Génie Mathématique et Modélisation 3

RAPPORT DE STAGE 2009-2010

DÉVELOPPEMENT DE

LA LIBRAIRIE D'INTERPOLATION

DU COUPLEUR OASIS

Ersin KILICOGLU

Sous la direction de Sophie VALCKE

29 mars - 27 août 2010

Remerciements

Je voudrais tout d'abord remercier ma tutrice de stage, Sophie VALCKE[1], pour la confiance qu'elle m'a accordé en me chargeant du développement de la librairie d'interpolation du logiciel de couplage de codes OASIS. Je la remercie pour l'aide et les conseils concernant les missions évoquées dans ce rapport, qu'elle m'a apporté lors des différents suivis. Mais aussi d'avoir été une tutrice de stage très disponible avec qui j'ai eu l'opportunité d'avoir des échanges passionnants de nature notamment mathématico-philosophique.
Les moyens qu'elle a mis à ma disposition tels que la formation Fortran 90 à Paris, mais aussi le profit issu de son expérience, et sa confiance sont les éléments essentiels qui m'ont permis de mener à bien cette mission.
Je dois aussi beaucoup à Madame VALCKE d'un point de vue personnel pour son accueil très chaleureux au sein de l'équipe GlobC et son encouragement qu'elle m'a témoigné tout au long de mon stage.

De la même façon, j'exprime ici ma gratitude envers Laure COQUART, ma co-tutrice de stage, pour sa disponibilité et ses conseils concernant notamment l'utilisation de scripts, mais aussi pour m'avoir fournit les fichiers informatiques nécessaires au bon déroulement de ma mission.

Mes sincères remerciements vont ensuite à Olivier BODART[2], mon responsable de stage à l'école, pour son soutien et sa visite qu'il a effectué lors de mon stage. Avec François BOUCHON, il est membre du jury chargé d'évaluer la qualité de mon travail. Je remercie aussi mon jury.

Aussi, mes remerciements vont à Élodie et Philippe, ingénieure et stagiaire, avec qui j'ai eu le plaisir de partager mon bureau. Les sujets de conversation plein d'intérêt que j'ai eu avec eux, Marie et Yohan tout au long des pauses prises après de dures journées de travail demeureront sans doute en ma mémoire. A ces personnes talentueuses, je souhaite un accomplissement dans tous les domaines de la vie.

Je tiens à remercier l'ensemble de l'équipe Global Change du CERFACS : Marie-Pierre, Christian, Philippe, Bertrand, Gabriel, Sophie, Christophe, Émilia, Thierry, Éric, Anthony et Jean-Philippe, pour leur bonne humeur, leur bonne compagnie, mais aussi pour leur chaleureux accueil lors de mon arrivée au CERFACS.

[1] valcke@cerfacs.fr, Ingénieure de recherche au CERFACS, Responsable du développement du logiciel OASIS
[2] olivier.bodart@math.univ-bpclermont.fr, Maître de conférences à l'Université Blaise Pascal, Enseigant en mathématiques à Polytech'Clermont-Ferrand

Résumé

J'ai effectué mon stage de fin d'étude au sein du département «Global Change» du CERFACS. Ce groupe effectue des travaux afin d'évaluer l'intensité du changement climatique et d'étudier ses conséquences sur la société. Intégré au sein de l'équipe travaillant au développement du coupleur OASIS, j'ai travaillé avec ma responsable de stage au développement d'un algorithme d'interpolation conservative, car l'algorithme existant retournait des valeurs erronées pour les mailles proches des pôles. Ces erreurs sont dûes au fait qu'à mesure qu'on s'approche des pôles les hypothèses sur lesquelles l'algorithme initial repose sont de moins en moins valides. En effet, celui-ci considère que les bords des mailles des grilles sources et cibles sont linéaires en longitude et latitude. Nous avons constaté que l'option existante consistant à projeter les mailles proches des pôles cause même dans certains cas plus de problèmes : les calculs d'intersections se font dans le repère projeté mais on revient dans le repère sphérique initial pour les calculs d'intégrales de ligne. Cette façon de faire génère des valeurs interpolées inexactes. Après avoir déterminé les causes des erreurs dans les cas avec et sans projection, nous avons écrit et codé un algorithme permettant de s'en affranchir. Nous avons introduit la rotation des mailles se trouvant au-delà d'une certaine latitude pour les ramener près de l'équateur puisque c'est dans cette région que les hypothèses sont les plus valides. Puis nous avons procédé aux calculs pour ces mailles tournées. Enfin, nous avons raccordé les mailles tournées et non-tournées. La méthode a été validée par des tests théoriques ainsi que sur des grilles issus de modèles réels. Aussi, les résultats obtenus montrent que les erreurs observées avec l'ancien algorithme disparaissent.
Mots clés : Coupleur, Modélisation du climat, Simulation numérique, Fortran90.

Abstract

I achieved my third year training at Global Change team at CERFACS, in Toulouse. This team works on European projects about the global change to judge its consequences on the society. During my training, I worked with the group which develops the new version of OASIS coupler. In this software, there is a conservative remapping method which permits the communication of information between different grids of different models. This algorithm is based on hypotheses which become less valid when the grid cells are near the poles. In fact, it considers that the borders are linear in longitude latitude between the corners of a cell. We verified that the using of an existing option, which allows the user to do a projection of cells which are above a given latitude causes others problems due to the fact that some calculations are done after projection and others in the initial frame. We wrote an algorithm which includes the rotation of cells which are above a given latitude. After this rotation, the cells which were near the poles come near the equator where basic hypotheses are more valid. So we implemented this algorithm and completed it by joining the rotated and not rotated cells. The method was validated on theoretical test cases and with some real model grids. We showed that our method gives more exact results for all tests cases.
Keywords : Model coupler, Climate modelling, Numerical simulation, Fortran90.

Table des figures

Table des matières

INTRODUCTION

Dans les laboratoires de recherches, les scientifiques sont au service de la société. J'ai constaté ce fait lorsque j'ai pratiqué mon stage de fin d'études au CERFACS à Toulouse. Ce laboratoire est une société civile où se trouvent des équipes travaillant à la résolution, par la modélisation et la simulation numérique, des problèmes scientifiques nécessitant le recours aux moyens de calcul les plus puissants. L'une de ces équipes est chargée du développement de la version 4 du coupleur OASIS. Celui-ci est actuellement utilisé pour coupler des modèles climatiques dans une trentaine de pays.

Une des méthodes utilisées dans le logiciel permet d'interpoler les valeurs provenant des points de mailles d'une grille source sur les points de mailles d'une grille cible. Ceci est nécessaire puisque les grilles utilisées par les différents modèles sont différentes et que pour faire communiquer les modèles il est nécessaire à un modèle cible de disposer des informations provenant d'un modèle source. Le sujet de mon stage consiste à écrire et coder en langage Fortran une méthode d'interpolation conservative 2D permettant d'interpoler justement. En effet, l'algorithme utilisé jusqu'à présent retournait des valeurs d'interpolation erronées pour les mailles proches des pôles.

Après avoir présenté, d'une part, l'environnement de mon stage et d'autre part, le coupleur OASIS, nous identifierons les défauts de l'algorithme actuel. Ensuite, dans une troisième partie, nous expliquerons les anomalies constatées sur les mailles proches du pôle nord et proposerons d'introduire la rotation dans l'algorithme pour y remédier.

1 LE CERFACS ET OASIS, UNE HISTOIRE DE 20 ANS

Le CERFACS a pour domaine d'activité la résolution par modélisations et simulations numériques des problèmes nécessitant le recours au calcul à hautes performances.

1.1 Le CERFACS, une société dynamique !

Le **CERFACS** (**C**entre **E**uropéen de **R**echerche et **F**ormation **A**vancées en **C**alcul **S**cientifique) est une société civile financée pour environ la moitié de son budjet par 7 actionnaires : le CNES, EDF, Météo-France, EADS, ONERA, SAFRAN et Total dont les logos apparaissent sur la figure 1.1.
L'autre moitié de son budjet provient de contrats privés ou publics.

FIG. 1.1 – Les logos du CERFACS et de ses sept actionnaires.

Le CERFACS s'emploie à la résolution, par la modélisation et la simulation numérique, des problèmes scientifiques nécessitant le recours aux moyens de calcul les plus puissants. Il associe de manière interdisciplinaire, pour la recherche comme pour la formation avancée, des physiciens, des mathématiciens appliqués, des numériciens et des ingénieurs.
Le CERFACS est gouverné par un Conseil de Gérance des représentants de ses sept actionnaires et bénéficie des recommandations de son Conseil Scientifique.

Une partie des activités de recherches du CERFACS se fait en association avec le CNRS (Centre National de la Recherche Scientifique, www.cnrs.fr), sous la forme d'Unité de Recherche Associée (SUC, URA 1875). Le CERFACS et l'INRIA (Institut National de Recherche en Informatique et Automatique, www.inria.fr) ont aussi combiné une partie de leurs activités dans un laboratoire commun. Le CERFACS participe au pôle TVE (Terre Vivante et Espace, http ://tve.omp.obs-mip.fr/). Il est aussi membre du RTRA/STAE (Réseau Thématique de Recherche Avancée «Sciences et Technologies pour l'Aéronautique et l'Espace», www.fondation-stae.net/fr/, voir aussi www.cerfacs.fr/RTRA-STAE.mpg) et participe aux activités du pôle de compétitivité AESE («Aéronautique, Espace et Systèmes Embarqués», www.aerospace-valley.com).

Environ 115 personnes travaillent au CERFACS, incluant plus de 95 chercheurs et ingénieurs, venant d'une dizaine de pays différents. Ils travaillent sur des projets spécifiques dans 9 principaux domaines de recherche tels que l'algorithmique parallèle, le couplage de codes, l'aérodynamisme, les turbines de gaz, la combustion, le climat, l'étude de l'impact environnemental, l'assimilation de données et l'électromagnétisme.

L'équipe "Modélisation du Climat et de son Changement Global" du CERFACS a été créée en 1990 et devient, en 1998, une branche du CNRS, en prenant le statut d'Unité de Recherche Associée (URA 1875). Cette équipe conduit des recherches appliquées dans le domaine de l'étude du climat. Son principal objectif est d'apporter une contribution significative dans la compréhension et la prévision de l'environnement et du climat, à l'échelle régionale et mondiale. Sa stratégie repose sur une approche théorique et sur une étude des modèles, ainsi que sur le développement de logiciels de haut niveau devenant indispensables dans la science du climat. Cette politique est dirigée à travers la coordination et la participation de l'équipe aux programmes de recherches nationaux et internationaux.

Parmi les logiciels développés par l'équipe "Modélisation du Climat et de son Changement Global", on trouve PALM et OASIS. Ce sont tous les deux des logiciels de couplages, couramment appelés "coupleurs". Ils sont développés par deux équipes distincts.
Le projet PALM[1] est né en 1996 avec le projet d'océanographie opérationnelle MERCATOR. A son origine, on trouve la suggestion du responsable du projet MERCATOR de l'époque, Philippe Courtier, qui proposait qu'"une chaîne d'assimilation de données pouvait être conçue et implémentée comme un couplage". C'est alors que l'équipe PALM au CERFACS fut choisie pour le développement de la structure logicielle du projet. Il s'agissait d'un défi technologique : développer un coupleur parallèle dynamique et son interface utilisateur, destiné à être utilisé sur différentes plates-formes matérielles.
Le projet OASIS dépend de l'équipe à laquelle j'ai été intégré pour la durée de mon stage. Intéressons-nous plus en détails au coupleur OASIS.

[1]Documentation sur PALM est disponible depuis le lien : http ://www.cerfacs.fr/globc/PALM_WEB/

1.2 OASIS, un coupleur pour le climat

Le couplage de codes numériques est l'échange synchronisé d'informations entre différents codes. Un coupleur est un outil logiciel qui va permettre au développeur d'applications d'exécuter et de faire échanger ensemble un certain nombre de programmes qui n'ont pas forcément été prévus pour cela. En plus des aspects d'échange de données, le coupleur peut assurer un certain nombre de services comme des calculs intermédiaires sur les données, de la redistribution de structures de données ou de l'interpolation entre maillages différents. Le couplage peut aller du simple assemblage de composants par chaînage à des applications faisant intervenir des dizaines de modèles. Les codes de calcul doivent parfois s'exécuter en parallèle si le couplage intervient dans des processus itératifs internes aux entités de calcul.

Le couplage de codes a acquis ces dernières années une importance de tout premier plan dans plusieurs domaines scientifiques tels que la modélisation du climat, l'assimilation de données, et la mécanique des fluides et des structures. Dans ce cadre, l'équipe "Modélisation du Climat et de son Changement Global" du CERFACS, dans laquelle j'ai réalisé mon stage de fin d'études, développe le logiciel OASIS. Il est conçu et utilisé pour coupler des modèles représentant les différentes composantes du système climatique : modèles de circulation générale atmosphérique, de circulation générale océanique, de glace de mer, de sol, d'hydrologie, etc. De la même façon que PALM, OASIS[2] (Ocean, Atmospher, Sea Ice, Soil) est un logiciel utilisé par des laboratoires à travers le monde entier. Il est actuellement développé par le CERFACS (Toulouse, France), DKRZ (Hambourg, Allemagne) et le CNRS(Paris, France) et est intégré aux programmes IS-ENES et EU FP7 METAFOR.

Le projet IS-ENES a été instauré comme la continuité du projet PRISM EU Project (Program for Integrated Earth System Modelling) était un programme européen prévu pour la période 2001-2004. Pour un budget de 4 800 000 €, il réunissait 22 partenaires du secteur public et privé. En 2005, prolongé pour 3 ans, il prend le nom de PRISM Support Initiative (PSI) avec 7 partenaires dont le CERFACS, le CNRS ou le UK MetOffice et 9 partenaires associés.

IS-ENES (**I**nfra**S**tructure for the **E**uropean **N**etwork for the **E**arth **S**ystem Modelling) est un projet FP7 subventionné par la Commission Européenne. Le projet a été lancé le 1 mars 2009 et prendra fin le 28 février 2013.
Les modèles climatiques du système Terre sont les outils clefs pour la compréhention du changement climatique et de ses effets sur la société. Ces modèles sont à la base de ce projet.
IS-ENES rassemble 18 partenaires de pays européens et inclue les 6 principaux modèles climatiques globaux développés par les européens. IS-ENES combine l'expertise en modèlisation climatique du système Terre, en science de la programmation et en études des impacts du changement climatique.

[2]Documentation sur OASIS disponible depuis le lien : http ://pantar.cerfacs.fr/3-26318-OASIS-coupler-and-applications.php

1.3 L'architecture d'OASIS3

Le coupleur OASIS3 est à présent utilisé par une trentaine de groupes de climatologues travaillant sur des modèles climatiques en Europe, aux États-Unis d'Amérique, au Canada, en Australie, en Inde et au Brésil.

La version OASIS3 est l'évolution du coupleur initial OASIS développé depuis 1991 par le CERFACS. La portabilité et la flexibilité sont les lignes directrices du développement d'OASIS3. En mode coupleur, ce dernier agit comme un exécutable, dont la fonction principale est d'interpoler des champs donnés sur des grilles provenant d'un modèle source sur des grilles utilisées dans les modèles cibles, et comme une librairie devant être «linkée» aux modèles à coupler. Cette utilisation se fait seulement sur des domaines 2D.
On peut aussi utiliser OASIS seul, sans modèles, pour interpoler des champs présents dans les fichiers ; on dit alors qu'on utilise OASIS en mode interpolateur.
OASIS permet donc principalement de :

- échanger, de façon synchronisée, des champs de couplage entre des modèles provenant d'exécutables différents,
- interpoler, en 2 dimensions, des valeurs de ces champs de couplage sur des mailles d'une grille cible, à partir de valeurs provenant d'une grille source.

La modularité et la flexibilité, permettant de faire interagir entre eux un nombre quelconque de modèles échangeant un nombre quelconque de champs sur lesquels sont faites des interpolations quelconques, sont les avantages les plus prononcés de ce coupleur.

Le code d'OASIS3 relève du domaine public et est accessible depuis le site internet du CERFACS[3]. Il s'agit d'un code écrit principalement en langage Fortran90. Il utilise des librairies externes émanant aussi du domaine public telles que MPI1 et/ou MPI2, NetCDF ou parallel NetCDF, libXML, ainsi que des librairies inclues dans le code telles que GFDL mpp_io (librairie d'entrées/sorties) et LANL SCRIP (librairie d'interpolation).

OASIS3 possède, d'une part, ses propres codes sources à partir desquels sont gérés les deux modes de fonctionnement du coupleur. Et, d'autre part, il dispose de 8 librairies internes auxquelles il peut faire appel lors de l'exécution des fichiers sources propres.

Au premier niveau d'arborescence, il se présente 5 répertoires contenant les fichiers sources propres à OASIS3, les 8 librairies internes, la documentation, les cas tests et les éléments nécessaires à la compilation et éventuellement au debbugage. Parmi les librairies internes, celle sur laquelle nous avons travaillé durant mon stage est la librairie d'interpolation SCRIP et plus particulièrement sur le fichier d'interpolation conservative, remap_conserv.F90 qu'elle contient.
L'arborescence complète d'OASIS3 est représentée sur la figure 1.2, page 6.

