CASCADE :

LOGICIEL DE TRAITEMENT DES DONNEES D'ADCP DE COQUE
Version 3.0
Documentation maintenance - utilisateur

Auteurs : C. Kermabon, F. Gaillard - LPO

Dernière mise à jour : 12 Septembre 2002

Rapport interne DRO/LPO : 02-03

Ifremer

Laboratoire de Physique des Océans

Sommaire

1.Historique du logiciel

Version	Auteur	Date	Historique
0.9	C. Kermabon F. Gaillard	Septembre 1999	Version préliminaire du logiciel complet adapté au NB 75 Thalassa.
1.0	C. Kermabon	Février 2000	Adaptation au BB 150 Thalassa Modifications de format Version « POMME »
1.1	F. Gaillard	Mars 2000	Adaptation au BB 150 Suroît Mise à jour de la description des algorithmes de l'étape 2
2.0	C. Kermabon	Octobre 2000	Modification de l'accès aux variables NetCDF (suppression des 'getcdf_batch') Ajout de variables dans les fichiers NetCDF Simplification de l'interface utilisateur Création fichier info_navire Modification de ***ad2_ensmoy.m***
	F. Gaillard	Novembre 2000	Modification de l'estimation de l'amplitude, du désalignement et de l'assiette du navire (***ad2_headestim.m***)
	C. Kermabon	Janvier 2001	Modification de ***ad2_Beam2ADCP_0.m*** Modification de ***net_vit_cour.m*** (résolution d'un bug)
	C. Kermabon	Mars 2001	Modification de ***net_vit_cour.m*** et ***Net_vitesse.m***
	C. Kermabon	Mai 2001	Mise en place de la modification apportée par T. Terre sur les scripts ***jul_0h.m*** et ***greg_0h.m***
3.0	C. Kermabon	Mars 2002	Prise en compte du nouveau format des fichiers RDI navigation Prise en compte de la nouvelle convention roulis/tangage interne Modification pour traitement des fichiers RDI en majuscule
	P. Lherminier		Modification du nettoyage des données dans la partie 2 et la partie exploitation
	C. Kermabon		Modification des tracés en conséquence
	P. Lherminier	Juin 2002	Ajout de la date grégorienne dans les fichiers NetCDF. Modification de la partie exploitation (nettoyage des données et tracés de contourage)
	C. Kermabon	Juillet 2002	Modification de la partie exploitation (ajout

			de divers tracés et informations associés au fichier campagne – filtrage de ce fichier)

1.1Version 1.0

Cette version est installée dans le conteneur pour POMME 0.

Au mois de février 2000, le logiciel CASCADE, développé dans un premier temps pour traiter les données brutes ADCP NarrowBand 75 Khz issues du logiciel d'acquisition TRANSECT, a été modifié afin de pouvoir également traiter les données brutes ADCP BroadBand 150 Khz générées par TRANSECT. (C. Kermabon).
Pour ce faire,

◊ ont été créés les scripts matlab ci-dessous :
- ◊ ***ad1_brut2cdf_bb.m*** : Création de fichiers bruts au format NetCDF à partir de fichiers bruts binaires TRANSECT. Ce script fait appel à :
  - ◊ ***rdbbhead.m*** : Lecture de l'entête de chaque ensemble
  - ◊ ***rdbbflead.m*** : Lecture de la partie 'Fixed Leader' de chaque ensemble
  - ◊ ***rdbbvlead.m*** : Lecture de la partie 'Variable Leader' de chaque ensemble
  - ◊ ***rdbbvlead_fast.m*** : Idem que ***rdbbvlead.m*** si ce n'est qu'on ne lit que les informations à sauvegarder dans le fichier NetCDF résultat.
  - ◊ ***rdbb_bottom.m*** : Lecture de la partie 'Bottom ping' de chaque ensemble

Pour des informations complémentaires, l'utilisateur peut se référer à l'appendice D 'BBADCP Output Data Formats' de la documentation 'BBADCP Technical Manual'.
Le script ***ad1_brut2cdf.m*** a été renommé en ***ad1_brut2cdf_nb.m***.

◊ ont été modifiés les scripts ci-dessous :
- ◊ ***ad1_cree_nc.m*** : Selon le type d'ADCP (NarrowBand ou BroadBand), on fait appel au script ***ad1_brut2cdf_nb.m*** ou ***ad1_brut2cdf_bb.m***.
- ◊ ***ad1_cal_vraie_derive.m*** : Sur le navire THALASSA, selon le type d'ADCP (NarrowBand ou BroadBand), le roulis peut avoir un signe différent du roulis contenu dans les fichiers navigation NetCDF.
- ◊ ***ad1_corr_att.m*** : Sur le navire THALASSA, selon le type d'ADCP, le roulis peut avoir un signe différent du roulis contenu dans les fichiers navigation NetCDF.
- ◊ ***ad2_ensmoy.m*** : Sur le navire THALASSA, selon le type d'ADCP, le roulis peut avoir un signe différent du roulis contenu dans les fichiers navigation NetCDF. De plus, dans le cas de l'ADCP NarrowBand, le fichier NetCDF contient les données de vitesse exprimées en count. Il faut utiliser le terme '*dop2vit*' afin de transformer les vitesses de count en cm/s. Dans le cas d'un ADCP BroadBand, les vitesses sont déjà exprimées en cm/s. Ces vitesses sont calculées par l'ADCP à partir de la vitesse du son que l'ADCP estime (ou fixe à 1500m/s selon la configuration de l'ADCP). Il faut donc corriger cette vitesse en fonction de la vitesse du son réestimée par CASCADE d'où le terme *corr_vit_son*.

