C.E.R.F.A.C.S

Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique

DESCRIPTION DU BANC D'ESSAI PERMETTANT DE QUANTIFIER LA QUALITE DES INTERPOLATIONS AVEC LE COUPLEUR OASIS4

Laure Coquart, Sophie Valcke

Octobre 2007, TR/GLOBC/07/102

Table des matières

1 Introduction

La nouvelle version du coupleur océan-atmosphère OASIS4 (Ocean, Atmosphere, Sea, Ice, Soil) du CERFACS [2], utilisée pour la modélisation du système climatique global, est très différente des précédentes versions (OASIS3) jusque là utilisées.

La librairie de communication PSMILe et le code OASIS, qui gèrent l'échange des données, les interpolations et la redistribution parallèle des données entre les différents modèles, ont été entièrement réécrits de manière à ce que le coupleur, qui traite des composantes de modèles climatiques parallèles, soit lui-même parallèle. Cela doit permettre d'éviter les problèmes de goulot d'étranglement et de pertes de performance au niveau du coupleur, notamment du fait de la parallélisation de plus en plus poussée des modèles climatiques.

La librairie de communication PSMILe permet, comme pour OASIS3, de tourner soit avec la librairie MPI1 (mode statique) soit la librairie MPI2 (mode dynamique). Par contre, elle traite de façon parallèle l'échange de données entre deux grilles identiques mais partitionnées différemment, ou l'échange de données entre deux grilles différentes.

Une autre fonctionnalité du coupleur, fondamentale pour le couplage de plusieurs modèles numériques définis sur des grilles différentes, est l'interpolation des champs de couplage aux interfaces (air-mer ou glace-mer ...) puisqu'elle permet d'exprimer sur la grille du modèle cible l'information apportée par le modèle source mais sur sa propre grille. Elle est réalisée de façon parallèle par le Transformer dans le code OASIS. La recherche des voisins, qui consiste à déterminer les points de la grille source localisés autour de chaque point cible pour pouvoir interpoler les champs de couplage en ce point, est réalisée en parallèle d'une manière globale par les processeurs sur lesquels est décomposée la composante du modèle source.

Ce rapport décrit l'environnement qui a été mis au point pour tester et valider tous les types d'interpolations entre tous les types de grilles actuellement traités par OASIS4.

La première partie du rapport présente le fonctionnement du banc d'essai pour tester les différentes interpolations entre n'importe quelle grille, et ce de façon la plus générique possible. La seconde partie présente l'utilisation en

pratique de l'environnement qui a été construit, en expliquant, par exemple, le format des données à utiliser et les différentes options proposées pour selectionner la configuration que l'on désire tester. Enfin, la dernière partie présente le calcul de l'erreur d'interpolation, lors de l'échange de différentes fonctions analytiques trigonométriques tests, entre des grilles identiques et entre des grilles différentes.

2 Description de l'environnement permettant de tester la qualité d'interpolation dans OASIS4

Cette section décrit l'environnement mis au point pour tester "off-line", c'est-à-dire sans faire tourner et coupler de vrais modèles, la qualité d'interpolation entre deux grilles : la grille source (qui envoie le champ) et la grille cible (qui le reçoit).

Les trois types standards (ou génériques) de grilles sont respectivement : "longitude-latitude", "logiquement rectangle" et "gaussienne réduite".
Dans la suite, elles seront désignées respectivement par REGLONLATVRT, IRRLONLAT_REGVRT et GAUSSREDUCED_REGVRT, qui correspondent aux options utilisées dans les codes.

Une grille de type REGLONLATVRT est régulière en longitude, en latitude et selon la verticale. La longitude des points est une fonction uniquement de l'indice i (lon(i)), la latitude des points une fonction de l'indice j uniquement (lat(j)) et la verticale des points une fonction de k uniquement (z(k)).

Une grille IRRLONLAT_REGVRT est curviligne dans le plan (lon, lat) et régulière selon la verticale z. La longitude et la latitude des points sont des fonction des indices i et j (lon(i,j), lat(i,j)), tandis que la verticale des points reste une fonction de k uniquement (z(k)).

Pour une grille de type GAUSSREDUCED_REGVRT, les points sont repérés par un seul indice dans le plan (lon, lat), npt_hor, qui décrit le plan par bandes de latitudes. La grille est aussi régulière selon z. La longitude et la latitude des points ne dépendent donc que de l'indice npt_hor (lon(npt_hor), lat(npt_hor)), tandis que la verticale des points reste une fonction de k uniquement (z(k)).

Chaque grille est supposée représenter au mieux un domaine physique et elle est donc munie d'un masque permettant de modéliser de manière la plus réaliste possible la zone calculée (océan, terres, atmosphère ...).

Le banc d'essai est constitué de deux programmes source.F90 et target.F90. Ces programmes lisent respectivement le fichier NetCDF grid_source.nc contenant la définition de la grille source et le fichier grid_target.nc contenant celle de la grille cible. Chaque fichier de données a été écrit de façon à ne

dépendre que du type de la grille considérée, et la lecture se fait uniquement à partir des attributs des variables. L'écriture des données et les attributs choisis sont décrits plus en détail dans la partie suivante. Par ailleurs, les deux noms de fichiers, grid_source.nc et grid_target.nc, étant complètement génériques, il suffit de les faire pointer à chaque fois vers les grilles que l'on désire tester.

Les deux fichiers source_source_smioc.xml.in et target_target_smioc.xml.in sont les fichiers de configuration du couplage pour chaque modèle. Ils contiennent le type et la description des grilles de chaque composante du modèle source ou du modèle target, les fichiers d'entrée et de sortie nécessaires au couplage, ainsi que le type d'interpolation pour le modèle cible, qui reçoit un champ.