[3]Les indications nécessaires pour l'obtention des codes sources d'OASIS3 sont indiquées dans le lien https ://oasis-trac.cerfacs.fr/

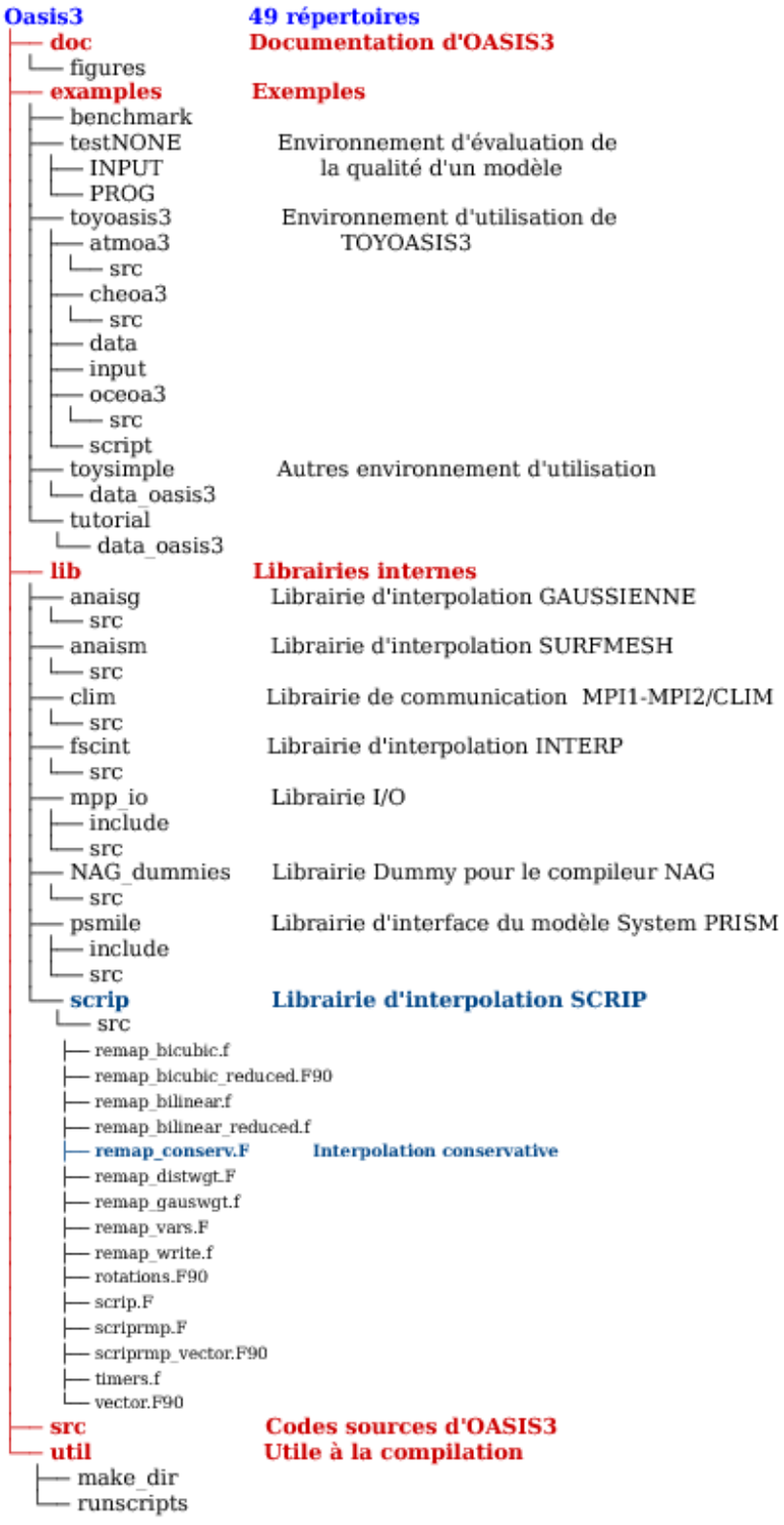


FIG. 1.2 – L'arborescence d'OASIS3.

2 L'INTERPOLATION CONSERVATIVE DE LA LIBRAIRIE SCRIP

L'interpolation conservative est codée dans le fichier *remap_conserv.F90* contenu dans le répertoire SCRIP, qui est aussi le nom de la librairie d'interpolation d'OASIS3 (le répertoire contient les fichiers sources de cette librairie). Nous nous placerons dans le sous-répertoire testNONE de examples pour tester le code de l'interpolation conservative.

2.1 L'environnement des tests

Dans le but de tester les procédures d'OASIS3, il a été prévu un certain nombre de cas tests où il devient facile de repérer les erreurs de codage ou de valider une procédure. En mode interpolateur (par opposition au mode coupleur), les tests pour l'utilisation de la librairie SCRIP, ont été implémentés dans le répertoire oasis3/examples/testNONE. Celui-ci contient deux sous-répertoires nommés INPUT et PROG, ainsi qu'un fichier script nommé *sc_run_NONE*.
Le répertoire INPUT contient notamment les 3 fichiers NetCDF *grids.nc, masks.nc, areas.nc* et un fichier de configuration nommé *namcouple*. PROG, quant à lui, il contient 2 fichiers Fortran90 *create_inputfield.f90* et *calc_errorfield.f90*.

2.1.1 Namcouple : le fichier de configuration

Le fichier de configuration *namcouple*[1] contient, sous forme de mots-clés prédéfinis, les informations nécessaires pour la configuration d'OASIS3 suivant les besoins de l'utilisateur. Si on décide, par exemple, d'exécuter OASIS3 en mode parallèle sur une machine multi-processeur, l'utilisateur doit établir un fichier namcouple par processus OASIS. Pour plus de détails sur le mode parallèle d'OASIS3, on pourra se reporter au chapitre 8 du manuel d'utilisation.

La *namcouple* est un fichier texte possédant les caractéristiques suivantes :

- l'ordre des mots clés n'a pas d'influence,
- le nombre d'espaces blancs entre deux chaînes de caractères est non-signifiant,
- toutes les lignes commençant avec le symbole # sont ignorées,
- les lignes blanches ne sont pas autorisées.

[1] Un exemple très détaillé de fichier *namcouple* est présent en annexe 1 de ce rapport

La première partie de la namcouple définit les paramètres généraux tels que le nombre de modèles utilisés lors de la simulation, le nombre de champs de couplage, la technique de communication, etc. La seconde partie recueille l'information spécifique à chaque champ de couplage, comme par exemple leur période de couplage, la liste des transformations ou interpolations auxquelles doit procéder OASIS3, etc. La seconde partie de la namcouple est la plus importante.

La namcouple est à adapter à chaque test, pour chaque paire de grilles. La première partie de la namcouple reste inchangée, une fois que les paramètres ont été établis pour une utilisation d'OASIS3 en mode interpolateur. La seconde partie connait cependant des changements notables lorsqu'on change de paire de grilles.
Enumérons alors les caractéristiques de ces grilles.

2.1.2 Les grilles, représentées par des fichiers NetCDF

Toute l'information dont OASIS3 a besoin à propos des grilles sources et cibles est stockée dans 3 fichiers au format NetCDF : *grids.nc*, *masks.nc* et *areas.nc*. Les 4 grilles que nous avons utilisées sont complètement contenues dans ces 3 fichiers. Donnons quelques précisions sur les caractéristiques de ce format.

DEFINITION 2.1.1. Le format de données NetCDF est "auto-documenté", c'est-à-dire qu'il existe un en-tête qui décrit la disposition des données dans le reste du fichier, et en particulier des tableaux de données. Cet en-tête contient aussi une liste arbitraire de métadonnées se présentant sous la forme d'attribut de type nom/valeur. Les tableaux de données sont linéaires et sont stockés de manière simple de façon à permettre un accès efficace à un sous-ensemble d'un tableau.
Ce format est couramment utilisé dans des applications de climatologie, de météorologie et d'océanographie (prévision météorologique, changement climatique, ...).

Dans le tableau 2.1, nous présentons la façon dont sont représentées les grilles que nous possédons. Ces 3 fichiers NetCDF contiennent toute l'information sur les grilles source et cible à tester.

grids.nc	masks.nc	areas.nc
contient les longitudes et les latitudes des coins et centre des mailles	contient 0 : si la maille est sur l'océan, 1 : si elle est sur une surface terrestre	contient l'aire de la maille de coordonnées (lat,lon) telle que considérée par le modèle (en m^2)

FIG. 2.1 – Le contenu général des fichiers NetCDF.

2.1.3 Les caractéristiques des grilles utilisées

Pour tester la méthode d'interpolation conservative, nous avons utilisé 8 grilles ayant des caractéristiques différentes. Nous les avons synthétisées dans le tableau 2.2.

Nom de la grille	Nombre de mailles	Longueur moyenne de maille	Nombre de coins	Surfaces de mailles
torc	149 latitudes et 182 longitudes, soit 27 118 mailles	1.2°, soit 135 km	4	disponibles
bt42	6 232 mailles numérotées selon une gaussienne réduite	2.8°, soit 310 km		
at42	64 latitudes et 128 longitudes, soit 8 192 mailles	2.8°, soit 310 km		
lmdz	72 latitudes et 96 longitudes, soit 6 912 mailles	2.5°, soit 280 km	non enregistrés	non disponibles
tne1 équivalente à torc en plus haute résolution	292 latitudes et 362 longitudes, 105 704 mailles	0.62°, soit 68 km	4	disponibles
t127 équivalente à bt42 en plus haute résolution	24 572 mailles numérotées selon une gaussienne réduite	0.47°, soit 52 km		
grdo et grda grilles créées pour les tests	2 latitudes et 2 longitudes, soit 4 mailles	entre 0.25° et 4°, soit entre 28 et 445 km		non disponibles

FIG. 2.2 – Les caractéristiques des grilles utilisées.

Le sous répertoire testNONE/INPUT contient donc essentiellement les 3 fichiers NetCDF *grids.nc*, *areas.nc*, *masks.nc* hébergeant toute l'information nécessaire sur les grilles que nous utilisons. Nous rappelons qu'il contient aussi le fichier de configuration *namcouple*.
Voyons maintenant les caractéristiques des fichiers contenus dans le répertoire testNONE/PROG ainsi que le script *sc_run_NONE*.

2.1.4 Le dossier PROG et le script *sc_run_NONE*

Le dossier PROG contient deux fichiers. Le premier, *create_inputfield.f90* crée un champ en calculant les valeurs d'une fonction analytique aux points de maille d'une grille source passée en argument. Le second, *calc_errorfield.f90* permet de calculer l'erreur entre le champ interpolé et la valeur de la fonction analytique calculée aux points de mailles de la grille cible. Trois fonctions tests y ont été implémentées. Le script *sc_run_NONE* est celui que nous lançons pour la compilation d'OASIS3 et de ses bibliothèques. Nous y précisons notamment les noms des grilles source et cible qui doivent être les mêmes que ceux que nous écrivons dans la *namcouple* et la fonction analytique à tester. Les étapes du fichier script se présentent sous la forme suivante.

```
 ######################################################################
#                                                                      #
#  1°) Définition des répertoires de travail                           #
#  2°) Définition des emplacements des fichiers NetCDF                 #
#  3°) Choix des grilles source et cible                               #
#  4°) Choix de la fonction analytiquetest (1, 2 ou 3)                 #
#  5°) Création du répertoire de travail                               #
#  6°) Copie des fichiers nécessaires dans le répertoire de travail    #
#  7°) Compilation et exécution du programme créant le champ d'entrée  #
#  8°) Compilation et exécution d'OASIS3 en mode interpolateur         #
#  9°) Compilation et exécution du programme calculant                 #
#      le champ d'erreur sur le champ interpolé                        #
# 10°) Représentation des champs d'erreur avec ferret                  #
#                                                                      #
 ######################################################################
```

Les grilles sources et cibles sont indiquées dans ce script par les affectations «SRC_GRID» pour la grille source et «TGT_GRID» pour la grille cible. Le choix de la fonction test servant au calcul du champ analytique sur les deux grilles se fait dans la partie «export FLD_NBR=1».
Les 3 fonctions disponibles proposées par le test sont les suivantes :

$$
\begin{aligned}
\text{(1) } F &= 2 + \cos[\pi * acos(\cos(lat) * \cos(lon))] \\
\text{(2) } F &= 2 + [(\cos(lat))^2] * \cos(2 * lon) \\
\text{(3) } F &= 2 + [(\sin(2 * lon))^{16}] * \cos(16 * lon)
\end{aligned}
$$

Une fois choisie, cette fonction va être calculée aux centres des mailles sources et cibles. Ainsi, grâce aux méthodes *create_inputfield.f90* et *calc_errorfield.f90*, on va comparer ces valeurs à celles calculées par la méthode de l'interpolation conservative de la librairie SCRIP.

Ainsi, par la suite, le logiciel Ferret va permettre de visualiser les champs et les erreurs sur une échelle de couleur pour chaque maille cible. Les fichiers de commandes ferret utilisés sont détaillés à la figure 2.3

L'échelle est déterminée par la commande «lev=(-0.025,0.025,0.001)». Ici, on se place sur une échelle où l'erreur relative se trouve entre -2.5% et +2.5% avec un pas de 0.1 point.

use error.nc plot 'BT42.LON' go polymark polygon/lev=(-0.025,0.025,0.001) 'BT42.LON' 'BT42.LAT' ERROR square 0.6 frame/file=error_2.gif	use error.nc shade/lev=(-0.025,0.025,0.001) ERROR frame/file=error_2.gif

FIG. 2.3 – Fichiers de commandes pour le tracé des erreurs : une grille cible bt42 (à gauche) et les 3 autres (à droite).

2.2 La méthode d'interpolation conservative

La fonction « conservative remapping » de la bibliothèque SCRIP est très importante dès lors qu'elle joue un rôle essentiel dans l'échange des valeurs des champs de couplage entre des grilles provenant de modèles différents.

2.2.1 L'interpolation conservative : qu'est-ce que c'est ?

Dans le but d'étudier le système climatique terrestre, les modèles climatiques ont souvent été créés par le couplage de modèles développés individuellement. Chacun de ces modèles simule l'océan, l'atmosphère, glace ou la surface terrestre. Ces modèles ont été mis en place indépendamment les uns des autres ou, certaines fois, par paire (par exemple, océan-atmosphère, océan-glace) et utilisent des grilles étudiées pour ces cas précis.

Dans un modèle climatique couplé, les variables d'état, les échanges de chaleur et d'eau doivent être communiqués périodiquement d'un modèle à l'autre. Les grilles de chaque modèle étant différentes, **il est nécessaire d'interpoler** ces valeurs à chaque fois qu'il y a échange d'informations entre les modèles. En particulier, les flux doivent être interpolés de façon conservative pour assurer la conservation de l'énergie et de l'eau du système climatique couplé. A présent, beaucoup de modèles climatiques utilisent une façon d'interpoler basée sur la fraction d'aire intersectée entre les mailles source et cible. Une telle méthode est automatiquement conservative.

Pour implémenter un flux F sur une nouvelle grille (grille cible) qui résulte d'un flux f sur une ancienne grille (grille source), sur une maille cible k, F doit satisfaire[2] :

$$\bar{F_k} = \frac{1}{A_k} \iint_{A_k} f dA \tag{2.1}$$

où :

- $\bar{F_k}$ est le flux moyen sur la maille cible k,
- A_k est l'aire de la maille cible k.

[2]Le chapitre 3 (Conservative remapping) du manuel d'utilisation de la SCRIP figure en annexe 2 de ce rapport

Si l'équation (2.1) est satisfaite, alors la condition de conservation globale, la sera aussi (équation (2.2)) :

$$\iint_A F dA = \iint_A f dA \tag{2.2}$$

où l'intégrale est faite sur la totalité de la surface A de la grille.

Puisque, dans l'équation (2.1), l'intégrale est faite sur la surface de la maille cible, seulement les mailles de la grille source intersectant des mailles cibles vont y contribuer.
Si la maille cible k intersecte N mailles sources, on peut écrire l'équation (2.1) de la façon suivante :

$$\bar{F_k} = \frac{1}{A_k} \sum_{n=1}^{N} \iint_{A_{nk}} f_n dA \tag{2.3}$$

où :

- A_{nk} est l'aire de la maille cible k couverte par la maille source n,
- f_n est la valeur locale du flux sur la maille source n.

Sous l'hypothèse où la fonction f_n est constante sur chaque maille n de la grille source (ainsi $f_n = \bar{f_n}$), de l'équation (2.3) découle le schéma du premier ordre que nous utilisons dans les modèles couplés.

$$\bar{F_k} = \sum_{n=1}^{N} \bar{f_n} \omega_{nk} \tag{2.4}$$

où :

$$\omega_{nk} = \frac{1}{A_k} \iint_{A_{nk}} dA \tag{2.5}$$

En choisissant une fonction appropriée, le théorème de la divergence donne l'équation 2.6,

$$\iint_{A_{nk}} dA = \oint_{C_{nk}} -\sin(lat) dlon \tag{2.6}$$

où C_{nk} est le chemin orienté dans le sens trigonométrique autour de A_{nk}.

Ainsi, nous aurons à calculer les coefficients ω_{nk}.

$$\omega_{nk} = \frac{1}{A_k} \oint_{C_{nk}} -\sin(lat) dlon \tag{2.7}$$

Notons que pour simplifier les calculs, la SCRIP considère que $\sin(lat)$ est une fonction linéaire de lon.