◊ a été modifié le fichier :
- ◊ info_adcp : Prise en compte de l'ADCP BroadBand 150 Khz.

1.2Version 1.1

A l'occasion de tests sur les données Suroît, quelques programmes ont été modifiés (F. Gaillard).

- pour l'étape 1:
  ad1_cal_vraie_dérive.m et ***ad1_cor_att.m*** pour tenir compte de l'absence d'attitude externe. Sur ce navire, un seul jeux de données attitude est disponible : HDMS, il est stocké dans la variable attitude GPS des fichiers attitude externe.

- pour l'étape 2:

ad2_ensmoy.m: Correction de l'algorithme de réaffectation des bins
ad2_headestim.m **:** Modification des critères de sélection des données prises en compte pour le calcul, graphiques supplémentaires.
Les variables **acc_min, acc_max** ne sont plus utilisées.

1.3Version 2.0

- Des panneaux de configuration distincts pour chaque étape ont été développés.
  Pour ce faire, ont été créés les scripts suivants :
  - ***info_camp.m :*** Interface permettant la saisie des informations campagne (nom de la campagne, nom du navire, type de l'ADCP (NB75, BB150, ...etc.))
  - ***info_etape1.m :*** Interface permettant la saisie des informations utiles à la bonne exécution de l'étape 1, à savoir : nom du répertoire des fichiers des données brutes ADCP RDI, racine de ces fichiers, nom du répertoire où seront stockés les fichiers NetCDF, nom du répertoire où seront stockés les fichiers temporaires. C'est via cette interface que l'utilisateur indique s'il dispose d'attitude externe. Si oui, il doit alors préciser le répertoire et la racine des fichier associés.
  - ***info_etape2.m :*** Interface permettant la 'saisie' des informations « répertoires et fichiers » utiles à la bonne exécution de l'étape 2, à savoir : nom du répertoire où seront stockés les fichiers NetCDF, nom du répertoire où seront stockés les fichiers temporaires, nom du fichier de navigation.
  - ***info_etape2_2.m :*** Interface permettant la saisie des constantes de traitement utiles à l'étape 2. Ces constantes de traitement sont explicitées dans le paragraphe 5.2.
  - ***info_etape3.m :*** Interface permettant la saisie des constantes de traitement utiles à l'étape 3. Ces constantes de traitement sont explicitées au paragraphe 4.3.4.
- L'accès des variables de fichier NetCDF s'effectuaient à partir de la subroutine 'getcdf_batch'. Pour accéder à différentes variables d'un même fichier, cette subroutine effectue à chaque fois une ouverture et fermeture de fichier. Pour y remédier et ainsi accélérer l'accès aux données NetCDF, cette subroutine a été remplacée.
- Selon les navires, les données d'attitude externe sont différentes. Elles peuvent être absentes, issues d'une MRU, d'un GPS-3D, ...etc. Afin que CASCADE fonctionne quelquesoit le navire, le fichier '***info_navire***' a été créé (cf. annexe).

  Ce fichier permet de :
  - mettre à jour la liste des navires et types d'ADCP disponibles utiles lors de la saisie des informations campagne (cf. ***info_camp.m***)

- mettre à jour la liste des attitudes externes disponibles utiles à l'étape 1.

Pour lire ce fichier, le script ***lit_info_adcp_navire.m*** a été créé.

- Dans l'étape 1,
  - ***ad1_derive_adcp.m*** a été modifié afin de prendre en compte le nouveau format du fichier navigation (*n.*) issu du logiciel d'acquisition RDI TRANSECT.
  - lors du calcul du polynôme (***ad1_cal_poly.m***), la possibilité d'éliminer manuellement des points a été rajoutée.
- Dans l'étape 2,
  - ***ad2_ensmoy.m*** a été modifié :
    - le calcul des profondeurs pour le BB150 a été modifié selon la formule :

      prof = blank +1/2(binsize + transmit_time + lag)
    - les données de Bottom-ping sont prises en compte.
  - ***ad2_Beam2ADCP_0.m*** a été modifié afin d'essayer de déterminer une solution à 3 faisceaux au cas où une solution à 4 faisceaux est impossible. En effet, dans le cas où aucun des faisceaux n'est en panne, il se peut qu'un faisceau ait aléatoirement une portée inférieure aux autres. Aussi, pour chaque bin de chaque profil, si une solution 4 faisceaux est déterminée, on la garde. Dans le cas contraire, on essaie d'y pallier par une solution à 3 faisceaux. Cette modification permet de récupérer des données en fin de profil.
  - ***ad2_headestim.m*** a été modifié :
    - le facteur d'amplitude est le rapport des modules des accélérations
    - l'erreur d'alignement est donné par l'écart d'angle entre l'accélération navire et l'accéleration horizontale mesurée par l'ADCP.
- Dans l'étape 3 (***ad3_absolu.m***), les données de bottom-ping sont prises en compte afin d'éliminer les données 'sous le fond'. La possibilité de calculer les vitesses absolues à partir des vitesses du navire issues du bottom-ping est également fournie. Dans ce cas, lorsque les données du bottom-ping sont absentes, ce sont les données de vitesse du navire issues du fichier de navigation qui sont considérées.
- Dans la partie exploitation, le nettoyage (***net_vit_cour.m, Net_vitesse.m***) des vitesses du fichier campagne a été modifié suite à une remarque de M. Assenbaum. Le critère de 'flagguage' des vitesses n'est plus basé sur la moyenne ('mean') mais sur la médiane ('median').
- En Mai 2001, les scripts ***jul_0h.m*** et ***greg_0h.m*** prennent en compte les modifications apportées par T. Terre. Auparavant, dans certains cas, ***greg_0h.m*** donnait un résultat comportant 60 secondes (Ex : 20h19min60sec au lieu de 20h20min0sec). Les modifications apportées y remédient. Néanmoins, il est à noter que, l'ensemble des étapes étant basé sur les jours juliens, ce 'bug' n'a aucune incidence sur le traitement. Les données VM-ADCP traitées via CASCADE avant Mai 2001 ne sont pas remis en cause.