C'est le choix des options, imposées par des clés de précompilation au niveau du Makefile, dans les fichiers source.F90, target.F90, source_source_smioc.xml.in et target_target_smioc.xml.in, qui permet de fixer la configuration à tester.

La structure des deux fichiers source.F90 et target.F90 est très similaire. Dans chaque cas, on commence par initialiser l'environnement MPI nécessaire au couplage et au parallélisme éventuel du code. On lit ensuite la grille de calcul et son masque. Puis, en tenant compte du nombre de processeurs demandés, chaque domaine de calcul est partitionné selon la direction j. C'est la parallélisation utilisée par défaut car elle est valable quelle que soit le type de grille considéré (REGLONLATVRT, IRRLONLAT_REGVRT et GAUSSREDUCED_REGVRT). Les variables décrivant les coordonnées, les coins et le masque de la grille peuvent ensuite être initialisées.

Pour tester la qualité de l'interpolation entre la grille source et la grille cible, le programme source.F90 calcule une fonction analytique sur les points de la grille source et envoie ce champ de couplage test unique, grâce à la librairie de communication PSMILe, au programme target.F90, qui le reçoit après interpolation par OASIS4. Après réception, le programme target.F90 calcule l'erreur entre le champ reçu après interpolation et la fonction analytique calculée sur les points de la grille cible. L'analyse de l'erreur permet alors de quantifier la qualité de l'interpolation.

Trois fonctions analytiques trigonométriques, de variations spatiales très diffrentes, ont été utilisées pour tester la qualité d'interpolation entre les différentes grilles :

$$f1(i,j,k) = 2. - cos(\pi(\frac{(acos(cos(lat(i,j,k))\ cos(lon(i,j,k)))}{1.2\pi})) \quad (1)$$

$$f2(i,j,k) = 2. + cos(lat(i,j,k))^2\ cos(2\ lon(i,j,k)) \quad (2)$$

$$f3(i,j,k) = 2. + sin^{16}(2\ lat(i,j,k))\ \ cos(16\ lon(i,j,k)) \quad (3)$$

Ce choix permet, entre autres, de tester la qualité de l'interpolation dans les zones de fort ou faible gradient. Elles sont tracées sur la figure (1) sur une grille régulière. La fonction f1 est représentée en haut à gauche, la fonction f2 en haut à droite et la fonction f3 en bas à droite.

L'erreur d'interpolation est calculée sur la grille cible par la formule :

$$E(i,j,k) = \frac{f^{cible}_{recue}(i,j,k) - f^{cible}_{ana}(i,j,k)}{f^{cible}_{ana}(i,j,k)} \quad (4)$$

Ces programmes ont été conçus de façon à pouvoir tourner en mono-processeur (un processus par programme) ou en parallèle (plusieurs processus par programme) et donc à tester les fonctionnalités d'interpolation d'OASIS4 en parallèle.

La partie qui suit décrit l'utilisation du banc d'essai en pratique, ainsi que toutes les normes qui ont été choisies en se basant sur les conventions CF (Climate and Forecast) existantes.