2.2.2 Les étapes de la méthode

Le code de l'interpolation conservative contient un balayage de chaque maille source et un autre de chaque maille cible. Ainsi, il est possible de calculer les poids correspondant aux ω_{nk} cités dans la section précédente.
Les étapes de la méthode d'interpolation conservative sont les suivantes :

```
1°) Premier balayage de tous les bords des mailles sources
2°) Recherche des intersections avec les bords de mailles cibles
3°) Calcul et stockage des intégrales de ligne des segments trouvés

4°) Second balayage de tous les bords des mailles cibles
5°) Recherche des intersections avec les bords de mailles sources
6°) Calcul et stockage des intégrales de ligne des segments trouvés
```

Figure 3.1: An example of a triangular destination grid cell k overlapping a quadrilateral source grid. The region A_{kn} is where cell k overlaps the quadrilateral cell n. Vectors used by search and intersection routines are also labelled.

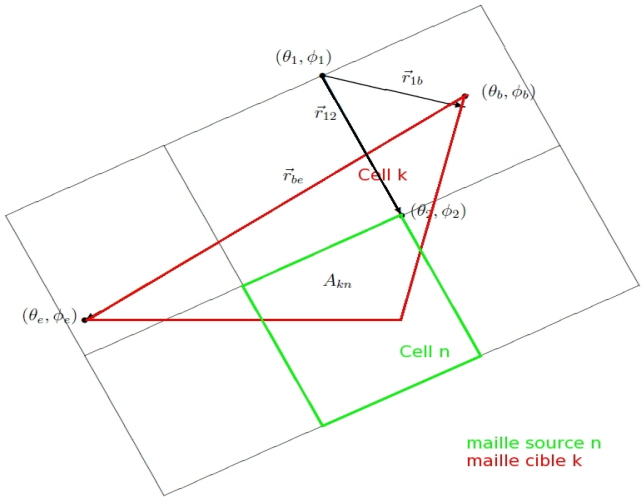


FIG. 2.4 – Exemple de mailles source et cible.

Pour effectuer ces calculs, la SCRIP ne possède pas la définition exacte des bords des mailles mais seulement la localisation des coins des mailles. Pour le calcul des intersections des coins des bords, la SCRIP suppose que les bords des mailles sont linéaires en longitude et latitude entre deux coins consécutifs d'une maille. Comme cette hypothèse devient de moins en moins valable au fur et à mesure qu'on s'approche des pôles, notons que l'utilisateur peut préciser une latitude (en radians) à partir de laquelle les mailles sources et cibles doivent être projetées. Dans ce cas, les calculs d'intersections se font dans le repère projeté, mais pas ceux d'intégrale de ligne.

Le calcul de l'intégrale de ligne entre les 2 extrémités, c'est-à-dire le point de coordonnées (beglon, beglat) et le point d'intersection, est en fait l'aire se trouvant sous la courbe délimitée par ces points et l'équateur. Ainsi, en ajoutant ces valeurs pour chaque coté d'une maille, on obtient l'aire de celle-ci.

Cette option, lorsqu'elle est activée, permet de projeter les mailles se trouvant au-dessus d'une latitude donnée, appelée « north_thresh ». Il s'agit de la projection azimutale équivalente de Lambert. Celle-ci permet de projeter une sphère sur un plan en conservant localement les surfaces mais pas les angles. C'est aussi une façon de représenter entièrement la surface de la Terre sous la forme d'un disque. Il s'agit donc d'une projection cartographique conçue en 1772 par le mathématicien alsacien Johann Heinrich Lambert.

2.2.3 Explications à propos du nombre de « bins »

Lors de l'écriture de la *namcouple*, nous avons défini dans la seconde partie l'option « LATLON » avec le numéro « 10 ». Cette option[3] a été pensée pour restreindre le domaine de recherche des mailles cibles susceptibles d'intersecter les bords des mailles sources du premier balayage, et aussi restreindre le domaine de recherche des mailles sources susceptibles d'intersecter les bords des mailles cibles du deuxième balayage.

Lors du premier balayage, les « bins » interviennent pour restreindre le domaine de recherche des mailles cibles à prendre en compte pour le calcul des points d'intersection. Avec l'option « LATLON » dans la namcouple et une valeur de $NBIN$ fixée, les grilles source et cible sont partagées en $NBIN^2$ boîtes. De ce fait, lors du premier balayage, pour chaque maille source, la SCRIP détermine dans quelle zone (boîte) elle se trouve.

Ainsi, la SCRIP sait dans quelle zone cible vont se trouver les mailles susceptibles de s'intersecter avec cette maille source. On réduit alors le domaine de recherche d'intersection de la grille toute entière à la boîte. Si, par exemple, $NBIN = 10$, alors les grilles vont être partagées en 100 boîtes et la recherche se fera de façon beaucoup plus ciblée. Après cette première restriction, une seconde est effectuée pour diminuer encore le domaine de recherche. Des restrictions de domaines similaires sont aussi utilisées lors du second balayage.

2.3 Motivations du stage : les défauts de l'interpolation conservative

Afin d'évaluer la qualité de l'interpolation conservative, exécutons le code en notre possession sur les grilles : bt42 en source, torc en cible. La fonction test est la première listée ci-dessus.
Une fois le script lancé, son exécution génère des fichiers images représentant l'erreur. Nous pouvons constater sur la figure 2.5 , celle provenant de l'exécution du fichier donné à droite en figure 2.3.

Dans un premier temps, étudions les résultats de l'interpolation conservative sans activer la projection (cf. section 2.2.2).

[3]Pour plus de détails sur les bins le lecteur pourra se reporter à l'annexe 3 de ce rapport

Nous remarquons sur la figure 2.5 que quelques mailles ont une erreur élevée : en plus de quelques mailles près des côtes qui ont des erreurs élevées liées à des masques terre-mer source et cible différents (ce qui ne fait pas l'objet de notre étude), on peut noter que certaines mailles du pôle nord présentent des erreurs relatives de l'ordre 1,4%. Or, lors de l'exécution des modèles, une attention toute particulière est portée sur les échanges de champs de couplage ayant lieu aux pôles. De ce fait, on ne peut pas accepter qu'une erreur relative de l'ordre de 1,4% puisse se produire sur des mailles se trouvant au pôle.

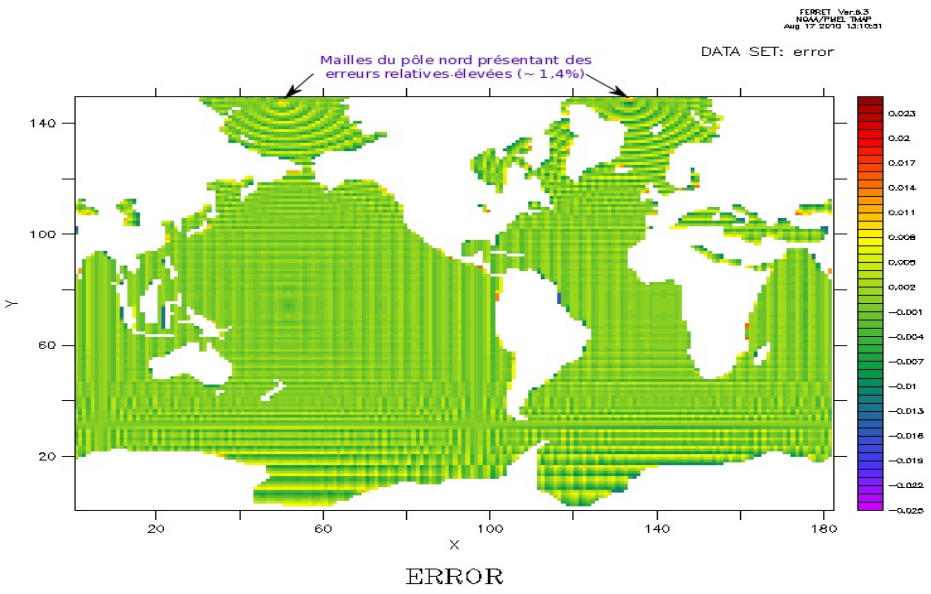


FIG. 2.5 – Erreur d'interpolation, sans projection, de bt42 sur torc.

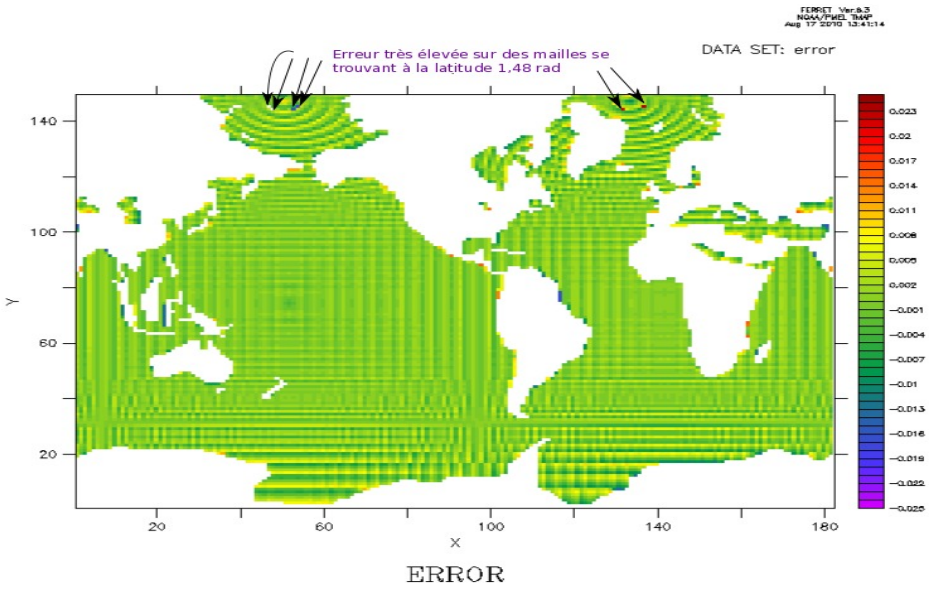


FIG. 2.6 – Erreur d'interpolation, avec projection, de bt42 sur torc.

Afin de corriger cette erreur observée au pôle nord, activons maintenant la projection pour le calcul des intersections au-delà de 1.48 radians.

Lorsqu'on décide de projeter les mailles se trouvant au-dessus de la latitude 1,48 radians, nous obtenons le résultat illustré par la figure 2.6. Visiblement, l'erreur observée au pôle sur la figure 2.5 s'est atténuée avec l'effet de la projection. Cependant, comme le montre la figure 2.6, certaines mailles se trouvant sur la latitude 1,48 radians ont vu leur erreur relative augmenter considérablement. En effet, l'erreur relative de ces mailles est de 5%.

La cause de cette nouvelle erreur est très clairement un problème de raccordement. Le code gère mal les mailles intersectant « north_thresh » car le calcul des intersections est fait dans l'espace projeté pour certains de leurs bords et dans l'espace non-projeté pour les autres. Une telle incertitude ne peut évidemment pas être tolérée.

Nous pouvons aussi regarder le cas des grilles torc en source et bt42 en cible. Ce qui correspondrait à l'interpolation inverse de ce que nous venons d'étudier.

Lorsque nous exécutons le code d'interpolation sans projection, de la grille torc sur la bt42, nous observons sur la figure 2.7 que 4 mailles se trouvant au pôle nord ont une erreur relative de l'ordre de -1,8% et une présentant une erreur légèrement inférieure. Nous voudrions corriger cette inexactitude et exécutons alors le code avec l'option de projection, toujours avec « north_thresh = 1,48 radians ». Nous constatons, cette fois, sur la figure 2.8, que 3 des 5 mailles voient leur erreur relative diminuer franchement tandis que les 2 autres l'ont augmentée. D'autre part, 4 mailles qui se comportaient bien dans le cas sans projection se sont dégradées avec la projection. L'erreur relative atteint pour le cas avec projection la valeur de -2.7%.

Nous gagnons donc en précision sur certaines mailles alors que nous en perdons sur d'autres. D'autant plus que ce processus semble se faire de façon aléatoire, c'est-à-dire que rien, à priori, ne nous permet de deviner quelles seront les mailles allant bien se comporter avec la projection et lesquelles seront dégradées.

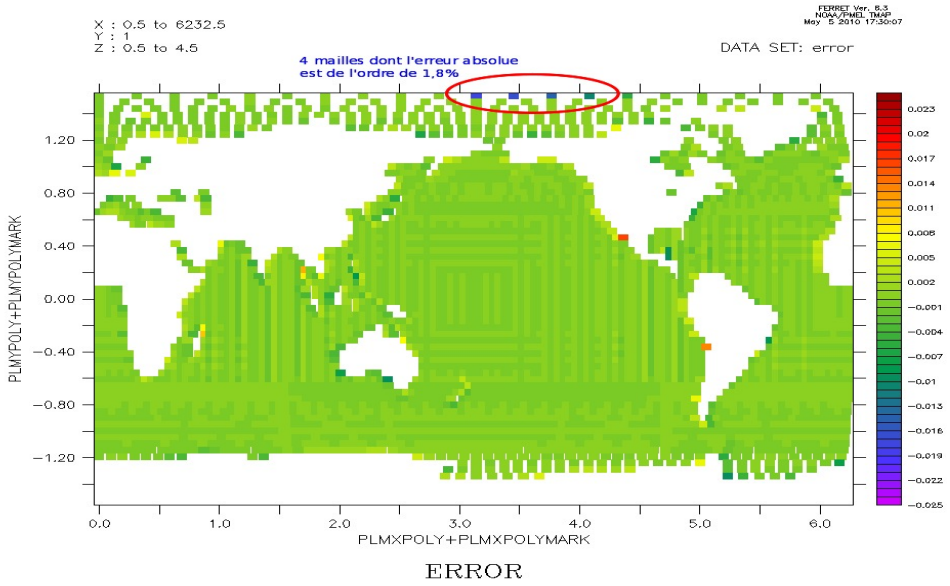


FIG. 2.7 – Erreur d'interpolation, sans projection, de torc sur bt42.

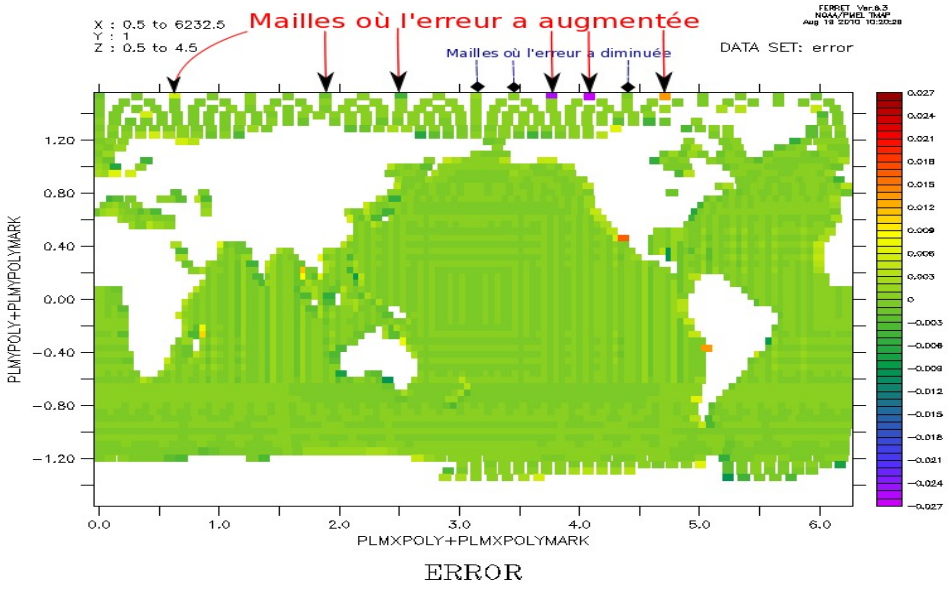


FIG. 2.8 – Erreur d'interpolation, avec projection, de torc sur bt42.

Afin de corroborer ou d'abroger ces constats, exécutons aussi l'interpolation sur des grilles à plus haute résolution, c'est-à-dire à maillage plus fin : tne1 (équivalente de torc) et t127 (équivalente de bt42).

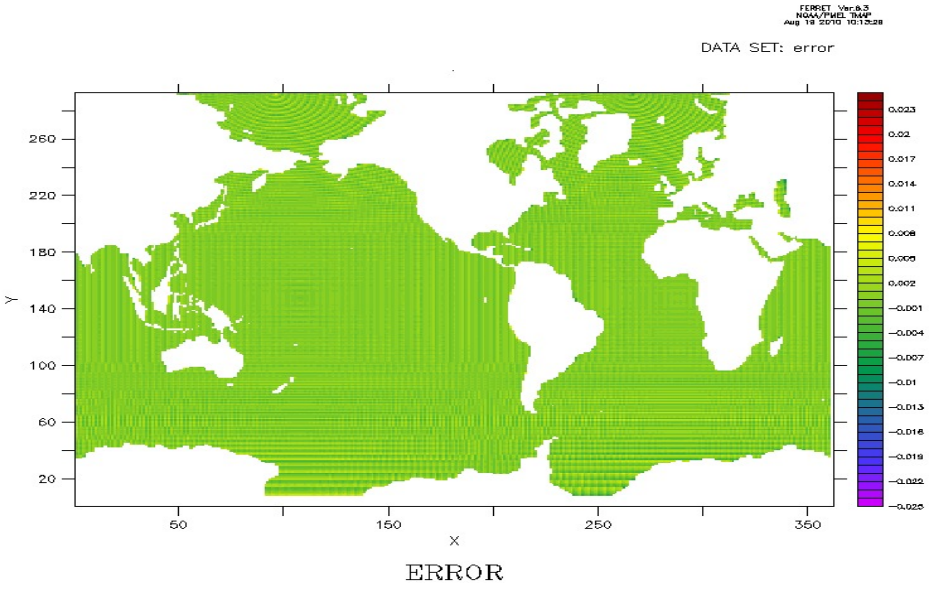


FIG. 2.9 – Erreur d'interpolation, sans projection, de t127 sur tne1.