1.4Version 3.0

En Mars 2002, CASCADE a été modifié afin de :

•1 Pouvoir traiter les fichiers RDI ayant un nom en majuscule (cf. ***info_camp.m, ad1_cree_nc.m***)

•2 Prendre en compte le nouveau format des fichiers navigation RDI. Ce fichier contient désormais un message ADCP temps réel comprenant :

- 3 L'heure bord
- 4 La navigation brute
- 5 La navigation intégrée
- 6 Le cap navire (gyrocompas)
- 7 Les données d'attitude :
    - 1 Cap
    - 2 Roulis
    - 3 Tangage
    - 4 Pilonnement

  Ces données sont issues soit de la centrale d'attitude HDMS, soit de la centrale d'attitude MRU, …etc. En principe, c'est la meilleure attitude externe possible.

  La convention de signe est la suivante :
    - 1 Le roulis est positif lorsque le navire gîte sur tribord (***convention inverse à CASCADE***)
    - 2 Le tangage est positif lorsque le navire lève le nez. ***(convention conforme à CASCADE)***

Lors de la lecture du fichier navigation RDI (pour l'estimation de la dérive de l'ADCP ***ad1_derive_adcp.m***), CASCADE crée un fichier NetCDF d'attitude comprenant :

- 1 L'heure bord
- 2 Le cap
- 3 Le roulis (avec le signe de la convention CASCADE)
- 4 Le tangage
- 5 La latitude issue de la navigation brute
- 6 La longitude issue de la navigation brute
- 7 La vitesse U du navire
- 8 La vitesse V du navire
- 9 L'accélération en X du navire
- 10 L'accélération en Y du navire

Il est à noter qu'un filtre de Butterworth de largeur 2*16s est appliqué sur les vitesses et accélérations navire. Ce filtre permet d'éliminer les données aberrantes ainsi que l'effet de la houle (d'une période de 15s environ). Cet effet est à supprimer car pour le calcul des vitesses absolues, on déduit les vitesses navire des vitesses ADCP sur des ensembles moyennés sur 1 à 2 minutes. Pour ces ensembles, l'effet de la houle ne subsiste donc plus.

D'autre part, désormais, un pré-traitement des données est possible à bord des navires à partir de la navigation de ce fichier. Ce fichier NetCDF d'attitude comprend la navigation brute et non la navigation intégrée car cette dernière comprend des filtres non adaptés au traitement ADCP. Le traitement final peut s'effectuer par la suite à partir d'une meilleure navigation (issue de TRINAV par exemple). (cf. ***Cdf_att.m, ad1_corr_att.m, ad2_adnav.m***)

- 1 Prendre en compte le fait que l'attitude interne à l'ADCP n'est plus dans le repère navire mais dans le repère ADCP (cf. ***ad1_cal_vraie_derive.m, ad1_atti_pl0.m***).

  Remarque : L'attitude interne n'intervient pas dans les calculs mais seulement au niveau du calage des horloges.
- 2 Nettoyer au mieux les données lors de la phase 2 et la phase d'exploitation.
    - 3 Le script ***ad2_calmoy.m*** a été modifié afin d'invalider les données dont la vitesse verticale est supérieure à une valeur fournie par l'utilisateur.
    - 4 Le script ***net_vit_cour.m*** a été modifié afin de :
        - o1 'flagguer' à 7 les données sous le fond (si l'utilisateur le désire).

- o2 ‘flagguer’ à 6 les vitesses absolues dont 1 composante horizontale est supérieure à 4m/s.
- o3 ‘flagguer’ à 5 les données absentes.
- o4 ‘flagguer’ à 4 les données dont la vitesse verticale ou l’écart-type de l’erreur est supérieure à une valeur fixée par l’utilisateur.
- o5 ‘flagguer’ à 3 les données dont l’une des composantes horizontales a un cisaillement vertical différentiel supérieur à une valeur fixée par l’utilisateur.
- o6 ‘flagguer’ à 0, 1 et 2 les données selon un critère de filtre médian.
- o7 ‘flagguer’ à 1 les singletons et doublons isolés.
- o8 ‘flagguer’ à 1 les profils dont moins de la moitié des données de la couche de référence est ‘flagguée’ à 0.

- •5 Auparavant, on ‘flagguait’ les profils entiers (et non les cellules). Les tracés étaient basés sur ce principe. Ces derniers ont été modifiés afin de prendre en compte que le ‘flagguage’ est désormais effectué par cellule.

