator area*
coarse T-grid - *fine T-grid* - *fine F-grid*

Example 2: Along the north boundary

```
&coarse_grid_files
    cn_parent_coordinate_file = 'coordinates_ORCA_R2.nc'
    cn_position_pivot = 'T-grid'
/

&nesting
    nn_imin = 52
    nn_imax = 55
    nn_jmin = 145
    nn_jmax = 145
    nn_rhox  = 2
    nn_rhoy = 2
/
```

```
openps
filecoord='/usr/home/blelod/.../coordinates_ORCA_R2.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
...
gphit=read_ncdf('gphit',file=filecoord,/nostruct)
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
...
domdef,-180,0,83,90
plt,glamt,...,/nocolorbar,/orthographic,thick=1,map=[40,180,0],/portrait
vargrid='F'
tracegrille,/rmout,color=75,thick=1,xstride=1,ystride=1

filecoord='/usr/home/blelod/.../1_coordinates_ORCA_R2_north_rox2_roy2.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
...
gphit=read_ncdf('gphit',file=filecoord,/nostruct)
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
...
domdef,-180,0,83,90
plt,glamt,...,/nocolorbar,,/orthographic,thick=1,map=[40,180,0],/noerase

vargrid='U'
tracegrille,/rmout,color=0,thick=2,xstride=1,ystride=1
vargrid='V'
tracegrille,/rmout,color=180,thick=2,xstride=1,ystride=1

closeps
printps
```