Sans projection, de t127 sur tne1, nous observons sur la figure 2.9, des erreurs faibles pour les mailles du pôle nord. Dans ce cas, la méthode d'interpolation semble fonctionner comme souhaité. L'erreur relative maximale est de 0.7%.

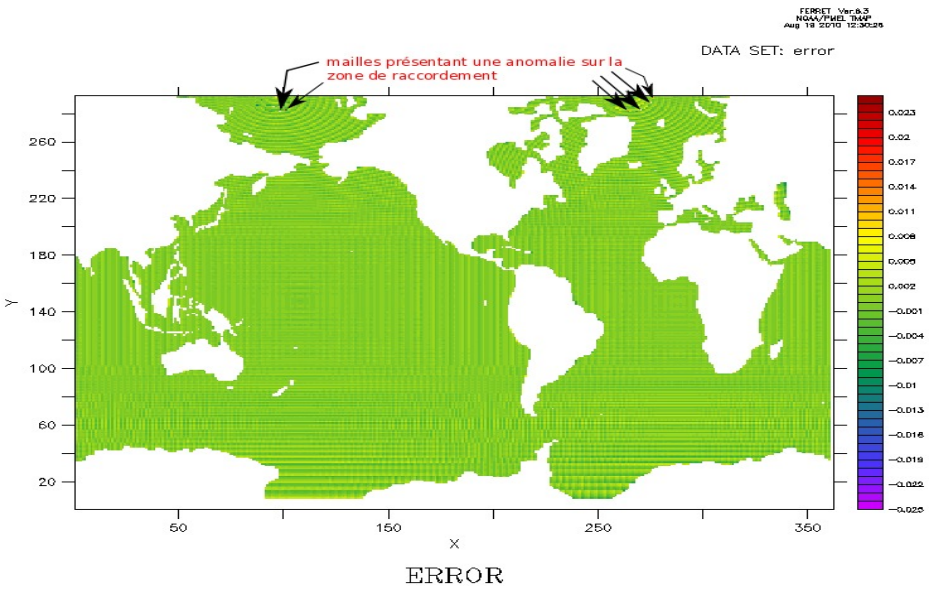


FIG. 2.10 – Erreur d'interpolation, avec projection, de t127 sur tne1.

Pour l'interpolation avec projection de t127 sur tne1 (north_thresh=1,48 radians), nous observons sur la figure 2.10, des mailles où l'erreur relative a considérablement augmenté pour atteindre 25% sur une des mailles. Toutes ces mailles sont sur la même latitude, sur la frontière de projection. Le problème du

raccordement se retrouve encore une fois ici.

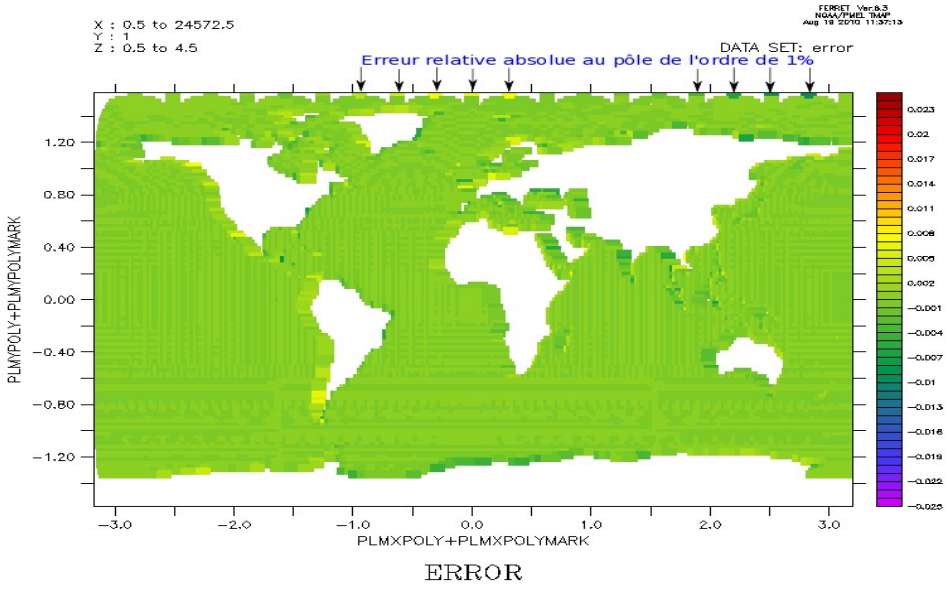


FIG. 2.11 – Erreur d'interpolation, sans projection, de tne1 sur t127.

Pour l'interpolation sans projection, de tne1 sur t127, nous observons sur la figure 2.11, des erreurs pour certaines mailles proches du pôle nord de l'ordre de 0.94% pour les maximales (celles marquées par des flèches).

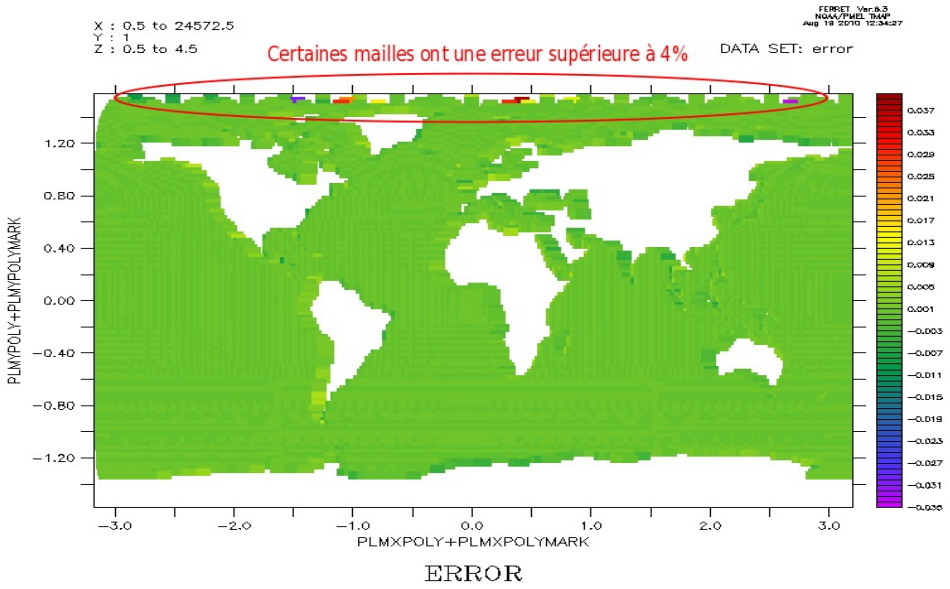


FIG. 2.12 – Erreur d'interpolation, avec projection, de tne1 sur t127.

Pour l'interpolation de tne1 sur t127 avec projection (north_thresh=1,48 radians), la figure 2.12 montre des mailles où l'erreur augmente considérablement. La projection n'améliore pas la situation des mailles se trouvant au-delà de north_thresh puisque l'erreur relative atteint 6.5% pour certaines des mailles se trouvant dans la zone entourée (cf. figure 2.12).

Toutes ces observations montrent que nous sommes en présence d'un certain nombre de conflits :

- sans projection : certaines mailles du pôle ne sont pas traitées correctement (voir les figures 2.5, 2.7 et 2.11), puisqu'elles présentent des erreurs relatives considérablement élevées,
- avec projection : certaines mailles présentant une anomalie sont corrigées alors que de nouvelles mailles pathogènes apparaissent (voir les figures 2.8 et 2.12). La projection n'améliore donc pas toujours l'erreur des mailles du pôle,
- avec projection : même lorsque de nouvelles mailles à problème n'apparaissent pas au pôle, nous sommes confrontés à un problème de raccordement des mailles se trouvant sur la frontière identifiée par « north_thresh » (voir les figures 2.6 et 2.10).

Notre but est bien évidemment de proposer des explications et des solutions à ces différents problèmes. L'interpolation conservative ne peut pas admettre d'avoir d'aussi grandes erreurs aux pôles puisque les modèles portent une attention toute particulière aux échanges de champs de couplage se produisant aux pôles. Ainsi, en passant d'une grille (utilisée dans un modèle) à une autre (utilisée dans un autre modèle), on ne peut pas se permettre de perdre de l'information sur les valeurs aux pôles. De ce fait, il est nécessaire d'identifier ces pathologies présentent dans la méthode d'interpolation conservative et ainsi assainir la bibliothèque SCRIP.

Dans la prochaine section, étudions alors en détail comment procède la méthode d'interpolation conservative afin de mieux comprendre les processus qui la régisse, d'identifier les problèmes et de réfléchir à des solutions.

3 ÉTUDE DES ANOMALIES OBSERVÉES AU PÔLE NORD ET PROPOSITION DE SOLUTION

Dans cette troisième partie, nous présenterons la cause des erreurs sur certaines mailles du pôle nord. Puis, nous proposerons de faire une rotation des mailles polaires afin de remédier à ces difficultés.

3.1 Explications sur les constats effectués dans la partie 2

3.1.1 Description du système de mailles testées

Tout d'abord, rappelons que dans le cas de l'interpolation conservative, le but de la librairie SCRIP est de retourner les valeurs des poids pour chaque couple de mailles source-cible qui s'intersectent. Ces poids correspondent à la proportion de chaque maille cible intersectée par une maille source, comme illustré sur la figure 3.1.

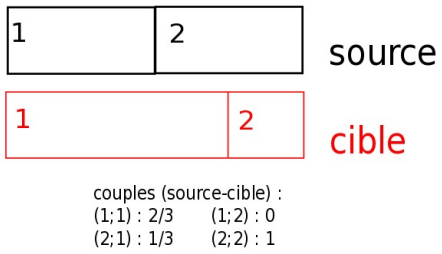


FIG. 3.1 – Exemple de calcul de poids.

Nous voulons connaitre la précision des méthodes pour les mailles intersectant le pôle ou pour celles qui y sont proches. Pour cela, nous avons défini deux grilles théoriques pour tester les méthodes que nous avons à notre disposition.
La grille source a été pensée de telle sorte que ses mailles recouvrent complètement les mailles de la grille cible. Dans un souci de facilité quand à la vérification des résultats, nous avons disposé les mailles de la façon suivante :

– 4 mailles sources identiques, disposées deux à deux de part et d'autre de l'équateur,

– 2 mailles cibles complètement recouvertes par les mailles sources, la première étant centrée sur l'équateur,
– les mailles sources intersectent chacune pour $\frac{1}{4}$ la maille cible 1,
– la SCRIP n'acceptant pas de grille à maille unique, nous ajoutons la maille cible 2. Cette dernière est aussi entièrement recouverte par les mailles sources.
– Appelons (A), l'unique point sur lequel se trouve un coin de chacune des mailles des 4 mailles sources ((A) est aussi le centre de la maille cible 1).

Ainsi nous nous attendons à ce que le poids soient de 0.25 pour les intersections des mailles sources-cibles (1 ;1), (2 ;1), (3 ;1), (4 ;1) et de 0.5 pour les couples (3 ;2) et (4 ;2). La figure 3.2 présente ce cas pour des mailles sources et cibles dont les coins sont écartés de 4 degrés en longitude et latitude.

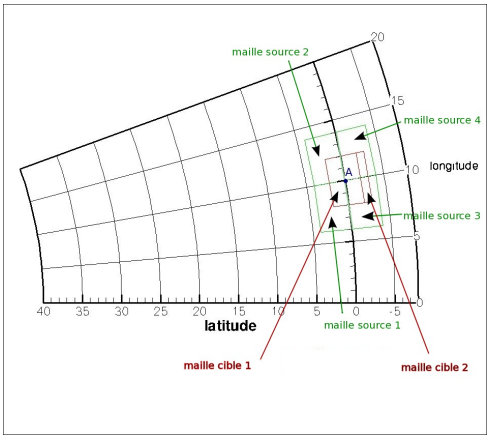


FIG. 3.2 – Maillage théorique : mailles de 4°, (A) placé sur l'équateur.

Afin d'évaluer l'impact de la taille des mailles dans la précision des méthodes de calcul, différentes mailles seront étudiées ; elles auront respectivement des coins espacés de 4°, 2°, 1°, 0.5° et 0.25°, en longitude et latitude. On sait d'ores et déjà que les maillages des modèles couplés réels qui seront mis en œuvre dans les prochaines années tendront à atteindre cette dernière valeur.

Différents tests seront effectués en plaçant le centre du système (A) à des latitudes variant :
– pour les mailles de 4° : entre 60° et 90°, avec un pas de 1°,
– pour les mailles de 2° et 1° : entre 70° et 90°, avec un pas de 1°,
– pour les mailles de 0.5° : entre 80° et 90°, avec un pas de 0.5°,
– pour les mailles de 0.25° : entre 84° et 90°, avec un pas de 0.25°.

Dès lors, nous sommes en présence de 41 grilles sources et 41 grilles cibles, soit 82 grilles ! Ce sont celles que nous avons appelées génériquement grdo et grda, dans la section précédente. Nous avons alors créé 3 fichiers NetCDF par grille (*grids.nc*, *masks.nc* et *areas.nc*) et une *namcouple* afin de les fournir en entrée de la SCRIP. Nous donnons 2 exemples de ces maillages représentés sur les figures 3.3 et 3.4.

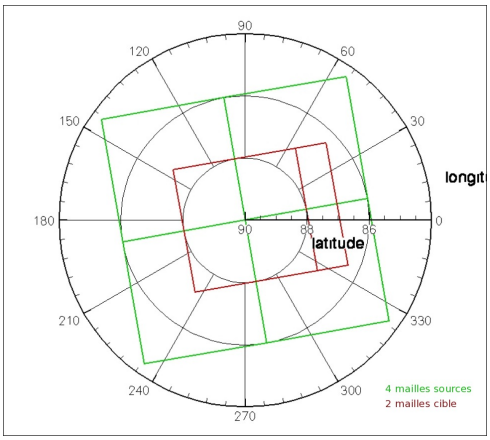


FIG. 3.3 – Maillage théorique : mailles de 4°, (A) placé à 90°.

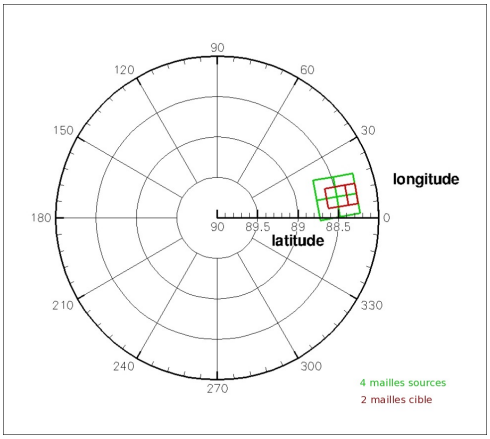


FIG. 3.4 – Maillage théorique : mailles de 0.25°, (A) placé à 88.5°.

3.1.2 Illustration des bords des mailles considérés par la SCRIP, sans projection

Rappelons que la SCRIP suppose que les bords des mailles sont linéaires en longitude et latitude entre deux coins consécutifs d'une maille. Evaluons ici l'impact de cette hypothèse près du pôle lorsque les calculs se font sans projection de Lambert.

Pour ce faire, sur les grilles source et cible de 4° où (A) est à 90°, nous avons implémenté une fonction Fortran permettant de visualiser « le chemin suivi par la fonction 'intersect' contenu dans le module remap conserv ». La figure 3.5 en donne une illustration.

Ainsi, nous montrons que les mailles près du pôle sont très déformées et en particulier «le chemin reliant un coin situé au pôle» dépend de la longitude qui a été enregistrée pour ce point au pôle. Ce résultat parait innattendu puisque n'importe quelle longitude est valide pour un point au pôle, à partir du moment où sa latitude est de 90°. La courbure suivie est en fait tangente à la longitude enregistrée

pour le point au pôle. Dès lors, l'hypothèse de linéarité de bords des mailles fait que la SCRIP considère en fait des mailles déformées près des pôles. Ces déformations sont la source des erreurs que nous avons observées dans la partie précédente, pour les calculs sans projection, notamment sur les figure 2.7 et 2.11.