En Juin 2002, CASCADE a été modifié ainsi :

- •1 Dans les fichiers NetCDF campagne, section et station :
  - o1 Ajout de la dimension DATE_TIME et la variable REFERENCE_DATE_TIME associée (de type ‘yyyyymmjjhhmmss’) qui correspond à la date grégorienne référence du fichier.
  - o2 la variable PTGS est remplacée par la variable JULD qui est le jour julien relatif à REFERENCE_DATE_TIME.
- •2 Dans le fichier NetCDF campagne,
  - o1 Ajout de la variable DATE_TIME_UTC qui contient les dates grégoriennes du fichier au format ‘yyyymmjjhhmmss’.
- •3 Dans la partie exploitation :
  - o1 ***net_vit_cour.m*** :
    - •1 l’ordre de flagguage a changé. On passe d’abord le filtre médian (flag 1 et 2) puis on applique les tests de flagguage 3, 4, 5, 6, 7 et 1.
    - •2 Un filtre glissant sur une fenêtre de 11 points (en temporel) a été ajouté.
  - o2 ***cree_sec.m*** : On peut créer des sections avec des flags autres que 0 ou 1.
  - o3 Création de ***contour_sec_all.m*** (appelé par ***contour_sec.m***) qui permet de contourer toutes les sections les unes après les autres sans que l’utilisateur n’ait à cliquer dessus.

En Juillet 2002, CASCADE a été modifié ainsi :

- •1 Modification de ***ad1_cree_nc.m***. L’année des données est directement issue des fichiers navigation RDI. La variable **cruise_year** reste mais n’est désormais utile que pour les tracés.
- •2 Ajout de la variable PTIM dans les fichiers générés par ***ad2_ensmoy.m***. Cette variable correspond à la moyenne de l’heure ADCP ‘brute’ (non corrigée du GPS) sur le nombre d’ensembles à moyenner. Elle devient la variable JULD_ADCP dans le fichier campagne (cf. ***ad3_creat_cdf_c.m***)
- •3 Modification de la partie associée au fichier campagne dans la partie exploitation
  - o1 Tracer de la dérive de l’horloge ADCP pour toute la campagne (via les variables JULD_ADCP et JULD) (cf. ***trace_derive_fic_camp.m***)

- o2 Tracer de variables 1D ou 2D du fichier campagne. Les variables 1D sont tracées selon le temps ; les variables 2D selon le temps et la profondeur (cf. ***trace_fic_camp.m***)
- o3 Calcul de la marée (cf. ***calcul_maree.m***) :
  - •1 Ajout dans le fichier campagne des variables :
    - •1 U_maree, V_maree : vitesses U et V issues de la marée en cm/s
    - •2 H_maree : Hauteur de la marée en m
    - •3 U_corrigee, V_corrigee : vitesses U et V du fichier campagne corrigées de la marée.
- o4 Filtrage du fichier campagne (cf. ***appel_filtre.m***, ***filtre.m***):
  - •1 Des vitesses U et V corrigées ou non de la marée selon le choix de l'utilisateur
  - •2 Filtrage sur 3 points consécutifs selon les règles :
    - •1 1/ 4, 1/2, 1/4 dans le cas général
    - •2 1/3, 2/3 sur les bords
  - •3 Divers filtrages sont proposés :
    - •1 Filtrage vertical (selon la profondeur)
    - •2 Filtrage horizontal (selon le temps)
    - •3 Filtrage vertical et horizontal
- o5 Amélioration de ***info_fic_camp.m*** afin de :
  - •1 Mieux formatter la sortie écran
  - •2 Ajouter le bilan de tous les flags
  - •3 Calculer la moyenne de la vitesse verticale des données de flag 0 (pour vérifier l'assiette du navire)

- •4 Création d'un interface ***demande_flag.m*** pour demander à l'utilisateur quels flags sont à prendre en compte dans ***cree_sec.m***
- •5 Mise en global des variables utilisées dans l'interface ***demande_nettoie.m***
- •6 Dans ***vecteur_sec.m***, la sélection des sections est affinée comme dans ***contour_sec.m***
- •7 Ajout du tracé des écart-types sur les tracés de profil de station (cf. ***profil_sta.m***)
- •8 Dans ***contour_sta.m***, on demande à l'utilisateur s'il souhaite des limites communes pour le tracé des contourages des diverses stations. Si oui, les limites sont fonction des extrema des vitesses du fichier.

1.5Evolutions prévues

- Intégration de la possibilité de lire le fichier de navigation au format TRINAV et de le transformer en un fichier TRINAV NetCDF.
- Ajouter la racine des fichiers NetCDF dans les informations utiles à l'étape 2.
- Convertir le fichier sauv_etat.mat en un fichier ASCII sauv_etat.dat comprenant 1 ligne par fichier. Chaque ligne comprendra :
  - Numéro du fichier
  - Flag indiquant si l'estimation de la dérive a été effectuée (flag=0 ➔ non effectuée)
    (flag=1 ➔ effectuée)
  - 1 Flag indiquant si l'heure ADCP a été corrigée
  - 2 Flag indiquant si l'attitude externe a été rajoutée
  - 3 Flag indiquant si la navigation (position, vitesse du navire) a été rajoutée
- Mise en place d'un flag de qualité du positionnement
- Mise en place de la possibilité d'éliminer un faisceau jugé mauvais
- Développement d'un script permettant à l'utilisateur de créer des fichiers NetCDF similaires à ceux issus de ***ad2_ensmoy.m*** à partir des fichiers ADCP « processed » issus de TRANSECT.