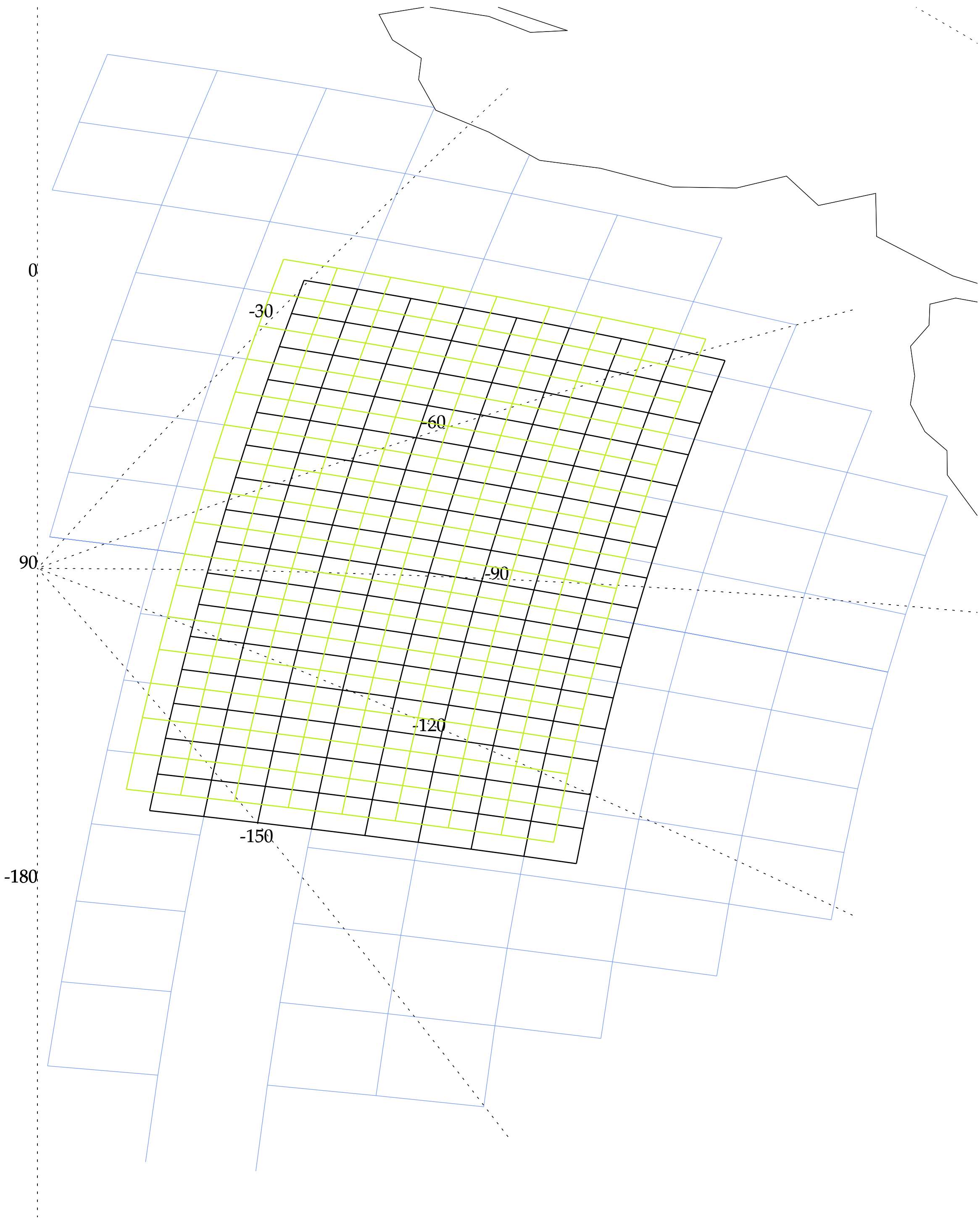


Figure 5.2: *Along northern boundary and ±180°discontinuity*
coarse F-grid - *fine U-grid* - *fine V-grid*

Example 3: Extraction in the arctic region

```
&coarse_grid_files
    cn_parent_coordinate_file = 'coordinates_ORCA_R2.nc'
    cn_position_pivot = 'T-grid'
/

&nesting
    nn_imin = 110
    nn_imax = 155
    nn_jmin = 144
    nn_jmax = 140
    nn_rhox  = 1
    nn_rhoy = 1
/
```

```
openps
filecoord='/usr/home/blelod/.../coordinates_ORCA_R2.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
...
gphit=read_ncdf('gphit',file=filecoord,/nostruct)
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
...
domdef,-180,180,63,90
plt,glamt,...,/nocolorbar,/orthographic,thick=1,map=[40,180,0],/nolandscape
vargrid='T'
tracegrille,/rmout,color=75,thick=1,xstride=1,ystride=1

filecoord='/usr/home/blelod/.../1_coordinates_ORCA_R2_north_rox1_roy1.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
...
gphit=read_ncdf('gphit',file=filecoord,/nostruct)
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
...
domdef,-180,180,63,90
plt,glamt,...,/nocolorbar,,/orthographic,thick=1,map=[40,180,0],/noerase

vargrid='T'
tracegrille,/rmout,color=40,thick=2,xstride=1,ystride=1

closeps
printps
```