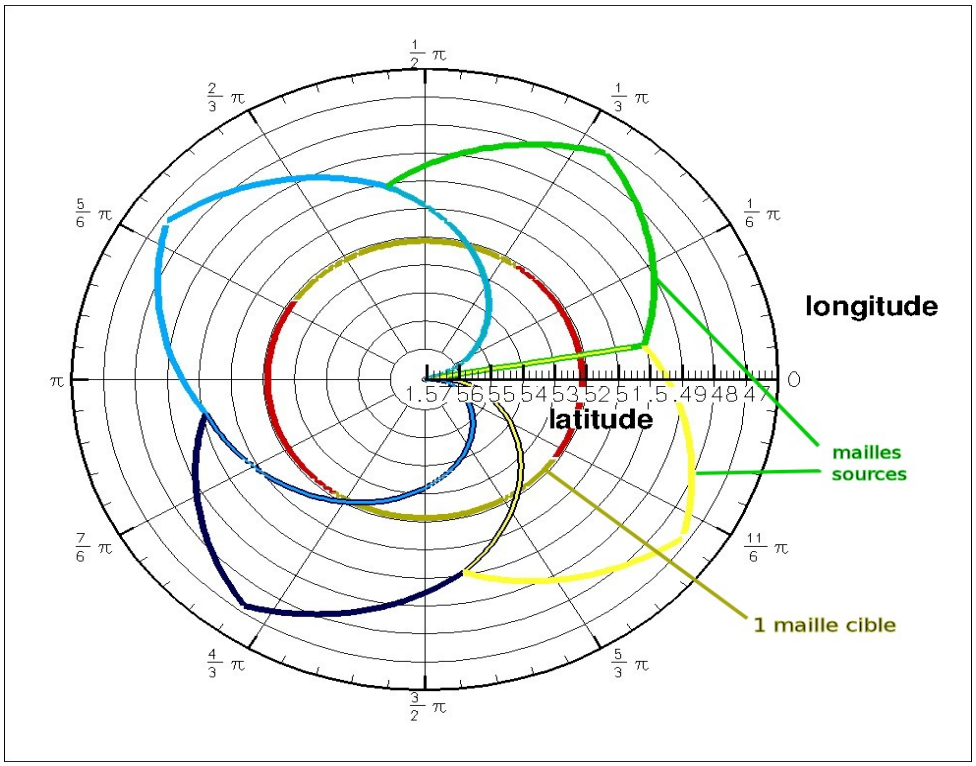


FIG. 3.5 – Les mailles sources et cibles telles que vues par la SCRIP (4°, (A) placé à 90°).

3.1.3 Considérations de la SCRIP, avec projection pour le calcul des intersections

Les anomalies observées dans le cas de la projection sont dûes d'une part à un problème de raccordement entre la zone projetée et celle non projetée, comme nous l'avons observé sur les figures 2.6 et 2.10.

D'autre part, il s'avère sur les figures 2.8 et 2.12 que les résultats sur les mailles du pôle ne sont pas toujours meilleurs, indépendamment du problème de raccordement. Ceci est dû au fait qu'après avoir fait la projection pour trouver les intersections, les calculs d'aires sont réalisés dans le repère sphérique initial. C'est-à-dire qu'une fois les intersections de bords trouvées dans le repère projeté, la transformation inverse de la projection est réalisée pour les calculs d'aires. Les calculs que nous avons effectués sur les mailles théoriques de 4°, avec le point (A) placé à 90°, retournent les résultats présentés dans le tableau 3.6. Précisons, que dans ce cas, il ne peut pas demeurer de problèmes de raccordement puisque nous initialisons $north\_thresh = 0.5rad$. Les mailles se trouvant sur le pôle nord (figure 3.3), à une latitude supérieure à 84°, la latitude de 0.5 rad (soit environ 28.6°) suffit pour s'assurer que toutes les mailles seront projetées.

maille source	maille cible	poids	Erreur relative
1	1	0.5001	100.0206
2		0.1665	33.3943
3		0.1667	33.3131
4		0.1667	33.3131

FIG. 3.6 – Avec projection : erreur relative sur les mailles théoriques de 4°, (A) placé à 90°. Les poids exacts devraient être de 0.25 chacun.

Alors que nous nous attendions à 4 poids de 0.25 pour la maille cible 1, les résultats retournés ne sont pas satisfaisants. Nous avons vérifié que les intersections trouvées entre les bords des mailles sources et cibles, calculées dans l'espace projeté, sont justes. Nous déduisons donc que l'erreur observée ici est du à un mauvais calcul d'intégral de ligne effectué lui dans un espace non projeté. Ainsi, nous montrons qu'au problème de raccordement s'ajoute la question de la fiabilité du calcul de l'intégrale de ligne dans l'espace sphérique initial. Plus précisément, nous savons que cette façon de procéder avec la projection pour le calcul des intersections seul n'est pas robuste puisque sur certaines mailles proches du pôle les poids retournés ne sont pas réalistes, alors que la raison de la mise en place de l'option de projection était justement de pouvoir s'affranchir des énormes erreurs obtenues pour ces mailles précises.

3.2 Notre solution, c'est la rotation

Puisque les mailles se trouvant au pôle sont interprétées suivant la longitude de leurs coins, et que plus la maille est proche du pôle, plus elle a tendance à être déformée, nous avons décidé de s'en affranchir en opérant une rotation sur les mailles se trouvant aux alentours du pôle nord.

3.2.1 La rotation

La rotation permettra de ramener toutes les mailles proches du pôle nord vers l'équateur pour ainsi minimiser les erreurs de considération de la SCRIP (notamment le fait que les bords sont considérés linéaires en (lat,lon)).

Soient,

$$X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} r\cos(lon)\cos(lat) \\ r\sin(lon)\cos(lat) \\ r\sin(lat) \end{pmatrix},$$

$$X' = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} r\cos(\lambda)\cos(\phi) \\ r\sin(\lambda)\cos(\phi) \\ r\sin(\phi) \end{pmatrix}$$

où X' est le point issu de la rotation de X, selon l'axe $\vec{e_2}$ (orthogonal au plan (xOz)) et d'angle $reflat$.

λ (la longitude du point X') et ϕ (la latitude du point X') sont les inconnues.

X et X' sont reliés par la formule suivante :

$$X' = \begin{pmatrix} \cos(reflat) & 0 & \sin(reflat) \\ 0 & 1 & 0 \\ -\sin(reflat) & 0 & \cos(reflat) \end{pmatrix} X$$

D'où,

$$\begin{pmatrix} r\cos(\lambda)\cos(\phi) \\ r\sin(\lambda)\cos(\phi) \\ r\sin(\phi) \end{pmatrix} = \begin{pmatrix} \cos(reflat) & 0 & \sin(reflat) \\ 0 & 1 & 0 \\ -\sin(reflat) & 0 & \cos(reflat) \end{pmatrix} \begin{pmatrix} r\cos(lon)\cos(lat) \\ r\sin(lon)\cos(lat) \\ r\sin(lat) \end{pmatrix}$$

Puis,

$$\begin{cases} \cos(\lambda)\cos(\phi) & = \cos(lon)\cos(lat)\cos(reflat) + \sin(lat)\sin(reflat) \\ \sin(\lambda)\cos(\phi) & = \sin(lon)\cos(lat) \\ \sin(\phi) & = -\cos(lon)\cos(lat)\sin(reflat) + \sin(lat)\cos(reflat) \end{cases}$$

Ainsi, nous pouvons adapter cette formule pour toute rotation d'angle $reflat$ et d'axe $\vec{e_2}$.
$\vec{e_2}$ étant l'axe issu de la rotation de $\vec{e_2}$ selon un angle $reflon$ et d'axe $\vec{e_3}$.

$$\begin{cases} \cos(\lambda - reflon)\cos(\phi) & = \cos(lon - reflon)\cos(lat)\cos(reflat) + \sin(lat)\sin(reflat) \\ \sin(\lambda - reflon)\cos(\phi) & = \sin(lon - reflon)\cos(lat) \\ \sin(\phi) & = -\cos(lon - reflon)\cos(lat)\sin(reflat) + \sin(lat)\cos(reflat) \end{cases}$$

Enfin, si $\cos(\phi) \neq 0$,

$$\begin{cases} \cos(\lambda - reflon) & = \dfrac{1}{\cos(\phi)}(\cos(lon - reflon)\cos(lat)\cos(reflat) + \sin(lat)\sin(reflat)) \\ \sin(\lambda - reflon) & = \dfrac{1}{\cos(\phi)}\sin(lon - reflon)\cos(lat) \\ \sin(\phi) & = -\cos(lon - reflon)\cos(lat)\sin(reflat) + \sin(lat)\cos(reflat) \end{cases}$$

La rotation que nous avons codée est issue de ce dernier système d'équations. Elle prend en entrées les réels (en double précision) : lon, lat, $reflon$ et $reflat$; et elle retourne les réels (en double précision) : la nouvelle longitude λ et la nouvelle latitude ϕ.
Nous choisirons $reflon = 0$ et $reflat = 90$. Ainsi, toutes les mailles sources et cibles se trouvant près du pôle seront ramenées proche de l'équateur.

3.2.2 Les étapes de l'interpolation conservative avec rotation

Nous avons inséré la rotation dans le code de l'interpolation conservative. L'idée est d'utiliser les résultats des calculs avec rotation pour toutes les mailles cibles se trouvant à cheval ou au-dessus d'une latitude donnée lat2. Nous avons défini deux valeurs de latitudes $lat1 < lat2$ pour procéder à la rotation de toutes les mailles sources se trouvant au-dessus de $lat1$ et des mailles cibles se trouvant à cheval ou au-dessus de $lat2$. Il faut dès lors choisir $lat1$ de façon à faire tourner touytes les mailles soures qui intersectent potentiellement les mailles cibles à cheval ou au-dessus de $lat2$ même si ces mailles sources ont elles-même une latitude inférieure à $lat2$. Toutes les mailles non-tournées subiront le même traitement que celui expliqué dans le paragraphe 2.2.2. Nous pouvons présenter les différentes étapes du code avec la rotation de la façon suivante :

```
1°)Premier balayage sur toutes les mailles sources src
   Si max(lat(src)) > lat1 alors enregister la maille src  (A)
   Recherche des intersections avec des mailles cibles tgt
   Si intersection trouvée avec tgt alors
     Si max(lat(tgt)) =< lat2 alors calcul complet sans rotation
   Fin si
2°)Deuxième balayage sur toutes les mailles cibles tgt
   Si max(lat(tgt))>lat2 alors
     Enregistrer la maille tgt et passer à la suivante (B)
   sinon
     calcul complet sans rotation
   Fin si
3°)Appliquer la rotation à toutes les mailles enregistrées en (A) et (B)
4°)Troisième balayage sur toutes les mailles sources tournées
   enregistrées en (A)
   Calcul complet avec les mailles de (A) tournées
5°)Quatrième balayage sur toutes les mailles cibles tournées
   enregistrees en (B)
   Calcul complet avec les mailles de (B) tournées
6°)Combinaison des poids calculés avec et sans rotation
```

** après le premier balayage, seront stockées grâce à (A) toutes les mailles sources telles que max(lat(maille cible)>lat1. Ainsi, toutes les mailles sources qui s'intersectent avec une maille cible se trouvant à cheval ou au-delà de lat2 subiront une rotation.
** après le deuxième balayage seront stockées grâce à (C) toutes les mailles cibles situées au-delà ou à cheval sur lat2 (i.e max(lat(maille cible)>lat2). Ainsi, toutes les mailles cibles se trouvant à cheval ou au-delà de lat2 subiront une rotation.

3.3 Des résultats satisfaisants pour les mailles théoriques

Nous avons effectué les tests de cette nouvelle méthode d'interpolation conservative sur les mailles théoriques de 4°, 2°, 1°, 0.5° et 0.25° en plaçant le centre (A) du système de mailles comme expliqué dans la section 3.1.1. Ensuite, nous avons comparé les erreurs des poids entre les mailles sources et cibles à la valeur attendue de 0.25. Nous avons fait de même avec la méthode de la SCRIP originale avec et sans projection. Puis, nous avons représenté sur des graphes la fonction «log(erreur)» en fonction de la distance du point (A) au pôle. Rappelons que le north_thresh pour les cas avec projection a été choisi de telle façon que toutes les mailles soient projetées.

Les graphiques 3.7, 3.8, 3.9, 3.10 et 3.11 rapportent les résultats respectivement pour les mailles de 4°, 2°, 1°, 0.5° et 0.25°.

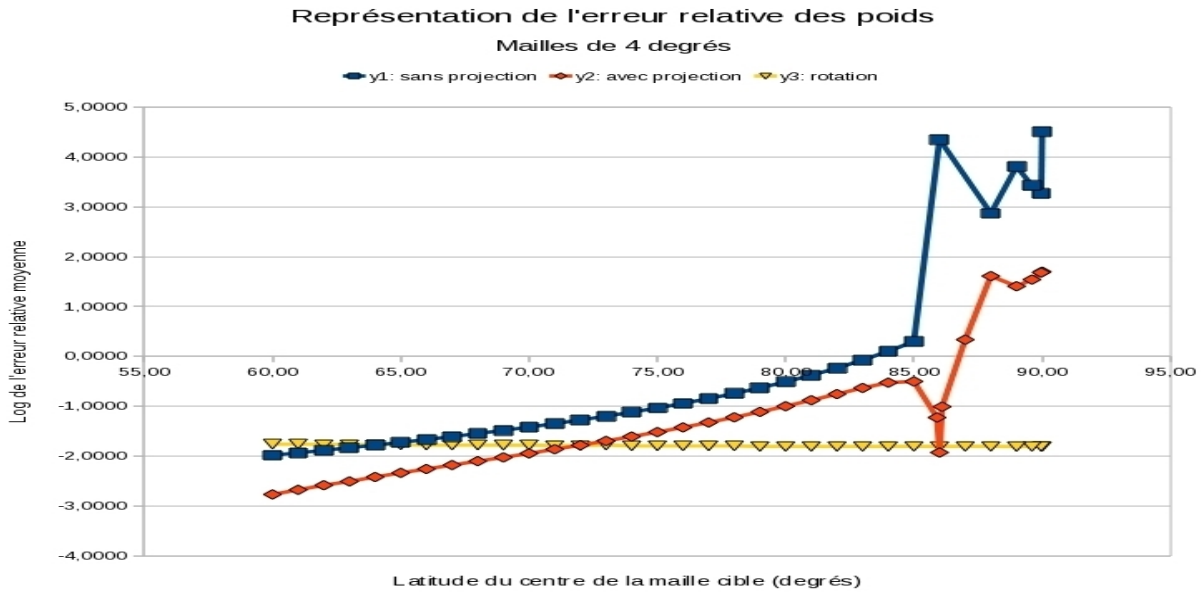


FIG. 3.7 – Log(erreur) en fonction de la distance de (A) au pôle nord, mailles de 4°.
Méthode originale sans projection, avec projection et notre nouvelle méthode avec rotation.

Nous constatons sur le graphique 3.7 que le log(erreur) reste constant et bas pour le cas de la rotation, qu'il diminue progressivement quand le point (A) s'éloigne du pôle nord pour les cas avec projection et sans projection. Aussi, tant que ces mailles de 4° se trouvent au-dessus de 73° environ, la méthode la plus précise est celle avec la rotation. Le cas avec projection se révèle être meilleur que le cas sans projection, quelque soit la latitude des mailles. Aussi, les méthodes originales avec et sans projection se montrent plus précise lorsque la latitude des mailles passe en-dessous de 65°, pour le cas sans projection et 73° pour celui avec projection.

Pour ces cinq figures 3.7, 3.8, 3.9, 3.10 et 3.11, nous constatons que les valeurs calculées par la SCRIP, sans projection, sont abhérentes pour les mailles dont le centre est relativement proche du pôle (inférieur à la taille de la maille), avec des valeurs du log(erreur) atteignant des valeurs supérieures à 3. Pour le

cas de la projection, l'erreur reste élevée lorsque les mailles se trouvent dans cette zone proche du pôle, mais moins importante que lors du cas précédent.

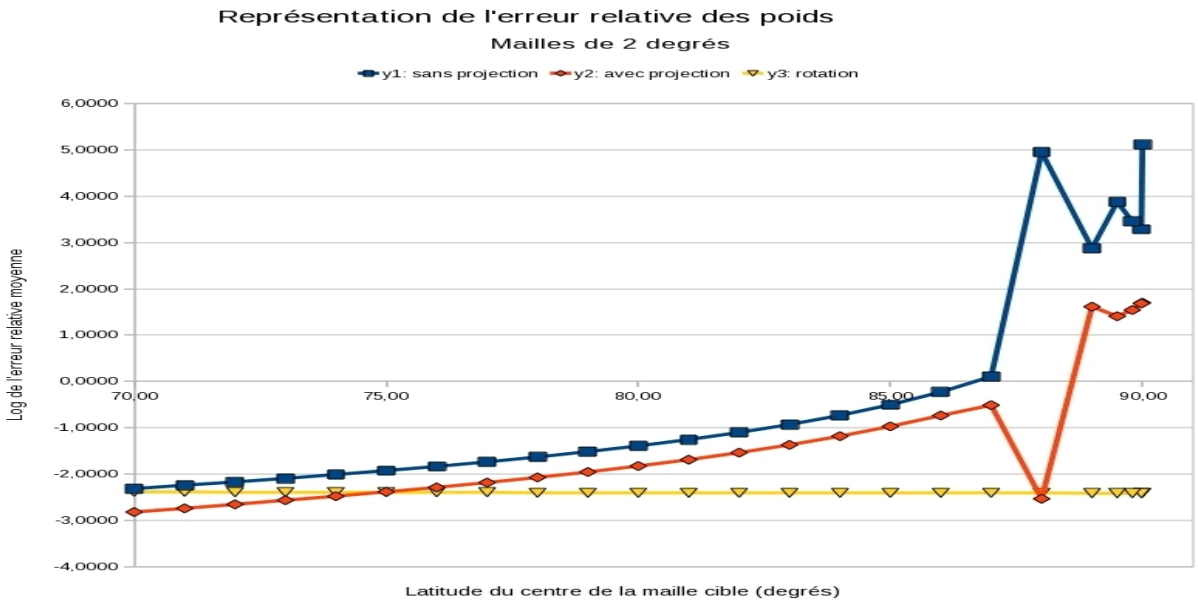


FIG. 3.8 – Log(erreur) en fonction de la distance de (A) au pôle nord, mailles de 2°.