2.Introduction

Ce document présente les diverses étapes et principes mis en œuvre dans le logiciel de traitement des données d'ADCP de coque (CASCADE). Il est destiné à la personne chargée de sa maintenance ainsi qu'aux utilisateurs.
Le chapitre 3 décrit succinctement le principe de l'ADCP de coque. Le chapitre 4 est consacré au cœur du logiciel et décrit les différentes étapes du traitement. Le chapitre 5 est dédié à l'interface utilisateur tandis que le dernier chapitre est consacré aux annexes.

Le logiciel CASCADE présenté dans ce document a été développé au Laboratoire de Physique des Océans (LPO) de l'IFREMER. Pour toutes remarques, contacter les personnes dont les coordonnées suivent:

GAILLARD Fabienne
IFREMER - DRO/UM/LPO
B.P. 70
29280 PLOUZANE
tél. : 02.98.22.42.88
émail : Fabienne.Gaillard@ifremer.fr

KERMABON Catherine
IFREMER - DRO/UM/LPO
B.P. 70
29280 PLOUZANE
tél. : 02.98.22.42.84
émail : Catherine.Kermabon@ifremer.fr

3.Le courantomètre Doppler (ADCP)

3.1Principe

L'ADCP est un système de mesure de courant par effet Doppler. L'ADCP émet des signaux acoustiques, à une certaine fréquence, suivant plusieurs directions qui frappent les particules en suspension dans l'eau. Ces particules réfléchissent les signaux et les « renvoient » à l'ADCP. La fréquence de ces signaux rétrodiffusés, reçus par l'ADCP, diffère de la fréquence émise. C'est l'effet Doppler, induit par les vitesses du navire et des diffuseurs. Pour l'ADCP de coque, en fonction de ce changement de fréquence, on peut déterminer les composantes de la vitesse (V) des particules relative au navire selon la direction de chaque faisceau acoustique en utilisant la formule suivante (valable au premier ordre en négligeant un facteur correctif de l'ordre de 1 pour mille):

- $V^i = F^i{}_D * (C_S / 2\ F_S)$ où :
  - V^i représente la vitesse relative au navire selon l'axe du faisceau *i*
  - C_S représente la vitesse du son (en cm/s) au niveau de la source (c'est-à-dire des transducteurs)
  - F_S représente la fréquence émise par la source
  - $F^i{}_D$ représente la fréquence Doppler (la différence entre la fréquence émise et la fréquence reçue) pour le faisceau *i*

L'utilisation de plusieurs ondes sonores, généralement 4 faisceaux orientés différemment, permet, par simple calcul trigonométrique, de calculer les composantes de la vitesse du courant suivant les axes relatifs à l'appareil ADCP ainsi qu'une erreur.
A partir de l'angle d'alignement de l'ADCP par rapport à l'axe du navire et de son attitude (cap, roulis, tangage), ces composantes sont ensuite converties afin d'obtenir les composantes de vitesse des particules relative au navire selon les axes géographiques (Nord-Sud) et (Est-Ouest) (c'est-à-dire les vitesses relatives horizontales et verticales des particules). La vitesse absolue des particules est calculée à partir de la mesure ADCP (vitesse relative au navire) et de la vitesse du navire issue de la navigation. Les particules étant supposées sans mouvement propre, leur vitesse représente la vitesse de la masse d'eau dans laquelle elles se trouvent c'est-à-dire la vitesse du courant.
L'ADCP de coque permet ainsi d'obtenir des profils de courant allant de la surface à une profondeur de 800m (resp. 400m) pour un ADCP 75Khz (resp. 150 Khz) sur toute la trajectoire du navire où l'ADCP est installé.

3.2Terminologie

L'ADCP a la possibilité de réaliser une moyenne des impulsions ('***ping'***) émises. Un cycle de mesures ('***ensemble'***) est le résultat de la moyenne de plusieurs impulsions. En mode '***mono-ping***', on n'effectue aucune moyenne; un ensemble correspondant à un seul ping. A

chaque cycle de mesures, le profil de vitesse ADCP est découpé sur la profondeur en segments uniformes appelés ***cellules*** ou ***bins***. Pour chaque **ensemble**, on a donc autant de valeurs de vitesse que de **bins**. La vitesse associée à chaque cellule est une moyenne des vitesses sur l'épaisseur de la cellule.

4.Le logiciel de traitement

Le logiciel fourni par le constructeur de l'appareil permet d'effectuer en aveugle les corrections d'attitude à partir des données enregistrées dans l'appareil. Il peut ensuite effectuer des moyennes et soustraire la vitesse du navire à partir des données GPS.
Ce type de traitement présente des inconvénients pour les raisons suivantes :

- les données d'attitude enregistrées dans l'appareil peuvent ne pas être de la meilleure qualité disponible, en particulier les mesures de gyrocompas. On peut souhaiter effectuer la correction d'attitude à partir de données de GPS-3D.
- les données de position GPS qui servent à calculer la vitesse du navire ne sont pas synchronisées avec l'ADCP : la seule relation de date qui existe avec les données ADCP est leur heure d'arrivée sur le PC. D'autre part, même si le constructeur fournit un logiciel de navigation, on peut souhaiter lui substituer un autre type de traitement.