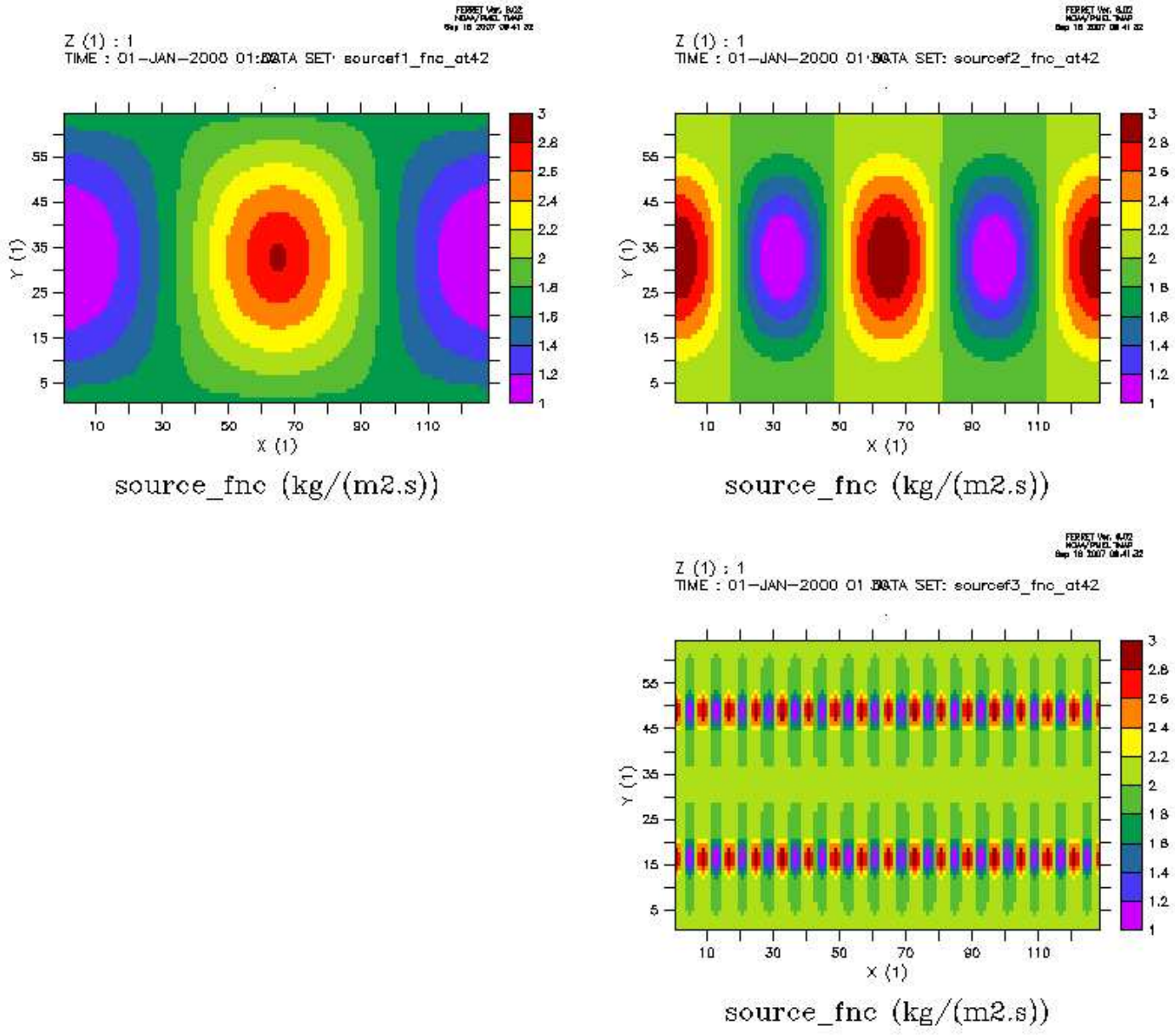

FIG. 1 – Fonctions analytiques f1, f2 f3 représentées sur une grille régulière

3 Utilisation pratique du banc-d'essai pour tester les interpolations dans OASIS4

De manière à pouvoir tester le plus facilement possible n'importe quelle interpolation entre n'importe quelles grilles avec OASIS4, les programmes source.F90 et target.F90 ainsi que les fichiers de configuration du couplage source_source_smioc.xml.in et target_target_smioc.xml.in ont été construits de façon la plus simple et la plus générique possible.

Ce sont les clés de précompilation, définies dans le Makefile, qui permettent ensuite de déterminer l'interpolation que l'on veut tester, sans avoir à retoucher aux programmes Fortran sources ou aux fichiers de configuration.

3.1 Format des données pour l'écriture des grilles source et cible

On rappelle que le type de grille est défini par l'une des options suivantes :

```
REGLONLATVRT
IRRLONLAT_REGVRT
GAUSSREDUCED_REGVRT
```

C'est cette clé de précompilation qui va permettre de sélectionner le type de la grille source et de la grille cible. La lecture des données, qui est différente pour chaque type de grille, découle directement de la clé choisie.

La définition des variables qui a été choisie pour écrire les données correspond à celle utilisée dans le coupleur OASIS4 [2].

Les fichiers NetCDF grid_source.nc et grid_target.nc contiennent les longitudes, les latitudes et les verticales des points de chaque grille. On stocke également les longitudes et les latitudes et verticales des quatre coins de chaque cellule de la grille (fondamentales pour l'interpolation), ainsi que le masque associé à la grille. Dans OASIS4, le masque est une variable logique tandis que dans le fichier de données il est stocké sous forme d'entiers 0 et 1, 0 correspondant à un point non valide (false) et 1 correspondant à un point valide (true). La conversion se fait au moment de la lecture des données.

Toutes les grilles associées aux champs de couplage sont bidimensionnelles (2D) dans l'espace longitude-latitude. OASIS4 ne traitant que des champs de couplage 3D, une cellule supplémentaire, centrée sur 0, est définie selon

la dimension verticale (z) pour chaque grille. On suppose que la grille est régulière cette direction, et donc indépendante de la longitude et de la latitude et que :

```
z(k=1) = 0
z_coin(k=1,1) = 1.
z_coin(k=1,2) = -1.
```

3.1.1 Conventions NetCDF Climate and Forecast (CF) pour les Meta-données et normes utilisées dans le banc d'essai

Les conventions CF utilisées pour écrire les fichiers de données NetCDF, utilisés entre autre dans la modélisation du système climatique, sont rappelées dans le document [1].

Ce sont les attributs décrivant les variables qui permettent de les caractériser. Certains des attributs sont obligatoires, d'autres non. Par exemple, pour les coordonnées, l'attribut *standard_name* qui contient le nom de la variable est optionnel tandis que l'attribut *units* est obligatoire et **standardisé**. C'est ainsi que pour la longitude, la latitude et la verticale on doit avoir :

```
pour la longitude : units = "degrees_east"
pour la latitude : units = "degrees_north"
pour la verticale : units = "meters"
```

C'est la norme que nous avons adopté pour l'écriture de nos données.

Dans la recherche des voisins pour réaliser les interpolations, il est nécessaire d'avoir les coordonnées des coins de chaque cellule du maillage. Nous avons donc rajouté les longitudes, les latitudes et la verticale des coins de chaque cellule de la grille dans les fichiers de données. Comme il n'existe pas d'attribut *units* standardisé pour ces variables, c'est en leur imposant par nous-même un attribut *local_name* bien défini que nous les avons caractérisées :

```
pour la longitude des coins : local_name = "cornerslon"
pour la latitude des coins : local_name = "cornerslat"
pour la verticale des coins : local_name = "cornersz"
```

Dans le cadre d'une écriture des fichiers de données la plus générique possible, le masque de la grille est stocké avec l'ensemble des coordonnées. Comme c'est un entier, sans unités, son attribut *units = "unitsless"* ne permet pas de le caractériser. C'est ainsi que nous avons défini par nous-même un attribut *local_name* :

pour le masque : local_name = "mask"

Nous avons fait le choix que les données relatives aux grilles soient écrites en double précision, sauf pour le masque bien évidemment puisque qui est un entier, et c'est sous cette hypothèse qu'elles sont relues dans les codes Fortran.