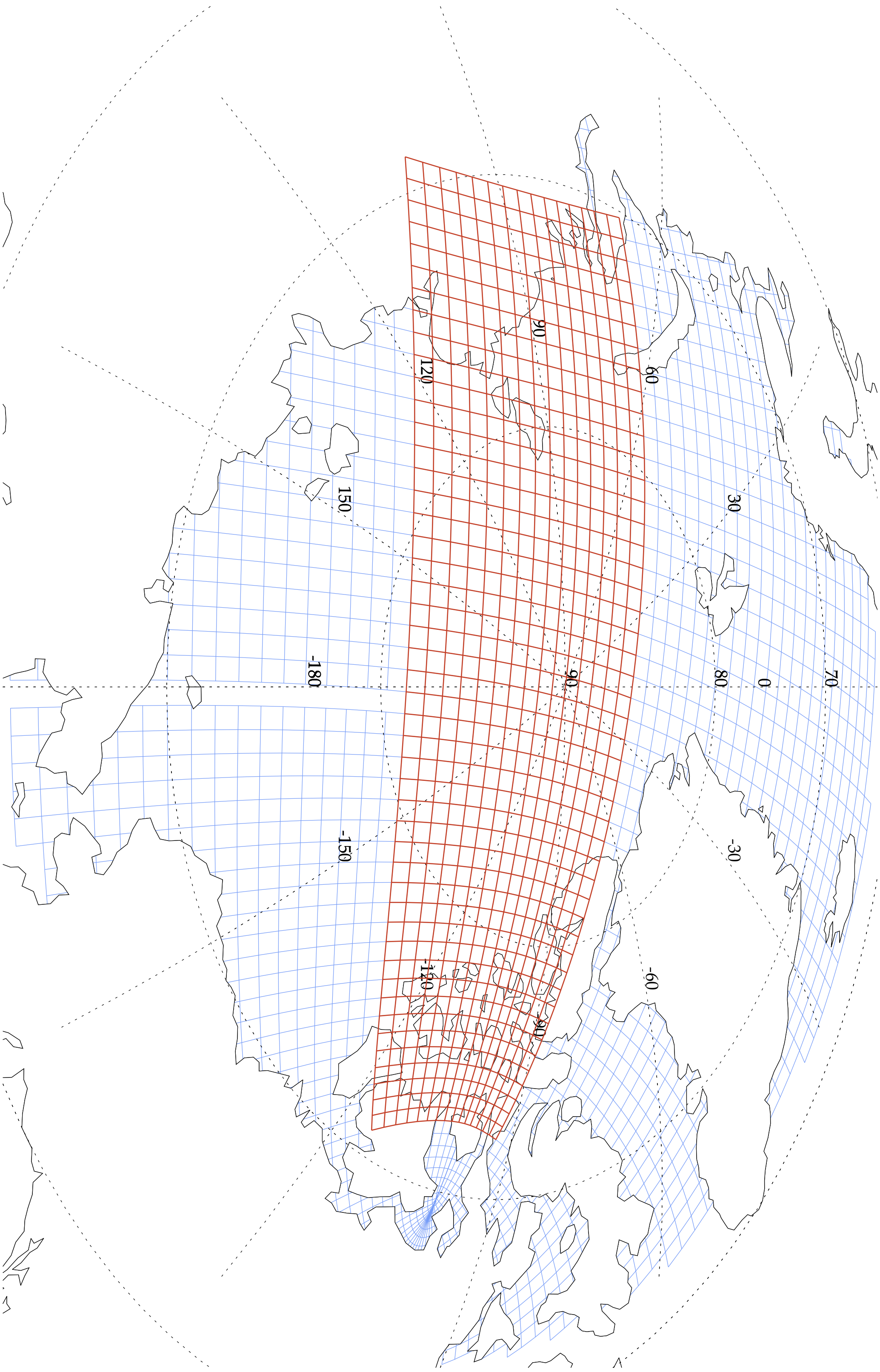


Figure 5.3: *Extraction in the arctic region*
coarse T-grid - *fine T-grid*

Example 4: The geographical north pole

```
&coarse_grid_files
    cn_parent_coordinate_file = 'coordinates_ORCA_R2.nc'
    cn_position_pivot = 'T-grid'
/

&nesting
    nn_imin = 49
    nn_imax = 52
    nn_jmin = 147
    nn_jmax = 147
    nn_rhox  = 2
    nn_rhoy = 2
/
```

```
openps
filecoord='/usr/home/blelod/.../coordinates_ORCA_R2.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
...
gphit=read_ncdf('gphit',file=filecoord,/nostruct)
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
...
domdef,-180,180,87,90
plt,glamt,...,/nocolorbar,/orthographic,thick=1,map=[40,180,0],/nolandscape
vargrid='T'
tracegrille,/rmout,color=75,thick=1,xstride=1,ystride=1

filecoord='/usr/home/blelod/.../1_coordinates_ORCA_R2_northpole.nc'
initncdf, filecoord,/fullcgrid
glamt=read_ncdf('glamt',file=filecoord,/nostruct)
glamu=read_ncdf('glamu',file=filecoord,/nostruct)
...
gphit=read_ncdf('gphit',file=filecoord,/nostruct)
gphiu=read_ncdf('gphiu',file=filecoord,/nostruct)
...
domdef,-180,180,87,90
plt,glamt,...,/nocolorbar,,/orthographic,thick=1,map=[40,180,0],/noerase

vargrid='T'
tracegrille,/rmout,color=40,thick=2,xstride=1,ystride=1
vargrid='F'
tracegrille,/rmout,color=110,thick=2,xstride=1,ystride=1

closeps
printps
```