Aussi, il est souhaitable de partir des données brutes (la mesure du Doppler suivant chaque faisceau) et d'effectuer un traitement par étapes permettant d'incorporer des mesures externes d'attitude et de position.

4.1Présentation du logiciel

4.1.1Généralités - Type de données traitées

Afin d'effectuer une meilleure correction d'attitude, il est préférable de traiter des données ADCP enregistrées en mode 'mono-ping', bien que le logiciel permette le traitement d'ensembles de plusieurs pings. Néanmoins, dans ce cas, la correction du (roulis/tangage) qui est effectuée n'est pas correcte. En effet, pour un ensemble, on a une seule valeur de (roulis/tangage). C'est cette valeur qui est utilisée pour la correction d'attitude. Ainsi, en multi-ping, la correction n'est pas juste, l'attitude pouvant fortement varier entre le premier et le dernier ping de l'ensemble.
Au moment de la rédaction de ce document, le logiciel fonctionne uniquement :

- pour des ADCPs de coque installés à (+/-) 45 degrés par rapport à l'axe du navire.
- pour des ADCPs dont les faisceaux acoustiques sont à 30 degrés par rapport à la verticale.
- pour des données issues d'ADCPs de coque NarrowBand et BroadBand concaves enregistrées avec le logiciel d'acquisition « Transect ».

4.1.2Environnement de développement

Le logiciel de traitement des données ADCP de coque est entièrement basé sur le logiciel de visualisation et de calculs numériques **MATLAB** (logiciel commercialisé par la société Scientific Software). Aussi, pour pouvoir l'exécuter, l'utilisateur doit modifier sa variable **MATLABPATH** afin d'y ajouter les chemins :

- **/home3/doelan/chemon/soft_adcp/adcp_lpo/global** pour l'accès aux scripts et fichiers communs à l'ensemble de l'application
- **/home3/doelan/chemon/soft_adcp/adcp_lpo/etape1** pour l'accès aux scripts associés à la première étape du traitement
- **/home3/doelan/chemon/soft_adcp/adcp_lpo/etape2** pour l'accès aux scripts associés à la seconde étape du traitement
- **/home3/doelan/chemon/soft_adcp/adcp_lpo/etape3** pour l'accès aux scripts associés à la troisième étape du traitement
- **/home3/doelan/chemon/soft_adcp/adcp_lpo/exploit** pour l'accès aux scripts associés à l'exploitation des données
- **/home3/doelan/chemon/soft_adcp/adcp_lpo/ihm** pour l'accès aux scripts associés à l'interface utilisateur.

Pour ce faire, l'utilisateur peut lancer la commande ci-dessous :

source /home3/doelan/chemon/soft_adcp/adcp_lpo/cascade.env

Remarque :

- Le logiciel est basé sur des fichiers au format NetCDF, l'utilisateur doit donc s'assurer que les outils suivants sont disponibles :
  - librairie NetCDF version 3.4. On peut la récupérer à partir du site WEB *www.unidata.ucar.edu/packages/netcdf*.
  - toolbox NetCDF (mexcdf) pour Matlab. On peut la récupérer à partir du site WEB *crusty.er.usgs.gov/~cdenham/MexCDF/nc4ml5.html*.

- toolbox getcdf pour Matlab. On peut la récupérer à partir du site WEB *www.marine.csiro.au/sw/matlab-netcdf.html* (matlab-netcdf-4.0.tar.Z).

L'utilisateur doit mettre à jour la variable MATLABPATH en conséquence.

- Le logiciel nécessite également la toolbox m_map pour Matlab disponible sur le site WEB *beluga.ocgy.ubc.ca/~rich/map.html*

4.1.3Fichiers internes au logiciel CASCADE nécessaires

Pour sa bonne exécution, le logiciel CASCADE requiert 4 fichiers :

- 2 fichiers ASCII : ces fichiers sont à mettre à jour par la personne chargée de la maintenance du logiciel.
  - fichier ***info_adcp*** : Ce fichier comprend autant de ligne que d'ADCP de coque disponibles. Chaque ligne comprend :
    - le type d'ADCP
    - sa fréquence réelle
    - le facteur d'échelle dans le cas où les données sont enregistrées en 'HIGH' range
    - le facteur d'échelle dans le cas où les données sont enregistrées en 'LOW' range

      Le facteur d'échelle permet de passer des vitesses exprimées en 'compte' Doppler en cm/s.

    Ce fichier se trouve en annexe.
  - fichier ***info_navire*** : Ce fichier comprend une ligne par couple (Navire-ADCP). Chaque ligne comprend :
    - le nom du navire
    - le type de l'ADCP
    - la profondeur des transducteurs
    - l'angle des faisceaux par rapport à la verticale
    - l'angle de l'ADCP par rapport à l'axe du navire
    - la convention C de signe pour le roulis interne :
      - C = 1 : le roulis est positif lorsque le navire gîte sur bâbord
      - C = -1 : le roulis est positif lorsque le navire gîte sur tribord
    - la convention D de signe pour le tangage interne :
      - D = 1 : le tangage est positif lorsque le navire lève le nez
      - D = -1 : le tangage est positif lorsque le navire baisse le nez

    Ce fichier est fourni en annexe.
- 2 fichiers au format MAT-file
  - ***conf.mat*** : Ce fichier permet de mémoriser les variables et constantes de traitement utiles à la bonne exécution du logiciel CASCADE.
  - ***sauv_etat.mat*** : Ce fichier comprend des 'flags' d'avancement du travail, à savoir :
    - **ajout_DGPS** : cette variable comprend les numéros de fichiers sur lesquels l'estimation de la dérive a été effectuée.
    - **modif_heure** : cette variable comprend les numéros de fichiers pour lesquels la correction de l'heure a été effectuée.
    - **modif_att** : cette variable comprend les numéros de fichiers pour lesquels la correction de l'attitude a été effectuée.