L'écriture exacte des fichiers de données en fonction du type de grille est explicitée ci-dessous.

3.1.2 Entête du fichier NetCDF d'une grille REGLONLATVRT

Pour une grille régulière, la longitude des points ne dépend que de i, la latitude des points ne dépend que de j et la verticale ne dépend que de k. Nous avons choisi la convention suivante pour l'écriture des données d'une telle grille (ici la grille nommée AT42) :

```
netcdf grid_at42 {
dimensions:
        i = 128 ;
        j = 64 ;
        k = 1 ;
        corners_ij = 2 ;
        corners_k = 2 ;
        character = 18 ;
variables:
        char prism_gridtype(character) ;
        double lon(i) ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
                lon:periodic = "true" ;
                lon:overlap = "0" ;
        double lat(j) ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
        double z(k) ;
                z:units = "meters" ;
                z:standard_name = "vertical" ;
        double clo(corners_ij, i) ;
                clo:local_name = "cornerslon" ;
        double cla(corners_ij, j) ;
```

```
                cla:local_name = "cornerslat" ;
        double clz(corners_k, k) ;
                clz:local_name = "cornersz" ;
        int imask(k, j, i) ;
                imask:units = "unitsless" ;
                imask:local_name = "mask" ;
                imask:unvalid_point = "0" ;
                imask:valid_point = "1" ;
}
```

Il n'y a besoin de définir que la longitude et la latitude de 2 coins pour chaque cellule dans le plan (i,j) puisque la grille est régulière, donc corners_ij = corners_k = 2.

Pour une grille régulière, prism_gridtype vaut "PRISM_REGLONLATVRT".

3.1.3 Entête du fichier NetCDF d'une grille IRRLONLAT_REGVRT

Pour une grille irrégulière, la longitude et la latitude des points dépend de i et de j, tandis que la verticale ne dépend que de k. Nous avons choisi la convention suivante pour l'écriture des données d'une telle grille (ici la grille nommée OPA8_T) :

```
netcdf grid_opa8_T {
dimensions:
        i = 182 ;
        j = 149 ;
        k = 1 ;
        corners_ij = 4 ;
        corners_k = 2 ;
        character = 22 ;
variables:
        char prism_gridtype(character) ;
        double lon(j, i) ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
                lon:periodic = "true" ;
                lon:overlap = "2" ;
        double lat(j, i) ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
```

```
        double z(k) ;
                z:units = "meters" ;
                z:standard_name = "vertical" ;
        double clo(corners_ij, j, i) ;
                clo:local_name = "cornerslon" ;
        double cla(corners_ij, j, i) ;
                cla:local_name = "cornerslat" ;
        double clz(corners_k, k) ;
                clz:local_name = "cornersz" ;
        int imask(k, j, i) ;
                imask:units = "unitsless" ;
                imask:local_name = "mask" ;
                imask:unvalid_point = "0" ;
                imask:valid_point = "1" ;
}
```

Contrairement aux grilles REGLONLATVRT et GAUSSREDUCED_REGVRT, il est nécessaire de définir les longitudes et latitudes des coins (corners_ij) de chaque cellule dans le plan (lon, lat) où elle est irrégulière, tandis qu'il n'en faut toujours que deux selon la verticale z où elle est régulière (corners_k=2).

Pour une grille irrégulière, prism_gridtype vaut "PRISM_IRRLONLATVRT_REGVRT".

3.1.4 Entête du fichier NetCDF d'une grille GAUSSREDUCED_REGVRT

Pour une grille de type GAUSSREDUCED_REGVRT, les points sont repérés par un seul indice dans le plan (lon,lat), npt_hor, qui décrit le plan par bandes de latitudes. La grille est régulière selon z. La longitude et la latitude des points ne dépendent que de l'indice npt_hor (lon(npt_hor), lat(npt_hor)), tandis que la verticale des points reste une fonction de k uniquement (z(k)).Nous avons choisi la convention suivante pour l'écriture des données d'une telle grille (ici la grille nommée GAUSSRED_64=BT42) :

```
netcdf grid_gaussred_64_mask {
dimensions:
        npt_hor = 6232 ;
        k = 1 ;
        corners_ij = 2 ;
        corners_k = 2 ;
        character = 25 ;
variables:
```

```
        char prism_gridtype(character) ;
        double lon(npt_hor) ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
                lon:periodic = "true" ;
                lon:overlap = "0" ;
        double lat(npt_hor) ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
        double z(k) ;
                z:units = "meters" ;
                z:standard_name = "vertical" ;
        double clo(corners_ij, npt_hor) ;
                clo:local_name = "cornerslon" ;
        double cla(corners_ij, npt_hor) ;
                cla:local_name = "cornerslat" ;
        double clz(corners_k, k) ;
                clz:local_name = "cornersz" ;
        int imask(k, npt_hor) ;
                imask:units = "unitsless" ;
                imask:local_name = "mask" ;
                imask:unvalid_point = "0" ;
                imask:valid_point = "1" ;
}
```

Comme pour une grille REGLONLATVRT, il n'y a besoin de définir que la longitude et la latitude de 2 coins pour chaque cellule dans le plan (i,j), donc corners_ij = corners_k = 2.

Pour une grille gaussienne réduite, prism_gridtype vaut "PRISM_GAUSSREDUCED_REGVR

3.1.5 Lecture des données

Deux cas sont possibles : soit on lit les données à partir du nom des dimensions et des variables, et il faut alors utiliser la clé de précompilation LEC_PAR_NOM_VAR, soit on lit les données d'une manière la moins dépendante possible du nom des dimensions et des variables, en utilisant directement les attributs. C'est le cas par défaut, il ne faut aucune clé de précompilation pour l'activer, car c'est le cas le moins "utilisateur dépendant".