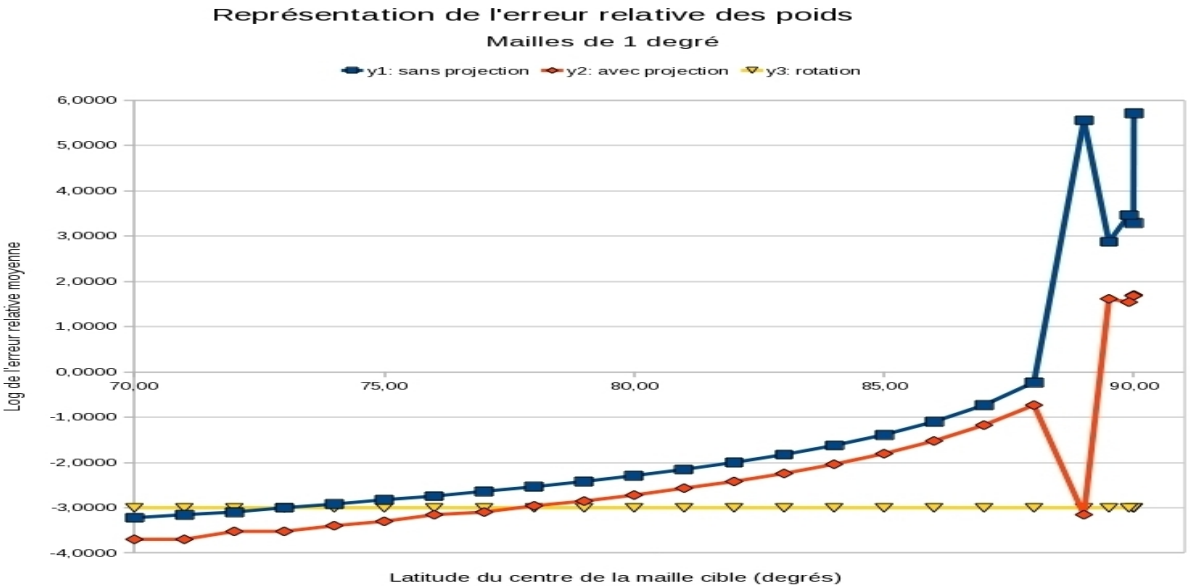


FIG. 3.9 – Log(erreur) en fonction de la distance de (A) au pôle nord, mailles de 1°.

Pour les mailles de 2° et 1° dont les résultats sont représentés sur les figures 3.8 et 3.9, nous constatons les mêmes faits que pour les mailles de 4° (figure 3.7). A la différence près que les cas avec projection et sans projection deviennent meilleurs plus rapidement que pour les mailles de 4°.
Pour les mailles de 2°, le seuil est de 70° sans projection, et 75° avec projection.

Pour les mailles de 1°, le seuil est de 73° sans projection, et 78° avec projection.

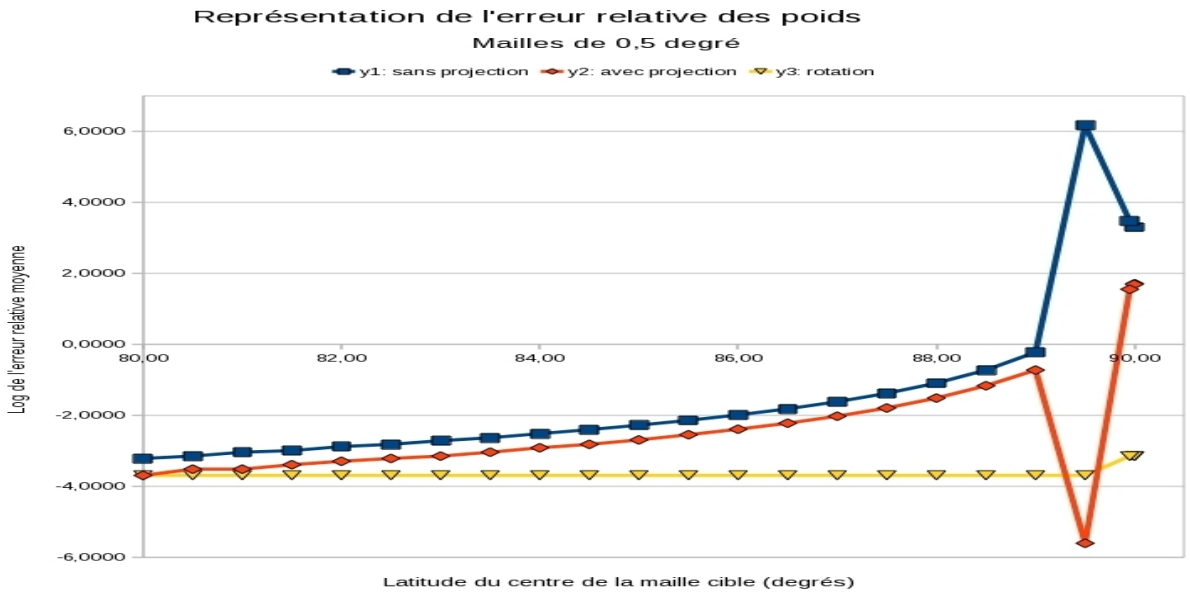


FIG. 3.10 – Log(erreur) en fonction de la distance de (A) au pôle nord, mailles de 0.5°.

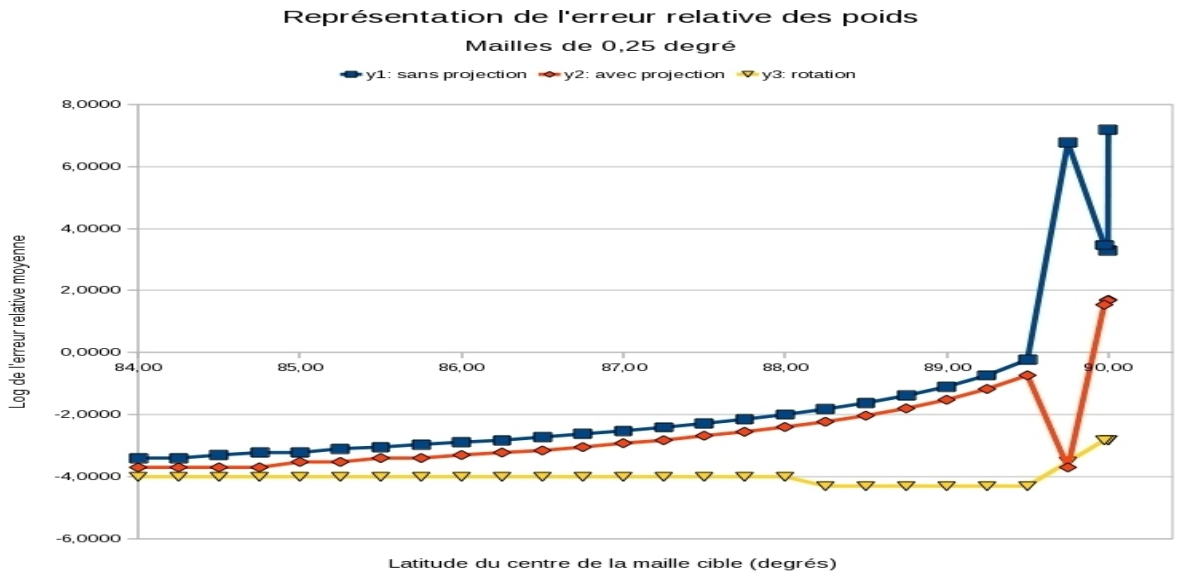


FIG. 3.11 – Log(erreur) en fonction de la distance de (A) au pôle nord, mailles de 0.25°.

Pour les mailles de 0.5° et 0.25° dont les résultats sont représentés sur les figures 3.10 et 3.11, nous faisons les mêmes constats que pour les mailles précédentes.

Sur ces cinq graphes, nous observons donc la même tendance, indépendamment de la taille des mailles. Le code avec rotation se comporte mieux que les deux autres, quand il s'agit du traitement des mailles proches du pôle. En effet, plus le centre du système de mailles (A) s'éloigne du pôle et plus les méthodes avec et sans projection tendent à se rapprocher du cas avec rotation. Ainsi, en-dessous d'une certaine latitude, les méthodes avec et sans projection deviennent meilleures que le cas de la rotation. Cette latitude limite varie selon la taille des mailles. Plus les mailles sont fines et plus cette limite est haute. En effet, à ces latitudes, l'erreur de la méthode originale avec et sans projection est faible et inférieure à l'imprécision supplémentaire apportée par le fait de devoir effectuer une opération additionnelle dans la méthode avec rotation.

La raison du pic minimum d'erreur observé sur les cinq figures, pour le cas de la projection, lorsque (A) se trouve exactement à la distance de la longueur de mailles du pôle n'a pas encore été élucidée. La particularité de ces mailles est que la maille source dont le coin est le plus au nord est confondu avec le pôle nord et que ce coin a pour longitude 10°, comme celle du point (A).

Nous avons montré dans cette section que notre méthode d'interpolation conservative avec rotation se comporte bien les mailles théoriques que j'ai créées et utilisées pour les tests. Une particularité de cette méthode est qu'elle ramène les mailles proches du pôle vers l'équateur (avec une rotation où $reflon$ est quelconque et $reflat = 90$) et que c'est justement proche de l'équateur que l'approximation de linéarité des bords de mailles en (lat,lon) est la plus valable. Pour les mailles proches du pôle, c'est-à-dire se trouvant à une distance inférieure à 10 fois la taille des mailles, la méthode que nous avons implémentée est donc plus juste dans les calculs que la méthode originale avec ou sans projection utilisée jusqu'à maintenant.

3.4 Des résultats satisfaisants pour les grilles issues de modèles réels

Afin d'appuyer les résultats obtenus dans la partie précédente, analysons comment réagit ce nouveau code d'interpolation conservative sur les grilles sur lesquelles nous avions noté des erreurs lors de l'exécution de la méthode sans projection et avec projection. Nous fixons dans les tests suivants les valeurs suivante de **lat1 = 1.1 radians et lat2 = 1.3 radians**.

Nous observons une nette amélioration des erreurs sur chacune des quatre figure 3.12, 3.13, 3.14 et 3.15, comparativement aux méthodes avec projection et sans projection visualisées entre les pages 15 et 19.

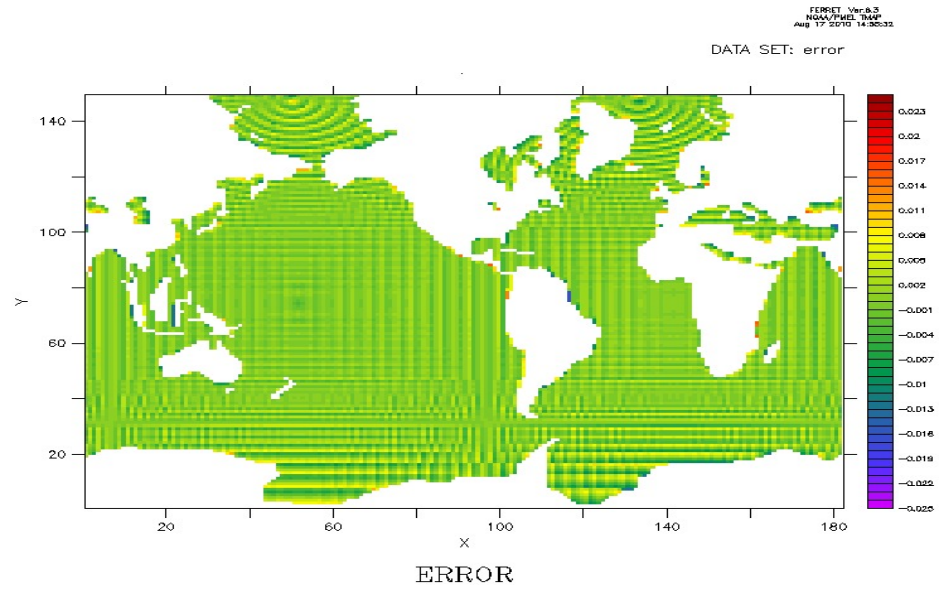


FIG. 3.12 – Erreur d'interpolation, avec rotation, de bt42 sur torc.

Sur la figure 3.12 qui représente l'erreur relative d'interpolation de bt42 sur torc avec la méthode incluant la rotation, nous constatons une erreur maximale près du pôle nord de 0.93%.
Celle-ci était de 1.4% pour la SCRIP originale sans projection (figure 2.5) et de 5% pour la SCRIP originale avec projection (figure 2.6).

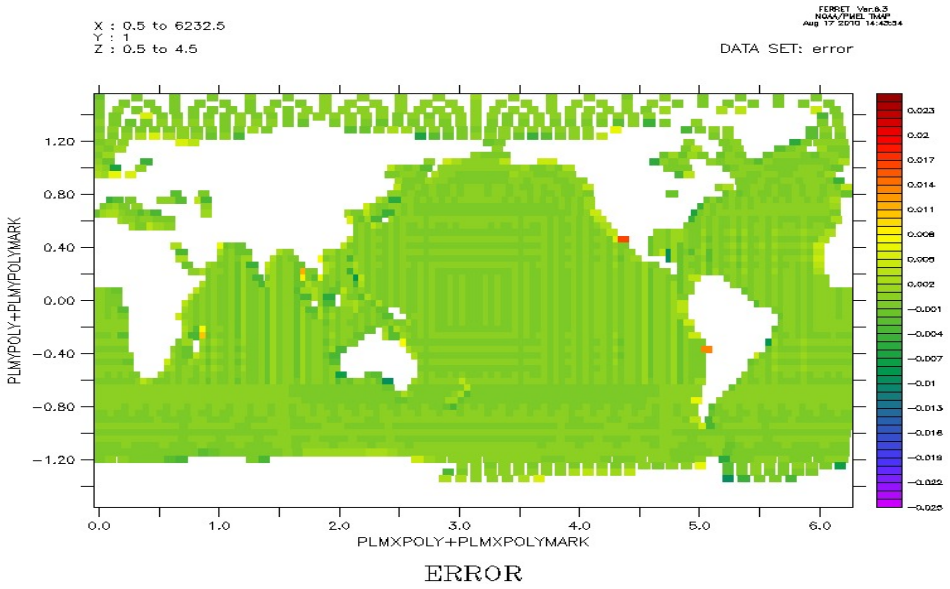


FIG. 3.13 – Erreur d'interpolation, avec rotation, de torc sur bt42.

Sur la figure 3.13, nous constatons que l'erreur relative maximale d'interpolation de torc sur bt42 avec la méthode incluant la rotation, est de 0.6% pour les mailles proches du pôle nord.
Celle-ci était de -1.8% pour la SCRIP originale sans projection (figure 2.7) et de -2.7% pour la SCRIP originale avec projection (figure 2.8).

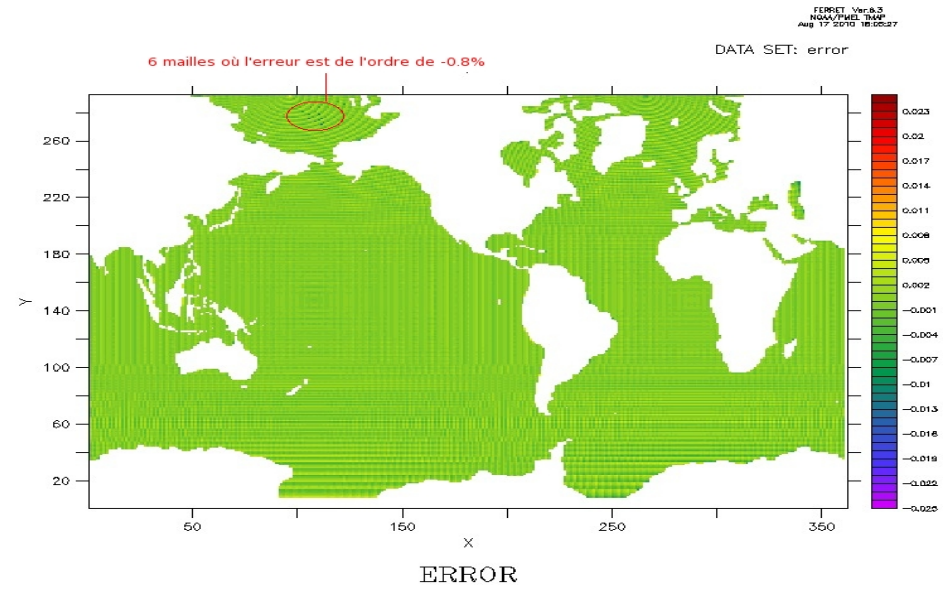


FIG. 3.14 – Erreur d'interpolation, avec rotation, de t127 sur tne1.