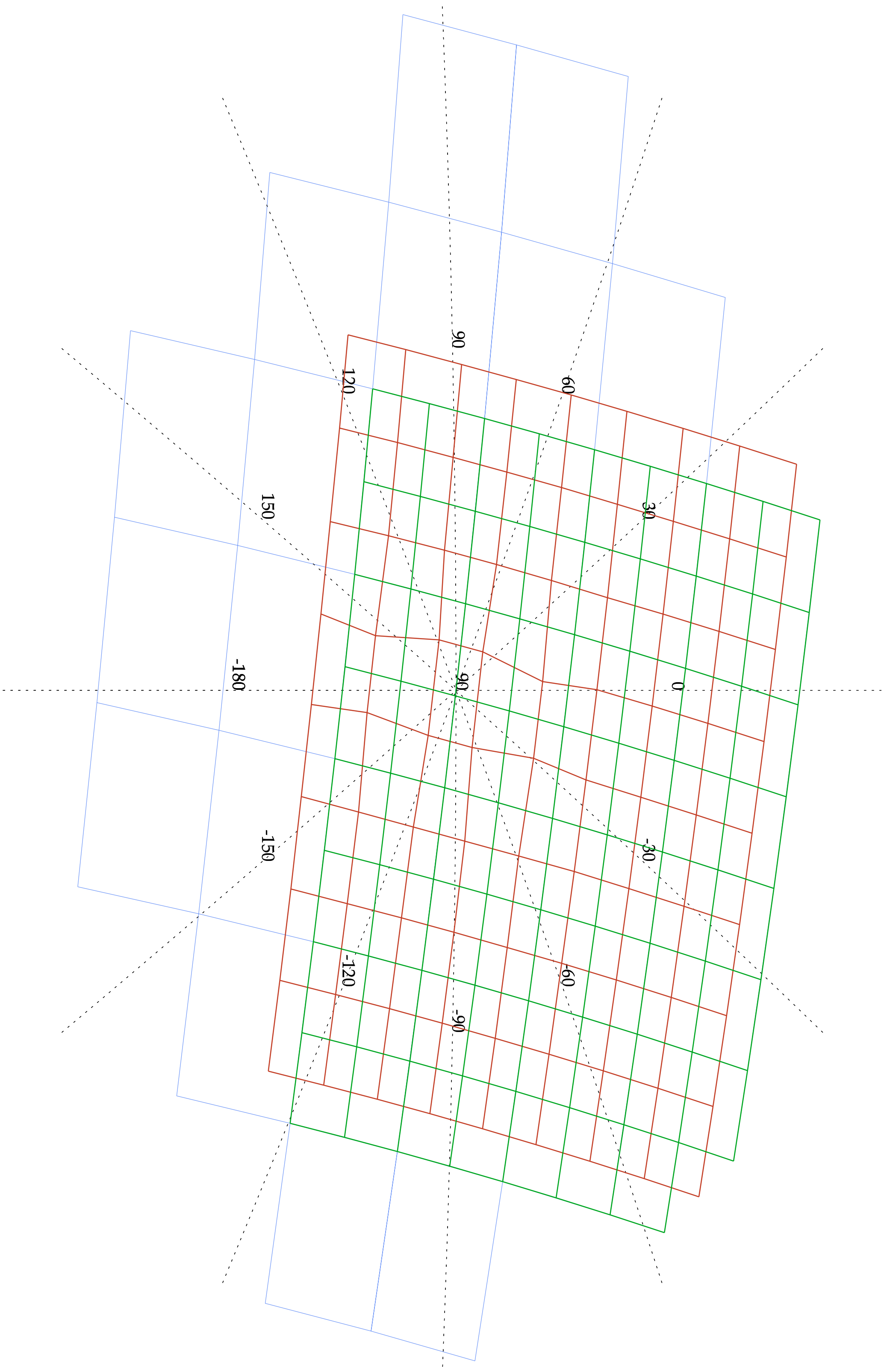


Figure 5.4: *Geographical north pole without stereographic projection*
coarse T-grid - *fine T-grid* - *fine F-grid*