L'utilisation de la première option nécessite d'avoir les mêmes noms de variables que ceux donnés dans les fichiers exemples. Il faut également que le

nombre de coins soient les dernières dimensions de la longitude, de la latitude et de la verticale des coins.

L'utilisation de l'option par défaut est plus souple, car il ne faut connaitre qu'un attribut pour lire la variable correspondante, même s'il faut également que le nombre de coins soient les dernières dimensions de la longitude, de la latitude et de la verticale des coins.
Concernant les attributs, il faut que les attributs définis au paragraphe 3.1.1, qui permettent de reconnaitre les différentes variables lues, soient présents.

3.2 Clés de pré-compilation

C'est dans le fichier Makefile, qui gère la compilation des programmes et la constitution des fichiers XML de configuration, que l'on spécifie toutes les clés de pré-compilation.

3.2.1 Clés de précompilation dans les fichiers de configuration

Dans les fichiers de configuration de la source et de la cible, source_source_smioc.xml.in et target_target_smioc.xml.in, les clés de pré-compilation permettent de sélectionner les variables correspondant à chaque type de grille comme suit :

```
#ifdef REGLONLATVRT
 grid_type="PRISM_reglonlatvrt">
#elif defined IRRLONLAT_REGVRT
 grid_type="PRISM_irrlonlat_regvrt">
#elif defined GAUSSREDUCED_REGVRT
 grid_type="PRISM_gaussreduced_regvrt">
#endif
```

Dans le fichier de configuration de la cible, target_smioc.xml.in, les clés de pré-compilation permettent également de sélectionner l'interpolation, puisque c'est elle qui reçoit le champ :

```
#if defined (BILINEAR)

#elif defined (TRILINEAR)

#elif defined (BICUBIC)

#elif defined (CONSERVATIVE)
```

```
#elif defined (NNEIGHBOUR)

#elif defined (NNEIGHBOUR2D)

#endif
```

3.2.2 Clés de précompilation dans les fichiers Fortran

Les clés de précompilation REGLONLATVRT, IRRLONLAT_REGVRT ou GAUSSREDUCED_REGVRT permettent de définir le type de grille dans les fichiers source.F90 et target.F90. Cela permet, entre autre ensuite, de lire correctement les données relatives au type de grille sélectionné. On peut souligner qu'une écriture des données au format préconisé dans la partie (3.1) permet de reconnaitre très facilement le type de la grille source ou de la grille cible rien qu'à partir de la dépendance des variables en fonction des indices i, j, k.

Les calculs peuvent ensuite s'effectuer soit en simple, soit en double précision. Nous avons déjà signalé que les données relatives aux grilles devaient être forcément en double précision. Par contre, on peut choisir de travailler en simple ou en double précision au moment de définir les variables et les champs de couplage dans les codes source.F90 et target.F90. La clé de précompilation permettant de faire les calculs en double précision est USE_DOUBLE_PRECISION. Pour réaliser nos tests sur la qualité des interpolations dans OASIS4, nous ne travaillons qu'en double précision.

Lorsque la clé USE_MASK est activée, le masque lu dans le fichier de données est associé à la grille correspondante. Sinon par défaut le masque vaut .TRUE. partout et tous les points de la grille sont alors valides.

Les trois fonctions analytiques tests présentées dans la partie (2), ainsi qu'une fonction constante, ont été codées dans les deux fichiers source.F90 et target.F90. Ce sont les clés CONSTANTE, FUNCTION_1, FUNCTION_2 ou FUNCTION_3 qui permettent de sélectionner le champ de couplage envoyé de la source à la cible. Cette clé doit être activée aussi bien pour la source que pour la cible et doit être la même car c'est à partir de cette fonction que la cible calcule l'erreur d'interpolation.

Les programmes source.F90 et target.F90 peuvent tourner soit en monoprocesseur soit en parallèle. Par défaut, la parallélisation est réalisée selon la

direction j car dans ce cas elle est indépendante du type de grille considéré et ne dépend que du nombre de processeurs demandés pour chaque composante des modèles. C'est dans le Makefile que l'on fixe le nombre de processeurs pour la source et pour la cible.

4 Erreur permettant de quantifier la qualité de l'interpolation dans OASIS4

Le code source source.F90 calcule le champ de couplage sous la forme d'une fonction analytique f1, f2 ou f3 (voir la partie 2) puis l'envoie à la cible directement ou via une interpolation, selon les cas.

L'erreur, entre la valeur du champs analytique envoyé et la valeur du champ reçue par la cible, est calculée dans le code cible target.F90 sur la grille cible, par la formule :

$$E(i,j,k) = \frac{f_{recue}^{cible}(i,j,k) - f_{ana}^{cible}(i,j,k)}{f_{ana}^{cible}(i,j,k)} \tag{5}$$

A titre d'illustration, nous présentons l'erreur obtenue lors de l'échange de la fonction f1 entre deux grilles régulières AT42, lors de l'échange de la fonction f2 d'une grille régulière AT42 vers la grille irrégulière ORCA2 avec une interpolation bilinéaire (pour laquelle il faut 4 points sources pour pouvoir faire l'interpolation) et lors de l'échange de la fonction f3 d'une grille régulière AT42 vers la grille gaussienne réduite BT42 avec une interpolation bicubique (pour laquelle il faut 16 points sources pour pouvoir faire l'interpolation). Toutes les grilles ont un masque.