- **ajout_nav** : cette variable comprend les numéros de fichiers pour lesquels la navigation (position, vitesse navire) a été rajoutée.

Sous le répertoire /home3/doelan/chemon/soft_adcp/adcp_lpo/global, ces 2 fichiers contiennent les variables et les constantes de traitement initialisées à des valeurs par défaut. Lorsque l'utilisateur débute un traitement, ce sont ces fichiers qui sont préalablement chargés. Par la suite, ces fichiers seront stockés sous le répertoire de travail de l'utilisateur.

4.1.4Fichiers Utilisateurs nécessaires

Le logiciel nécessite que l'utilisateur ait à sa disposition les fichiers suivants :

- les fichiers de **données ADCP brutes** enregistrées en 'BEAM coordinated' et en mode **mono-ping** (de préférence) (fichiers ADCP **r.???)*. Dans ces fichiers, on trouve notamment les fréquences Doppler par bin pour chaque faisceau. *A noter que pour le bon fonctionnement du logiciel, ces fichiers doivent avoir une durée comprise entre 6 et 24 heures*.
- les fichiers navigation associés **n.???*

Ces 2 types de fichiers sont issus du logiciel d'acquisition ADCP TRANSECT.

A partir de ces fichiers , l'utilisateur peut effectuer un pré-traitement des données à bord du navire. Si par la suite, il souhaite traiter les données avec une meilleure navigation que celle disponible en temps réel, il doit alors avoir à sa disposition la navigation issue de TRINAV sous la forme d'un fichier NetCDF comportant les positions et vitesses du navire. Le logiciel requiert que ces données aient pour nom : Lat_TRIN, Lon_TRIN, T_TRIN, VitU_TRIN et VitV_TRIN.

-

4.1.5Les différentes étapes

Le logiciel comporte 4 étapes consécutives bien distinctes :

1. Création des fichiers NetCDF contenant les mesures ADCP et des fichiers NetCDF contenant l'attitude et les positions - Ajout des données date corrigée et attitude externe
   - Conversion des fichiers RDI 'r' et 'n'
   - Calcul et ajout de l'heure ADCP corrigée du GPS
   - Ajout des données d'attitude externe
2. Création de fichiers NetCDF équivalents aux fichiers 'processed' issus du logiciel d'acquisition de données ADCP de coque TRANSECT
   - Conversion des données en coordonnées terrestres
   - Filtrage et moyenne de ces données
   - Estimation de l'angle de désalignement, du facteur d'amplitude et de l'erreur sur l'assiette du navire
3. Calcul des vitesses absolues (création de fichiers NetCDF campagne)
4. Exploitation des données (à partir d'un fichier campagne)
   - Nettoyage des données
   - Contrôle de qualité
     - ◊ comparaison vitesse navire/vitesse du courant
     - ◊ comparaison vitesse du courant en route/en station
     - ◊
   - Fichier campagne
     - ◊ Ajout de la marée

- ◊ Tracé de la dérive
- ◊ Tracé des variables à 1 ou 2 dimensions
- ◊ Filtrages des vitesses U et V

- Création de fichiers NetCDF section et de fichiers NetCDF station
- Tracé à partir de ces fichiers :
  - ◊ contourage
  - ◊ vecteurs
  - ◊ profils pour les stations
-

•

4.2Etape 1: Mise au format NetCDF et ajout des données annexes

4.2.1Mise au format

ad1_cree_nc.m :

Lors de l'acquisition des données, entre 2 arrêts de l'ADCP de coque, toutes les données brutes sont enregistrées dans un ensemble de fichiers de nom ayant une racine commune (exemple : ***cam98013r***.000, ***cam98013r***.001, ...etc.). ***ad1_cree_nc.m*** convertit ces fichiers binaires des données brutes en des fichiers au format NetCDF (**Net**work **C**ommon **D**ata **F**orm : format binaire auto-descriptif). Pour un même ensemble de fichiers (***cam98013r***.000, ***cam98013r***.001, ...etc.), on crée un fichier binaire ADCP NetCDF (***cam98013r***.nc). La structure du fichier NetCDF créé est présentée en annexe.