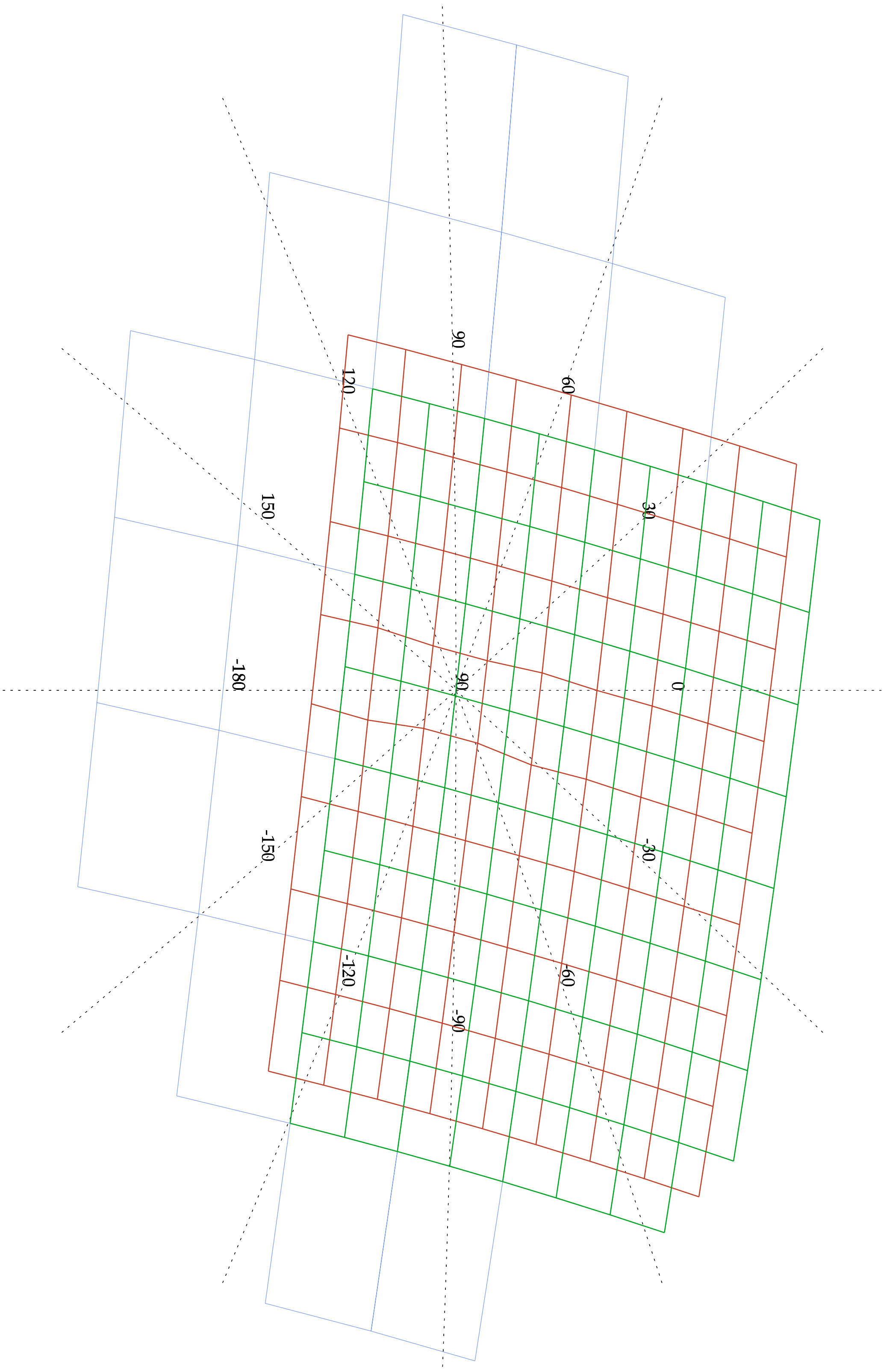


Figure 5.5: *Geographical north pole with stereographic projection*
coarse T-grid - *fine T-grid* - *fine F-grid*