La figure (2) représente, en haut à gauche, le champ source analytique f1 sur la grille régulière AT42. Le champ est reçu directement, sans interpolation, sur la grille cible régulière AT42 (en haut à droite). On observe que l'erreur (en bas à droite) vaut zéro sur tout le domaine de calcul.

La figure (3) représente, en haut à gauche, le champ source analytique f2 sur la grille régulière AT42. Le champ est reçu, après une interpolation bilinéaire, sur la grille cible irrégulière ORCA2 T (en haut à droite). Les valeurs minimale et maximale de l'erreur, $-3.7\ 10^{-2}$ et $5.8\ 10^{-2}$, sont localisées près des côtes (erreur en bas à gauche). Partout ailleurs l'erreur est inférieure à 0.3% (erreur en bas à droite).

La figure (4) représente, en haut à gauche, le champ source analytique f3 sur la grille régulière AT42. Le champ est reçu, après une interpolation bicubique, sur la grille cible gaussienne réduite BT42 (en haut à droite). Les valeurs minimale et maximale de l'erreur valent -0.13 et 0.2 mais elles sont

très localisées, près des côtes et là où la fonction varie très fortement spatialement (erreur en bas à gauche). Ailleurs, l'erreur est inférieure 0.2% (erreur en bas à droite).

Le masque de la grille source régulière atmosphérique AT42 permet de reproduire la disposition spatiale des océans et des continents de la grille océanique cible vers laquelle elle envoie le champ de couplage. Il y a donc moins de points sources que le nombre nécessaire pour l'interpolation près des côtes puisque ces points sont masqués côté source. Différentes options existent pour palier à ce type de problème, mais c'est ce qui permet d'expliquer que la qualité de l'interpolation soit moins bonne dans ces zones.

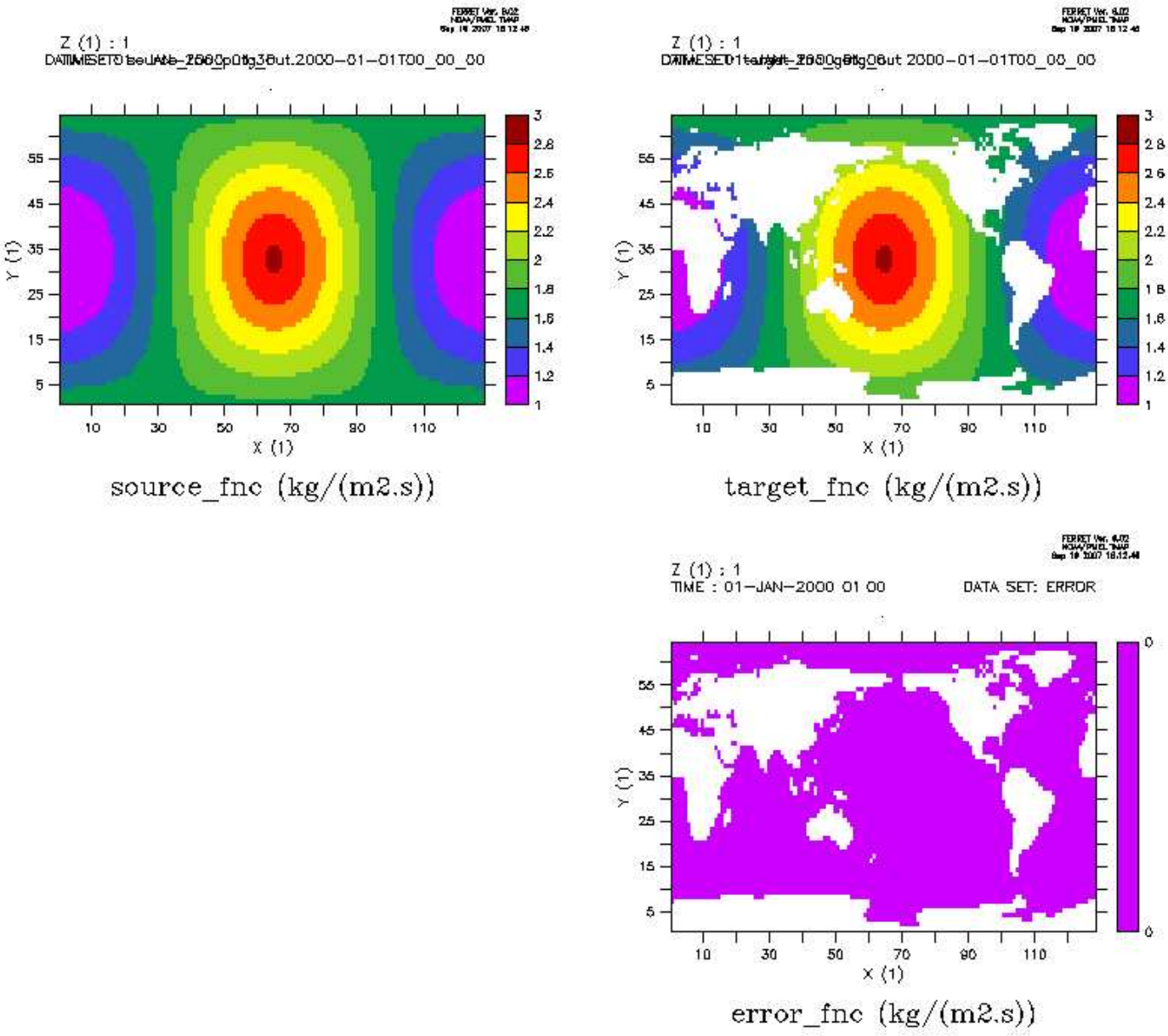

FIG. 2 – Echange de la fonction analytique f1 (1) de la grille AT42 (en haut à gauche) vers la grille AT42 (en haut à droite) avec l'erreur entre les champs (en bas à droite)