La figure 3.14 montre que l'erreur relative maximale d'interpolation de t127 sur tne1 avec la méthode incluant la rotation, est de -1% pour les mailles près du pôle nord.
Celle-ci était de 0.7% pour la SCRIP originale sans projection (figure 2.9) et de 25% pour la SCRIP originale avec projection (figure 2.10). Nous ne constatons donc pas d'amélioration notable par rapport à la SCRIP originale sans projection mais cette interpolation était celle où les calculs originaux sans projection donnaient justement des résultats corrects.
Notons que l'erreur de -1% apparait sur l'une des 6 mailles entourées de la figure 3.14. Nous n'avons pas eu le temps de le faire, mais la raison de cette erreur mérite être étudiée en détail.

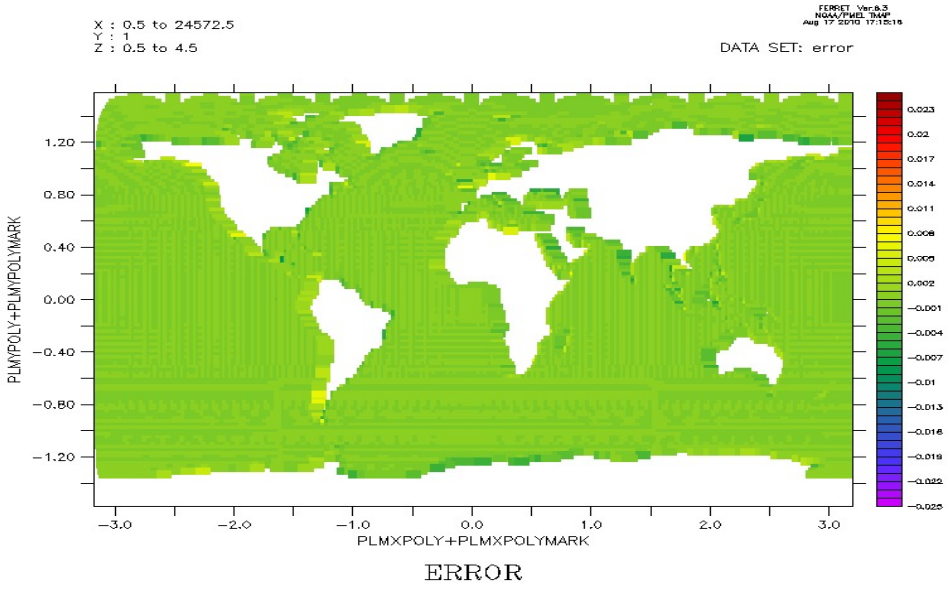


FIG. 3.15 – Erreur d'interpolation, avec rotation, de tne1 sur t127.

Sur la figure 3.15 qui représente l'erreur relative d'interpolation de tne1 sur t127 avec la méthode incluant la rotation, nous constatons une erreur maximale près du pôle nord de 0.1%.
Celle-ci était de 0.94% pour la SCRIP originale sans projection (figure 2.11) et de 6.5% pour la SCRIP originale avec projection (figure 2.12).

La méthode d'interpolation conservative qui inclue la rotation des mailles se trouvant proches du pôle se révèle donc être utile pour corriger toutes les erreurs que nous avions notées dans la partie 2 du rapport.

4 CONCLUSIONS ET PERSPECTIVES

4.1 Conclusions

Le but de mon stage ingénieur était d'apporter des solutions afin d'améliorer l'algorithme d'interpolation 2D conservatif dans le coupleur OASIS. Dans cet algorithme, la contribution d'une maille de la grille source dans le calcul de la valeur d'une maille cible dépend de la surface d'intersection des deux mailles. L'algorithme initial reposait sur des hypothèses qui deviennent de moins en moins valides à mesure que l'on se rapproche des pôles (Nord ou Sud).

Dans un premier temps, nous nous sommes intéressés à l'architecture du coupleur afin de mieux le connaitre. Puis, nous nous sommes familiarisés avec les environnements de tests qui avaient déja été implémentés. Après avoir testés la méthode d'interpolation conservative initiale sur des cas typiques, nous avons mis en place des outils afin de déterminer précisément les causes des erreurs constatées pour certaines mailles près des pôles. La méthode initiale fournissait aussi la possibilité d'activer une option permettant de projeter les mailles se trouvant au-delà d'une certaine latitude.

Nos différentes études nous ont montré que nous étions en présence d'un certain nombre de conflits. D'une part, lors de l'utilisation de la méthode en maintenant inactive l'option de projection, nous constatons que certaines mailles proche du pôle présentent des erreurs relatives considérablement élevées dues au fait que les hypothèse de linéarité des bords de mailles en longitude latitude sur lesquelles reposait l'algorithme deviennent de moins en moins valide à mesure que l'on s'approche des pôles.

D'autre part, lorsqu'on active l'option de projection, nous constatons que certaines mailles présentant une anomalie avec la méthode sans projection sont corrigées alors que de nouvelles mailles pathogènes apparaissent. Donc, non seulement la projection n'améliore donc pas toujours l'erreur des mailles du pôle, mais même lorsque de nouvelles mailles à problème n'apparaissent pas au pôle, nous sommes confrontés à un problème de raccordement des mailles se trouvant sur la frontière séparant les mailles projetées et celles non-projetées.

Afin de faire face à toutes ces causes d'erreurs, nous avons mis en place un algorithme où nous effectuons une rotation sur les mailles sources et cibles se trouvant au-delà d'une certaine latitude. Cette rotation ramène les mailles se trouvant au pôle nord vers l'équateur. Ainsi, puisque les hypothèses sur lesquelles reposait l'algorithme initial étaient plus valides vers l'équateur, les calculs que nous effectuons

sur ces mailles ramenées proche de l'équateur se révèlent être beaucoup plus précises que les méthodes avec et sans projection proposées initialement par la SCRIP.

Les tests effectués sur des mailles tests que nous avons créées montrent la précision de notre nouvelle façon de faire. Aussi, les cas tests reposant sur des grilles de vrais modèles couplés sur lesquelles nous avions constaté les erreurs sont corrigés par cette même nouvelle méthode.

Ce projet a évidemment une continuité. La prochaine étape de ce travail est de penser à des techniques permettant de rendre l'algorithme plus efficace lors de la recherche d'intersections entre les bords des mailles tournées. Une façon de faire inspirée de celle des «bins» de l'algorithme initiale est une piste à explorer.

4.2 Autres propositions de solution

4.2.1 Le calcul complet dans le repère projeté

Une solution à laquelle nous avons pensé, autre que la rotation, est celle de reprendre le code original de l'interpolation conservative et d'effectuer tous les calculs (intersection et intégrale) des mailles proches du pôle dans le repère projeté. Plutôt que de chercher une intersection dans le repère projeté et de revenir au repère initial pour calculer les intégrales, nous avons pensé qu'il pourait être plus judicieux de procéder à tous les calculs dans la zone projetée, pour les mailles se trouvant au-dessus d'une certaine latitude. Ainsi, nous pourons espérer s'affranchir des erreurs visualisées sur les figures 2.8 ou encore 2.12. Aussi, nous pourions utiliser la même idée que pour la rotation pour rejoindre les parties projetées et non projetées des grilles et s'affranchir des erreurs observées dans les figures 2.6 ou encore 2.10.

Cette double correction apportée au code original est une méthode à expérimenter : pour les mailles au-dessus d'une certaine latitude, effectuer les calculs d'intersections et d'intégrales de ligne dans le repère projeté et procéder à une bonne manière de combiner les mailles traitées avec et sans projection.

4.2.2 Projection locale dans un plan pour toutes les mailles sources et cibles

Une troisième solution à laquelle nous avons pensé est la projection locale de chacune des mailles des grilles source et cible. L'idée consiste à effectuer un balayage. Lors de celui-ci, on projette (en utilisant une projection conservant les surfaces) chacune des mailles cibles et les mailles sources qui sont susceptibles de s'intersecter avec elle dans un plan local tangent au centre de la maille cible. On procède à la totalité des calculs d'intersections et d'intégrales de ligne dans ce repère local, de telle sorte qu'une fois balayée, la maille cible soit totalement traitée. On calcule ainsi les poids. Puis on normalise à la fin selon l'option choisie. L'idée de cette méthode nous vient de l'Institut Pierre Simon Laplace (IPSL, laboratoire situé à Paris).

BILAN PERSONNEL

Mon stage de fin d'études réalisé au CERFACS a été une réussite pour moi. Les 5 mois que j'ai passé dans cet environnement agréable m'ont convaincu que le domaine de la recherche obéit à des lois qui lui sont propres. L'ambiance de travail, la disponibilité de ma responsable de stage, mais aussi celle des autres employés de cette société civile ont incontestablement été d'une grande aide dans la réussite de notre travail.

La culture de travail développée par l'équipe Global Change de ce site toulousain réunit les notions d'efficacité au travail, de moments de partage et de convivialité. D'autre part, les projets développés par cette équipe sont d'autant plus ambitieux que certains sont des projets européens de portée mondiale. De même deux équipes développent deux logiciels différents utilisés pour les prévisions climatiques. Ces développements font partis de projets européens et sont co-développés avec différentes équipes de différents pays européens.

Au sein du département Global Change, j'ai eu le plaisir de travailler à la résolution d'un problème qui était connu depuis des années et d'avoir ainsi contribué, à mon échelle, à un projet européen d'une grande importance visant à évaluer le changement climatique et ses impacts sur la société. J'ai eu aussi la satisfaction d'avoir pu discuter et proposer des idées pour la réussite de notre mission.

Ainsi, durant mon stage au CERFACS j'ai pu allier le travail à la formation, l'autonomie au travail en équipe et l'écriture de codes à la réflexion méthématico-philosophique concernant les notions de conservation d'énergie... Ces cinq mois de ma vie passés à Toulouse ont été plus que ceux d'un stage, je considère avoir effectué une véritable expérience professionnelle, tant par mon implication personnelle que par la volonté de ma responsable de stage de me faire profiter des moyens de l'entreprise.

Bibliographie - Webographie

VALCKE S., OASIS3 User's Guide,
document disponible depuis le lien http ://www.cerfacs.fr/3-25801-Technical-Reports.php

Site web d'OASIS, https ://oasistrac.cerfacs.fr/
Site web de IS-ENES, https ://is.enes.org/
Site web de SCRIP, http ://www.prism.enes.org/Publications/index.php
Site web de PALM, http ://www.cerfacs.fr/globc/PALM_WEB/

JONES P. W., A User's Guide for SCRIP : A Spherical Coordinate Remapping and Interpolation Package, Version 1.4, Theoretical Division, Los Alamos National Laboratory, chapter 3 : Conservative Remapping, p. 14 à 20, document disponible en annexe 2 de ce rapport et depuis le lien :
climate.lanl.gov/Software/SCRIP/SCRIPusers.pdf

JONES P. W., First- and Second-Order Conservative Remapping Schemes for Grids in Spherical Coordinates, Monthly Weather Review, 127, p. 2204-2210,
document disponible depuis le lien http ://journals.ametsoc.org/toc/mwre/127/9

CORDE P., Langage Fortran Support de cours, Idris,
document disponible depuis le lien http ://www.idris.fr/data/cours/lang/f90/choix_doc.html

CORDE P., Cours Fortran 95, version 10.0, Idris,
document disponible depuis le lien http ://www.idris.fr/data/cours/lang/f90/choix_doc.html

Zender C., NCO User's Guide - A suite of netCDF operators, Edition 4.0.3, for NCO Version 4.0.3, July 2010, Department of Earth System Science, University of California,
document disponible depuis le lien http ://nco.sourceforge.net

Annexes

.1 Annexe 1 : Exemple de fichier namcouple

```
# -*- Mode : ksh -*-
# This is a typical namcouple for OASIS3, illustrating, in mode NONE
#
# PREMIERE PARTIE
#############################################################
$SEQMODE
# This keyword concerns the coupling algorithm. Put here the maximum number
# of fields that have to be, at one particular coupling timestep,
# necessarily exchanged sequentially in a given order.
# In mode NONE, put always 1.
1
$END
#############################################################
$CHANNEL
# This describes the kind of message passing you want to use.
# Choices are NONE, MPI1 or MPI2.
NONE
$END
#############################################################
$NFIELDS
# In mode NONE, this is the total number of fields that will be interpolated.
1
$END
#############################################################
$JOBNAME
# This is just descriptive, it is an acronym for this given simulation
INT
$END
#############################################################
$NBMODEL
# This gives you the number of models running in this experiment +
# their names. In mode NONE, put always 0.
0
$END
#############################################################
$RUNTIME
# This gives you the total simulated time for this run in seconds.
# In mode NONE, put the number of time occurences to interpolate
# from the restart file.
1
$END
```

```
#####################################################################
$INIDATE
# This is the initial date of the run. This date is important only for the
# FILLING operation and for printing information in OASIS3 log file, cplout
00010101
$END
#####################################################################
$MODINFO
# Indicates if a header is encapsulated within the field brick
# for coupling field exchanges based on NONE, PIPE, SIPC and GMEM
# communication technique. (YES or NOT)
NOT
$END
#####################################################################
$NLOGPRT
# Index of printing level in output file cplout : 0 = no printing
# 1 = main routines and field names when treated, 2 = complete output
2
$END
#####################################################################
$CALTYPE
# Calendar type : 0 = 365 day calendar (no leap years)
# 1 = 365 day, or 366 days for leap years, calendar
# n (>1) = n day month calendar
# This is important only if FILLING analysis is used for a coupling
# field in the run.
1
$END
```

 # SECONDE PARTIE[1]

###

$STRINGS

###

field_in field_ou 1 1 3 fldin.nc fldou.nc EXPORTED

 # field_in : symbolic name for the field in the source model. It has to

 # match the argument name of the corresponding field declaration in the source model.

 # field_ou : symbolic name for the field in the target model.

 # 1 : index in auxiliary file cf_name_table.txt used by OASIS3 and PSMILe

 # to identify corresponding CF standard name and units

 # 1 : coupling and/or I/O period for the field, in seconds. If $CHANNEL is NONE, put "1".

 # 3 : number of transformations to be performed by OASIS3 on this field.

 # fldin.nc : created input field, fldou.nc : created output field

 # EXPORTED : exchanged between component models and transformed by OASIS3.

#

torc bt42

 # torc is the name of the input field, bt42 is the name of the output field

#

P 2 P 0

 # P : source grid first dimension characteristic ('P' : periodical ; 'R' : regional).

 # 2 : source grid first dimension number of overlapping grid points (2 for torc)

 # P : target grid first dimension characteristic ('P' : periodical ; 'R' : regional).

 # 0 : target grid first dimension number of overlapping grid points (0 for at42 - bt42 - lmdz)

#

CHECKIN SCRIPR CHECKOUT

 # CHECKIN : calculates the mean and extremum values of the source field (printed in cplout).

 # SCRIP : name of the used library.

 # CHECKOUT : calculates the mean and extremum values of an output field (printed in cplout).

#

INT=1

CONSERV LR SCALAR LATLON 10 FRACAREA FIRST

INT=1

 # CONSERV : (routine ../scrip/src/remap_conserv.F90) conservative interpolation

 # LR : (Logically Reduced) source grid type. Other options : D or U

 # SCALAR : field type

 # LATLON : the search restriction type (LATLON or LATITUDE)

 # 10 : the number of restriction bins

 # FRACAREA : normalisation option, FIRST : First order.

###

$END

[1]Les nombreuses options concernant la seconde partie de la namcouple sont détaillées dans le manuel d'utlisation OASIS3_UserGuide.pdf. Ici, il est seulement évoqué ce que nous avons utilisé lors de nos simulations.

.2 Annexe 2 : Méthode conservative

Méthode exposée dans le manuel d'utilisation de la SCRIP :

A User's Guide for SCRIP : A Spherical Coordinate
Remapping and Interpolation Package
Version 1.4
by Philip W. Jones

Chapter 3

Conservative Remapping

The SCRIP package implements a conservative remapping scheme described in detail in a separate paper (Jones, P.W. 1999 *Monthly Weather Review*, **127**, 2204-2210). A brief outline will be given here to aid the user in understanding what this portion of the package does.

To compute a flux on a new (destination) grid which results in the same energy or water exchange as a flux f on an old (source) grid, the destination flux F at a destination grid cell k must satisfy

$$\overline{F}_k = \frac{1}{A_k} \int \int_{A_k} f dA, \tag{3.1}$$

where $\overline{F}$ is the area-averaged flux and A_k is the area of cell k. Because the integral in (3.1) is over the area of the destination grid cell, only those cells on the source grid that are covered at least partly by the destination grid cell contribute to the value of the flux on the destination grid. If cell k overlaps N cells on the source grid, the remapping can be written as

$$\overline{F}_k = \frac{1}{A_k} \sum_{n=1}^{N} \int \int_{A_{nk}} f_n dA, \tag{3.2}$$

where A_{nk} is the area of the source grid cell n covered by the destination grid cell k, and f_n is the local value of the flux in the source grid cell (see Figure 3.1). Note that (3.2) is normalized by the destination area A_k corresponding to the normalize_opt value of 'destarea'. The sum of the weights for a destination cell k in this case would be between 0 and 1 and would be the area fraction if f_n were identically 1 everywhere on the source grid.

The normalization option 'fracarea' would actually divide by the area of the source grid overlapped by cell k:

$$\sum_{n=1}^{N} \int\int_{A_{nk}} dA. \tag{3.3}$$

For this normalization option, remapping a function f which is 1 everywhere on the source grid would result in a function F that is exactly one wherever the destination grid overlaps a non-masked source grid cell and zero otherwise. A normalization option of 'none' would result in the actual angular area participating in the remapping.