5.2 Global

5.2.1 Test for reproductibility for ORCA2 grid

Size of the input grid: **182×149**			
nn_rhox	6	2 → 3	3 → 2
nn_rhoy	6	2 → 3	3 → 2
$nxgmix$	2184	728 → 2172	1092 → 2168
$nygmix$	1788	596 → 1770	894 → 1764
$nxfine$	1082	362 → 1082	542 → 1082
$nyfine$	882	295 → 879	441 → 879

$j = 1$		long	lat	long	lat	long	lat	long	lat	long	lat
Grid T	$1 = n - 1$	78.83	-77.98	78.5	-77.98	78.83	-77.84	78.66	-77.91	78.83	-77.84
	$2 = n$	79.16	-77.98	79.5	-77.98	79.16	-77.84	79.33	-77.91	79.16	-77.84
Grid U	$1 = n - 1$	79	-77.98	79	-77.98	79	-77.84	79	-77.91	79	-77.84
	$2 = n$	79.33	-77.98	80	-77.98	79.33	-77.84	79.66	-77.91	79.33	-77.84
Grid V	$1 = n - 1$	78.83	-77.94	78.5	-77.87	78.83	-77.80	78.66	-77.84	78.83	-77.80
	$2 = n$	79.16	-77.94	79.5	-77.87	79.16	-77.80	79.33	-77.84	79.16	-77.80
Grid F	$1 = n - 1$	79	-77.94	79	-77.87	79	-77.80	79	-77.84	79	-77.80
	$2 = n$	79.33	-77.94	80	-77.87	79.33	-77.80	79.66	-77.84	79.33	-77.80

5.3

5.2.2 Test for check the periodicity of the overlap bands

Table 5.1: *Global test for ORCA05 grids*

Size of the input grid: **722×511**																
nn_rhox	1		2		3		4		3		3		4		1	
nn_rhoy	1		2		3		4		1		2		2		2	
$nxmix$	1444		2888		4332		5776		4332		4332		5776		1444	
$nymix$	1022		2044		3066		4088		1022		2044		2044		2044	
$nxfine$	722		1442		2162		2882		2162		2162		2882		722	
$nyfine$	511		1020		1530		2040		511		1020		1020		1020	
Check the periodicity with overlap bands (longitude)																
longitude	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**
col(1) = col(n-1)	72.75	**73**	72.87	**73**	72.92	**73**	72.94	**73**	72.92	**73**	72.92	**73**	72.94	**73**	72.75	**73**
col(2) = col(n)	73.25	73.5	73.12	73.25	73.08	73.17	73.06	73.12	73.08	73.17	73.08	73.17	73.06	73.12	73.25	73.5

Table 5.2: *Global test for ORCA2 grids*

Size of the input grid: **182×149**																
nn_rhox	1		2		3		4		3		3		4		1	
nn_rhoy	1		2		3		4		1		2		2		2	
$nxgmix$	364		728		1092		1456		1092		1092		1456		364	
$nygmix$	298		596		894		1192		298		596		596		596	
$nxfine$	182		362		542		722		542		542		722		182	
$nyfine$	149		296		444		592		149		296		296		296	
Check the periodicity with overlap bands																
longitude	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**
col(1) = col(n-1)	78	**79**	78.5	**79**	78.67	**79**	78.75	**79**	78.67	**79**	78.67	**79**	78.75	**79**	78	**79**
col(2) = col(n)	80	81	79.5	80	79.33	79.67	79.25	79.5	79.33	79.67	79.33	79.67	79.25	79.5	80	81

List of Figures

List of Tables