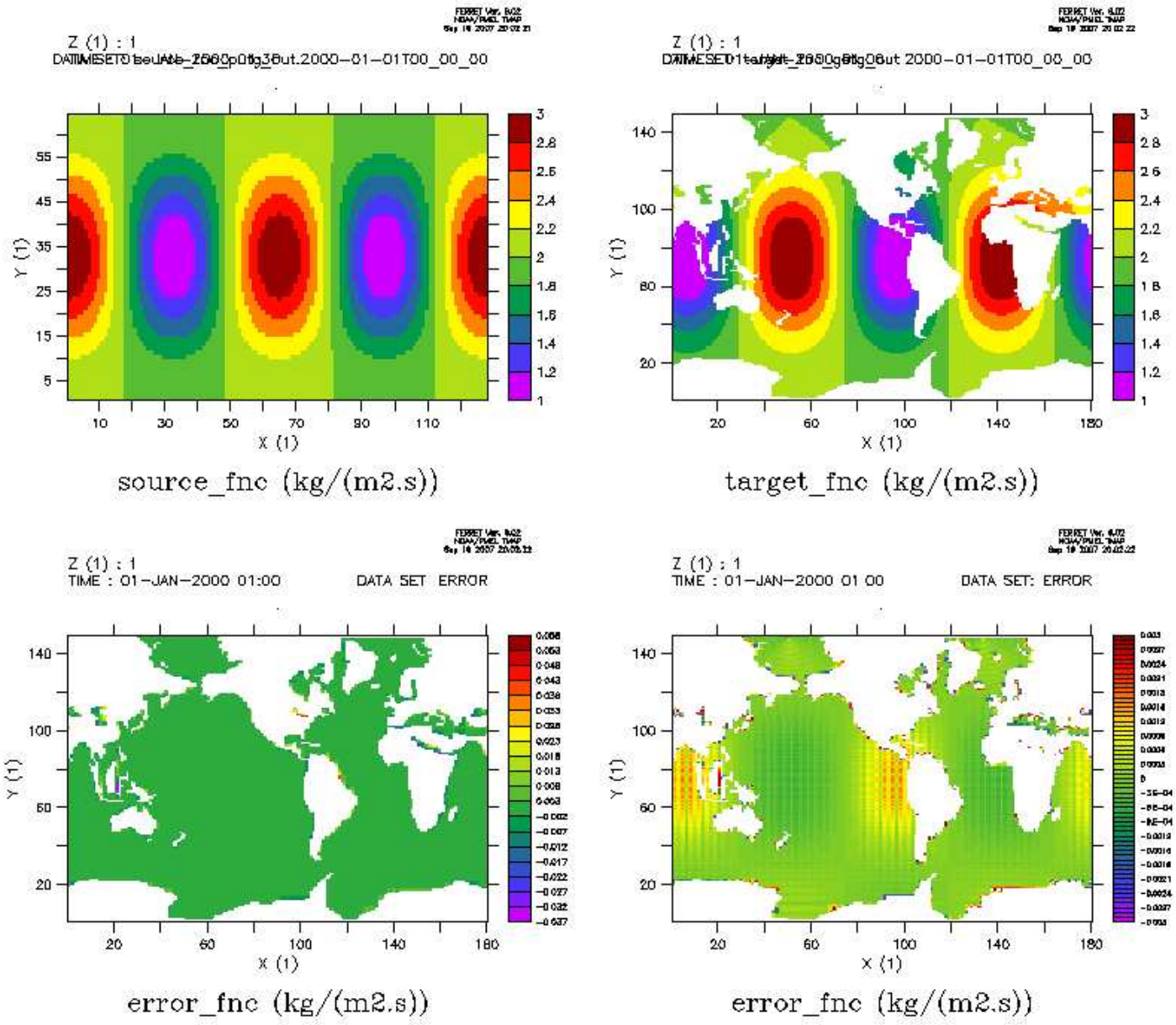

FIG. 3 – Interpolation bilinéaire de la fonction analytique f2 (2) de la grille AT42 (en haut à gauche) vers la grille ORCA2 (en haut à droite) avec l'erreur d'interpolation (en bas à gauche et à droite)