Assuming f_n is constant across a source grid cell, (3.2) would lead to the first-order area-weighted schemes used in current coupled models. A more accurate form of the remapping is obtained by using

$$f_n = \overline{f}_n + \nabla_n f \cdot (\vec{r} - \vec{r}_n), \tag{3.4}$$

where $\nabla_n f$ is the gradient of the flux in cell n and $\vec{r}_n$ is the centroid of cell n defined by

$$\vec{r}_n = \frac{1}{A_n} \int\int_{A_n} \vec{r} dA. \tag{3.5}$$

Such a distribution satisfies the conservation constraint and is equivalent to the first terms of a Taylor series expansion of f around $\vec{r}_n$. The remapping is thus second-order accurate if $\nabla_n f$ is at least a first-order approximation to the gradient.

The remapping can now be expanded in spherical coordinates as

$$\overline{F}_k = \sum_{n=1}^{N} \left[\overline{f}_n w_{1nk} + \left(\frac{\partial f}{\partial \theta} \right)_n w_{2nk} + \left(\frac{1}{\cos\theta} \frac{\partial f}{\partial \phi} \right)_n w_{3nk} \right], \tag{3.6}$$

where θ is latitude, ϕ is longitude and the three remapping weights are

$$w_{1nk} = \frac{1}{A_k} \int\int_{A_{nk}} dA, \tag{3.7}$$

$$\begin{aligned} w_{2nk} &= \frac{1}{A_k} \int\int_{A_{nk}} (\theta - \theta_n) dA \\ &= \frac{1}{A_k} \int\int_{A_{nk}} \theta dA - \frac{w_{1nk}}{A_n} \int\int_{A_n} \theta dA, \end{aligned} \tag{3.8}$$

and

$$\begin{aligned} w_{3nk} &= \frac{1}{A_k}\int\int_{A_{nk}} \cos\theta(\phi-\phi_n)dA \\ &= \frac{1}{A_k}\int\int_{A_{nk}} \phi\cos\theta dA - \frac{w_{1nk}}{A_n}\int\int_{A_n} \phi\cos\theta dA. \end{aligned} \tag{3.9}$$

Again, if the gradient is zero, (3.6) reduces to a first-order area-weighted remapping.

The area integrals in equations (3.7)–(3.9) are computed by converting the area integrals into line integrals using the divergence theorem. Computing line integrals around the overlap regions is much simpler; one simply integrates first around every grid cell on the source grid, keeping track of intersections with destination grid lines, and then one integrates around every grid cell on the destination grid in a similar manner. After the sweep of each grid, all overlap regions have been integrated.

Choosing appropriate functions for the divergence, the integrals in equations (3.7)–(3.9) become

$$\int\int_{A_{nk}} dA = \oint_{C_{nk}} -\sin\theta d\phi, \tag{3.10}$$

$$\int\int_{A_{nk}} \theta dA = \oint_{C_{nk}} [-\cos\theta - \theta\sin\theta]d\phi, \tag{3.11}$$

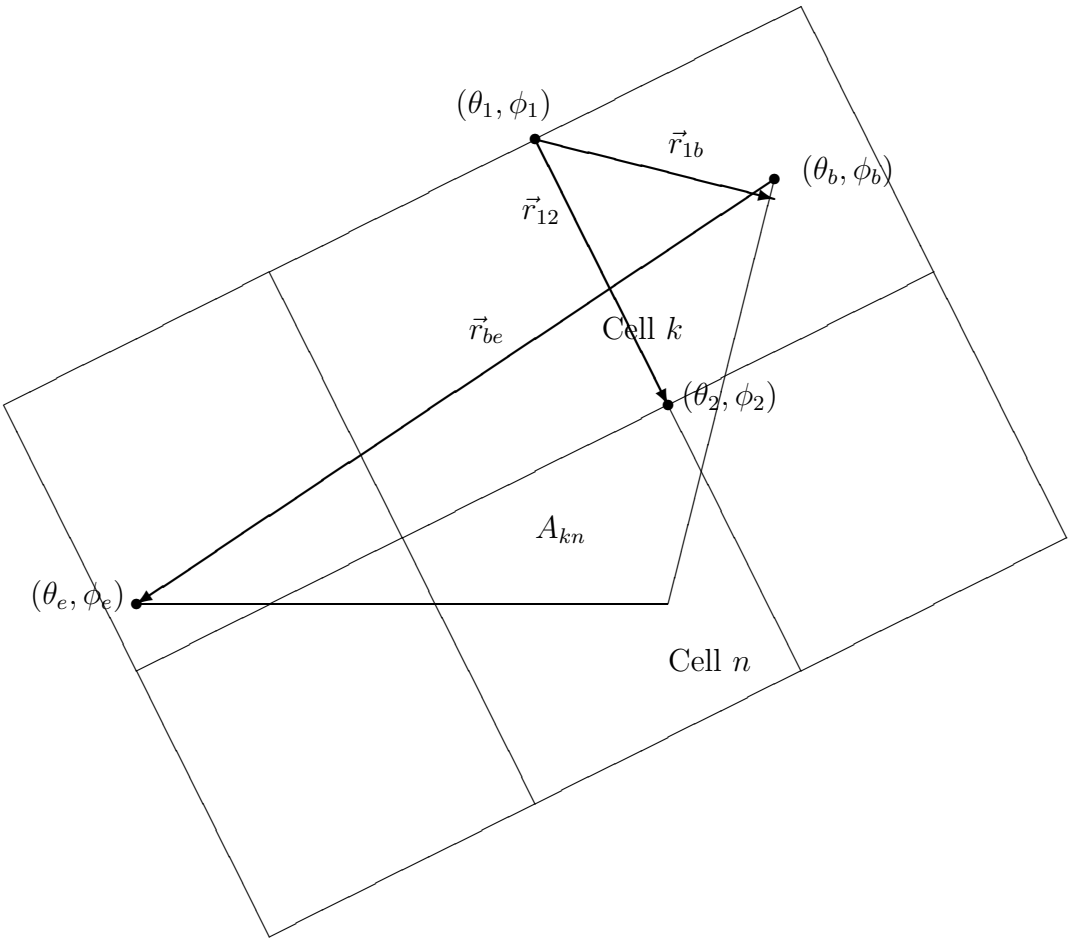
$$\int\int_{A_{nk}} \phi\cos\theta dA = \oint_{C_{nk}} -\frac{\phi}{2}[\sin\theta\cos\theta + \theta]d\phi, \tag{3.12}$$

where C_{nk} is the counterclockwise path around the region A_{nk}. Computing these three line integrals during the sweeps of each grid provides all the information necessary for computing the remapping weights.

3.1 Search algorithms

As mentioned in the previous section, the algorithm for computing the remapping weights is relatively simple. The process amounts to finding the location of the endpoint of a segment and then finding the next intersection with the other grid. The line integrals are then computed and summed according to which grid cells are associated with that particular subsegment. The most time-consuming portion of the algorithm is finding which cell on one grid

Figure 3.1: An example of a triangular destination grid cell k overlapping a quadrilateral source grid. The region A_{kn} is where cell k overlaps the quadrilateral cell n. Vectors used by search and intersection routines are also labelled.



contains an endpoint from the other grid. Optimal search algorithms can be written when the grid is well structured and regular. However, if one requires a search algorithm that will work for any general grid, a hierarchy of search algorithms appears to work best. In SCRIP, each grid cell address is assigned to one or more latitude bins. When the search begins, only those cells belonging to the same latitude bin as the search point are used. The second stage checks the bounding box of each grid cell in the latitude bin. The bounding box is formed by the cells minimum and maximum latitude and longitude. This process further restricts the search to a small number of cells.

Once the search has been restricted, a robust algorithm that works for most cases is a cross-product test. In this test, a cross product is computed between the vector corresponding to a cell side ($\vec{r}_{12}$ in Figure 3.1) and a vector extending from the beginning of the cell side to the search point ($\vec{r}_{1b}$). If

$$\vec{r}_{12} \times \vec{r}_{1b} > 0, \tag{3.13}$$

the point lies to the left of the cell side. If (3.13) holds for every cell side, the point is enclosed by the cell. This test is not completely robust and will fail for grid cells that are non-convex.

3.2 Intersections

Once the location of an initial endpoint is found, it is necessary to check to see if the segment intersects with the cell side. If the segment is parametrized as

$$\begin{aligned} \theta &= \theta_b + s_1(\theta_e - \theta_b) \\ \phi &= \phi_b + s_1(\phi_e - \phi_b) \end{aligned} \tag{3.14}$$

and the cell side as

$$\begin{aligned} \theta &= \theta_1 + s_2(\theta_2 - \theta_1) \\ \phi &= \phi_1 + s_2(\phi_2 - \phi_1), \end{aligned} \tag{3.15}$$

where $\theta_1, \phi_1, \theta_2, \phi_2, \theta_b$, and θ_e are endpoints as shown in Figure 3.1, the intersection of the two lines occurs when θ and ϕ are equal. The linear system

$$\begin{bmatrix} (\theta_e - \theta_b) & (\theta_1 - \theta_2) \\ (\phi_e - \phi_b) & (\phi_1 - \phi_2) \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} (\theta_1 - \theta_b) \\ (\phi_1 - \phi_b) \end{bmatrix} \tag{3.16}$$

is then solved to determine s_1 and s_2 at the intersection point. If s_1 and s_2 are between zero and one, an intersection occurs with that cell side.

It is important also to compute identical intersections during the sweeps of each grid. To ensure that this will occur, the entire line segment is used to compute intersections rather than using a previous or next intersection as an endpoint.

3.3 Coincidences

Often, pairs of grids will share common lines (e.g. the Equator). When this is the case, the method described above will double-count the contribution of these line segments. Coincidences can be detected when computing cross products for the search algorithm described above. If the cross product is zero in this case, the endpoint lies on the cell side. A second cross product between the line segment and the cell side can then be computed. If the second cross product is also zero, the lines are coincident. Once a coincidence has been detected, the contribution of the coincident segment can be computed during the first sweep and ignored during the second sweep.

3.4 Spherical coordinates

Some aspects of the spherical coordinate system introduce additional problems for the method described above. Longitude is multiple valued on one line on the sphere, and this branch cut may be chosen differently by different grids. Care must be taken when calculating intersections and line integrals to ensure that the proper longitude values are used. A simple method is to always check to make sure the longitude is in the same interval as the source grid cell center.

Another problem with computing weights in spherical coordinates is the treatment of the pole. First, note that although the pole is physically a point, it is a line in latitude-longitude space and has a nonzero contribution to the weight integrals. If a grid does not contain the pole explicitly as a grid vertex, the pole contribution must be added to the appropriate cells. The pole contribution can be computed analytically.

The pole also creates problems for the search and intersection algorithms described above. For example, a grid cell that overlaps the pole can result

in a nonconvex cell in latitude-longitude coordinates. The cross-product test described above will fail in this case. In addition, segments near the pole typically exhibit large changes in longitude even for very short segments. In such a case, the linear parametrizations used above result in inaccuracies for determining the correct intersections.

To avoid these problems, a coordinate transformation can be used poleward of a given threshold latitude (typically within one degree of the pole). A possible transformation is the Lambert equivalent azimuthal projection

$$\begin{aligned} X &= 2\sin\left(\frac{\pi}{4}-\frac{\theta}{2}\right)\cos\phi \\ Y &= 2\sin\left(\frac{\pi}{4}-\frac{\theta}{2}\right)\sin\phi \end{aligned} \tag{3.17}$$

for the North Pole. The transformation for the South Pole is similar. This transformation is only used to compute intersections; line integrals are still computed in latitude-longitude coordinates. Because intersections computed in the transformed coordinates can be different from those computed in latitude-longitude coordinates, line segments which cross the latitude threshold must be treated carefully. To compute the intersections consistently for such a segment, intersections with the threshold latitude are detected and used as a normal grid intersection to provide a clean break between the two coordinate systems.

3.5 Conclusion

The implementation in the SCRIP code follows closely the description above. The user should be able to follow and understand the process based on this description.

.3 Annexe 3 : Rôle des «bins»

La restriction de domaine intervient dans le but de diminuer le nombre de mailles de la grille cible à prendre en compte dans le calcul de l'intégrale de contour pour chaque maille de la grille source, et vice-versa. Les fichiers misent en œuvre dans la réalisation de cette restriction de domaine d'intégration sont :

- oasis3/lib/scrip/src/remap_conserv.F
- oasis3/lib/scrip/src/grids.f

Dans un premier temps, dans le fichier *grids.f*, 2 tableaux de dimension (2, $num\_srch\_bins^2$) sont alloués : bin_addr1 pour la grille source et bin_addr2 pour la grille cible. La variable «num_srch_bins» a pour valeur celle nommée «$NBIN» initialisée dans la *namcouple*, «10» dans notre cas. Elle définie le nombre de longitudes et de latitudes qui sera dessiné dans les grilles source et cible. Nous serons alors en présence d'autant de longitudes que de latitudes.

Puis, les points d'intersections des longitudes et des latitudes sont numérotés de 1 à $num\_srch\_bins^2$, dans le sens trigonométrique (vu du Pôle Nord) en commençant par la longitude la plus à l'Ouest et la latitude la plus au Sud. Les mailles de la grille source seront numérotés de la même manière de 1 à grid1_size ainsi que celles de la grille cible de 1 à grid2_size.

Ensuite, pour la grille source, la colonne j du tableau bin_addr1 correspondra au numéro j dans la numérotation des intersections longitude-latitude. Ainsi, le coefficient (1,j) de bin_addr1 contiendra le plus petit numéro de la maille (appelé l'adresse) qui intersecte, même partiellement, le domaine délimité par les longitudes et les latitudes les plus proches qui se trouvent dans la partie inférieure-gauche de j. Si aucune maille ne se trouve là, bin_addr1(1,j) sera laissé à «grid1_size + 1». De façon similaire, le coefficient (2,j) de bin_addr1 contiendra le plus grand numéro de la maille (l'adresse) qui intersecte, même partiellement, le domaine décrit ci-dessus. Si aucune maille ne se trouve là, bin_addr1(2,j) sera laissé à «0».
Les coefficients du tableau bin_addr2 sont remplis de façon analogue en considérant cette fois les mailles de la grille cible.

Ces opérations se font lors de l'exécution du fichier *grids.f*, puis, les tableaux bin_addr1 et bin_addr2 sont récupérés et utilisés par le fichier *remap_conserv.F*, le fichier où est implémenté le code de l'interpolation conservative. Dans ce dernier, on initialise les variables « min_add » et « max_add » qui contiennent les adresses des mailles de la grille cible entre lesquelles le calcul de l'intégrale sera fait.

Pour chaque maille grid1_add de la grille source, on parcourt le tableau bin_addr1. Pour chaque colonne j où bin_addr1(1,j) <= grid1_add <= bin_addr1(2,j), on affecte à min_add la valeur minimale entre min_add et bin_addr2(1,j), et à max_add la valeur maximale entre max_add et bin_addr2(2,j). Cette opération permet d'enregistrer dans min_add l'adresse de la maille cible dont le numéro est le plus petit et qui a une intersection non vide avec la maille grid1_add. Aussi, dans max_add elle enregistre l'adresse de la maille cible dont le numéro est le plus grand et qui a une intersection non vide avec la maille grid1_add. Ainsi, lors du calcul de l'intégrale de contour, au lieu de faire la recherche d'intersection

entre grid1_add et toutes les mailles de la grille cible, on limite le domaine de recherche aux mailles cible qui se trouvent entre min_add et max_add. Les lignes de codes qui permettent d'obtenir ce résultat sont les suivantes :

```
   !***
   !*** restrict searches first using search bins
   !***
  min_add = grid2_size
  max_add = 1
  do n=1,num_srch_bins
   if (grid1_add>=bin_addr1(1,n) .and. grid1_add<=bin_addr1(2,n)) then
     min_add = min(min_add, bin_addr2(1,n))
     max_add = max(max_add, bin_addr2(2,n))
   endif
  end do
```

Une fois ces opérations effectuées, une deuxième restriction est programmée. Il s'agit de définir un domaine ne contenant que les mailles cibles qui ont effectivement une intersection avec la maille source grid1_add. Ainsi, on limite une fois de plus le domaine de recherche des mailles cibles.

```
   !***
   !*** further restrict searches using bounding boxes
   !***
  num_srch_cells = 0
  do grid2_add = min_add,max_add
    srch_mask(grid2_add) =
&     (grid2_bound_box(1,grid2_add) <= grid1_bound_box(2,grid1_add)) .and.
&     (grid2_bound_box(2,grid2_add) >= grid1_bound_box(1,grid1_add)) .and.
&     (grid2_bound_box(3,grid2_add) <= grid1_bound_box(4,grid1_add)) .and.
&     (grid2_bound_box(4,grid2_add) >= grid1_bound_box(3,grid1_add))
    if (srch_mask(grid2_add)) num_srch_cells = num_srch_cells+1
  end do
```

Ces tableaux et restrictions permettent de réduire le domaine de recherche des mailles cibles. Sans eux, pour chaque maille source, on aurait parcourus la grille toute entière, alors qu'avec eux, la recherche se limite le plus souvent à un nombre de mailles compris entre 3 et 7. Évidemment, les restrictions analogues sont faites aussi pour le SWEEP 2. Ils sont même utilisés dans les autres formes d'interpolation : distwgt, gauswgt, bicubic et bilinear (mais codés différemment).