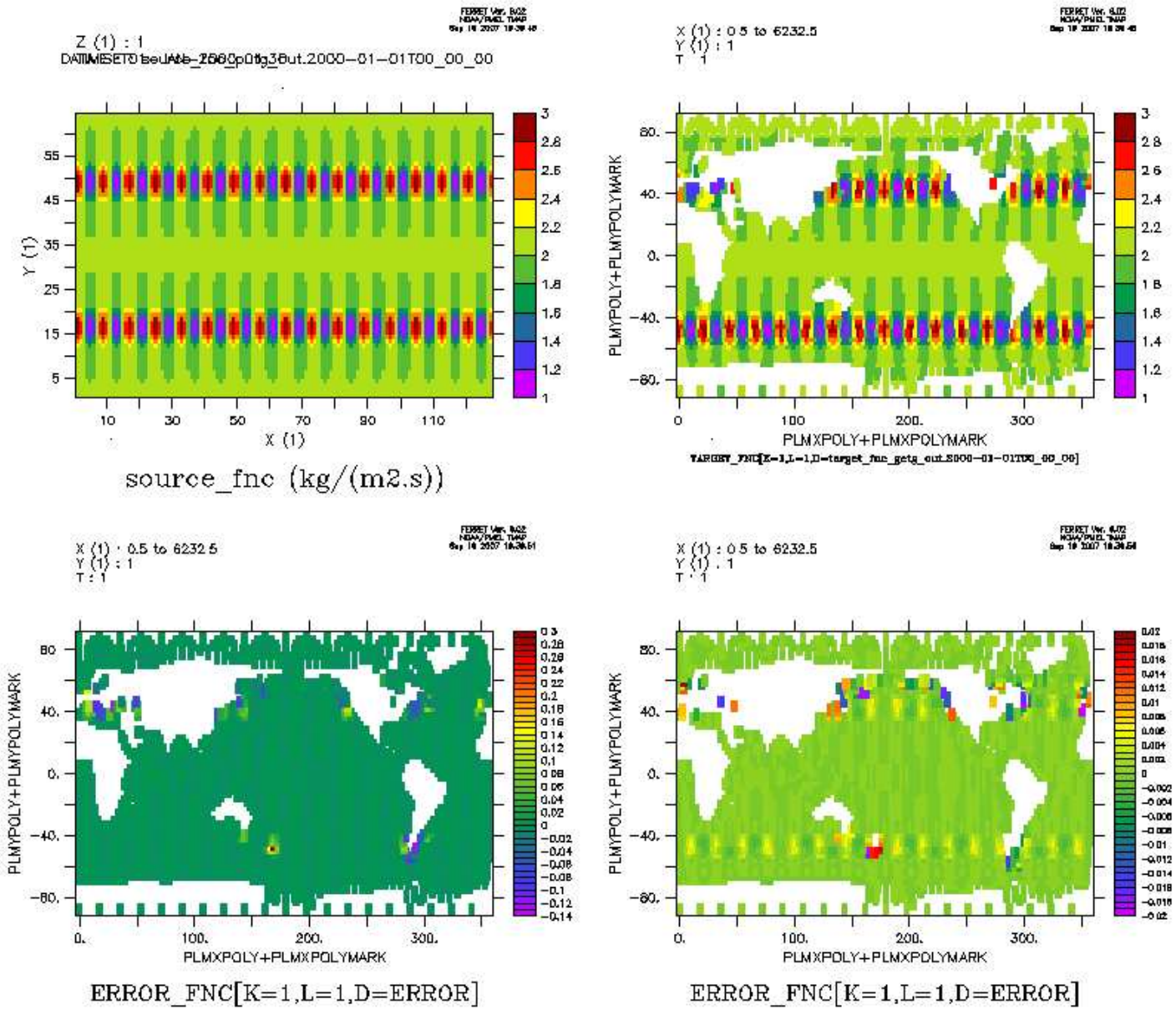

FIG. 4 – Interpolation bicubique de la fonction analytique f3 (3) de la grille AT42 (en haut à gauche) vers la grille ORCA2 (en haut à droite) avec l'erreur d'interpolation (en bas à gauche et à droite)

5 Conclusions, Perspectives

La première partie du document présente, de façon formelle, l'environnement qui nous permet de tester la qualité des interpolations dans la dernière version du coupleur OASIS4, c'est-à-dire les différents types de grilles supportés par OASIS4, ainsi que les différents fichiers constituant le banc d'essai.

La seconde partie s'attache à décrire en détail le format, choisi le plus générique possible, qui est utilisé pour stocker les données concernant les différents types de grilles et leurs masques. Le rôle des différentes clés de précompilation, permettant de fixer la configuration à tester, est ensuite expliqué.

L'objectif, avec cet outil, est de pouvoir tester n'importe quelle interpolation entre deux grilles (masquées), en utilisant des champs suffisamment réalistes, et en quantifiant à chaque fois la qualité de l'interpolation réalisée. Trois fonctions analytiques trigonométriques tests, f1, f2 et f3, présentant des variations spatiales très différentes, ont été retenues pour représenter les champs échangés.

De manière à illustrer le fonctionnement du banc d'essai, et donc le calcul de l'erreur d'interpolation en pratique, nous avons présenté les résultats obtenus lors de l'échange de la fonction analytique f1 entre deux grille régulières AT42, pour l'interpolation bilinéaire de la fonction f2 de la grille régulière AT42 vers la grille irrégulière ORCA2 et pour l'interpolation bicubique de la fonction f3 de la grille régulière AT42 vers la grille gaussienne réduite BT42.

L'outil mis en place pour tester les interpolations dans OASIS4 est utilisé actuellement dans le cadre du projet CICLE (Calcul Intensif pour le Climat et l'Environnement) financé par l'ANR (projet numéro ANR-05-CIGC-04), pour tester les interpolations entre différentes grilles de modèles globaux ou régionaux (OPA9, ARPEGE REDUITE, ALADIN, OPA-MED ou LMDZ). Les résultats obtenus avec les différentes interpolations sont visibles sur la page WEB :
http ://www.cerfacs.fr/ coquart/pagecerfacs/projet_cicle/
RESULTS_BASSE_RESOLUTION/projet_cicle.html.

Cet environnement est également déjà utilisé pour tester facilement des cas soumis par les utilisateurs d'OASIS4, sans avoir à refaire à chaque fois un environnement adapté.

Références

[1] B. Eaton (NCAR), J. gregory (UK Met Office), B. Drach (LLNL), K. Taylor (LLNL), and S. Hankin (NOAA). Netcdf climate and forecast (cf) metadata conventions, version 1.0. Technical report, NCAR, UK Met Office, LLNL, NOAA, 2003.

[2] S. Valcke and R. Redler. Oasis4_0_2 user's guide and reference manual. Technical report, CERFACS, Toulouse, France and NEC-CCRLE, Sankt-Augustin, Allemagne, 2006.