Remarque :

- Dans les fichiers NetCDF ADCP créés, les conventions de signe sont les suivantes :
  - le roulis est positif lorsque le navire gîte a bâbord
  - le tangage est positif lorsque le navire lève le nez
- Lors de l'acquisition des données, il est conseillé, pour des raisons de maniabilité des fichiers et de calcul de dérive, de *créer des fichiers bruts de données d'une durée de 6 à 24h* (12h semble optimal). De plus, il est à noter que lors de l'acquisition, les ensembles sont numérotés modulo 65535.
- Selon le type d'ADCP, ***ad1_cree_nc.m*** fait appel à :
  - ***ad1_brut2cdf_nb.m*** pour les ADCPs NarrowBand
  - ***ad1_brut2cdf_bb.m*** pour les ADCPs BroadBand

4.2.2Correction de l'heure ADCP

Les données ADCP sont datées au moyen d'une horloge interne qui dérive lentement. Dans la suite du traitement, ces données seront combinées à des mesures externes d'attitude et de position datées avec l'heure BORD. La première tâche à effectuer est donc d'évaluer la dérive de l'horloge interne de l'ADCP afin de corriger la date ADCP attachée aux données par rapport à la date BORD.

ad1_estim_derive_TT.m :

Outre les fichiers de données brutes ADCP, lors de l'acquisition, des fichiers navigation RDI (cam98013n.000, cam98013n.001, ...etc.) sont générés et enregistrés sur le PC d'acquisition. Ces fichiers contiennent des informations issues de l'ADCP et des informations issues du BORD. Une ligne ADCP comporte un numéro d'ensemble et l'heure du PC associé. Une ligne BORD comprend l'heure BORD, la position et l'attitude correspondantes (cf. annexe).

ad1_estim_derive_TT.m concatène tous les fichiers navigation RDI (*n.???) issus d'un même ensemble de fichiers, fait appel à ***ad1_derive_adcp*** qui calcule la dérive ADCP moyenne (en seconde) par rapport au GPS (+ un temps de transfert) pour cet ensemble de fichiers. Cette valeur est mémorisée dans la variable ***DGPS*** dans le fichier ADCP NetCDF correspondant.

***ad1_derive_adcp.m* :**

Ce script calcule la différence entre l'heure BORD et l'heure ADCP (variable **TIM** issue du fichier ADCP NetCDF) associée à l'ensemble qui précède la ligne BORD. Ceci est réalisé pour toutes les heures BORD rencontrées dans le fichier *n.??? concaténé. Les différences temporelles calculées représentent la somme :

Dérive_horloge_adcp_par_rapport_au_bord + temps_transfert + écart au dernier message BORD

Le temps de transfert correspond au :

1. temps de transfert de l'ensemble entre l'ADCP et le PC d'acquisition
2. temps de transfert entre le BORD et le PC d'acquisition (négligeable)

A partir des fichiers navigation RDI, ce script crée aussi les **fichiers d'attitude et position externe NetCDF** comprenant :

- Heure BORD
- Latitude
- Longitude
- Vitesse U du navire
- Vitesse V du navire
- Accélération en X du navire
- Accélération en Y du navire
- Cap externe
- Roulis externe
- Tangage externe

Il est à noter qu'un filtre de Butterworth de largeur 2*16s est appliqué sur les vitesses et accélérations navire (cf . ***ad1_filtre_pos.m***). Ce filtre permet d'éliminer les données aberrantes ainsi que l'effet de la houle (d'une période de 15s environ). Cet effet est à supprimer car pour le calcul des vitesses absolues, on déduit les vitesses navire des vitesses ADCP sur des ensembles moyennés sur 1 à 2 minutes. Pour ces ensembles, l'effet de la houle ne subsiste donc plus.

Ce fichier NetCDF d'attitude comprend la navigation brute et non la navigation intégrée car cette dernière comprend des filtres non adaptés au traitement ADCP. Le traitement final peut s'effectuer par la suite à partir d'une meilleure navigation (issue de TRINAV par exemple). (cf. ***Cdf_att.m, ad1_corr_att.m, ad2_adnav.m***)

Remarque :

- La *dérive ADCP finale (+ temps de transfert ADCP-PC)* pour un fichier NetCDF ADCP est la moyenne de toutes les différences temporelles ***inférieures à 15 minutes*** calculées sur les fichiers navigation RDI (*n.???) associés. La moyenne n'est effectuée que dans le cas où le nombre de différences à moyenner (inférieures à 15 minutes) est supérieur à 100. Dans le cas contraire, la valeur par défaut est conservée.

ad1_cal_vraie_derive.m :

Ce script permet de visualiser les données du fichier NetCDF d'attitude externe datées avec le BORD (+ temps de transfert) et les données d'attitude interne des fichiers ADCP NetCDF (variables **Hdg, Ptch, Roll**) datées avec l'ADCP (+ temps de transfert) (variable **TIM**). Les temps de transfert entre la centrale d'attitude et le Bord et entre la centrale

d'attitude et l'ADCP étant négligeables, visuellement, on peut donc estimer la vraie dérive entre l'heure ADCP et l'heure GPS.
Concrètement, ***ad1_cal_vraie_derive.m*** permet à l'utilisateur de tracer sur un même graphique le cap (resp. roulis, tangage) externe en fonction de l'heure BORD et le cap (resp. roulis, tangage) interne ADCP en fonction de l'heure ADCP. L'utilisateur peut ensuite modifier l'heure ADCP en lui ajoutant (+/-) X secondes afin de superposer au mieux les 2 courbes. (+/-) X représente la vraie dérive. Cette vraie dérive est déterminée à différents instants et est mémorisée dans un fichier « Derives » comprenant :

numéro_fichier_NetCDF_ADCP	heure_adcp (en jour grégorien) JJ/MM/AAAA HH:MIN:SEC	vraie_dérive (en seconde)

Remarques :

- Le « calcul » de la vraie dérive ne s'effectue que si le fichier en cours à traiter a une durée supérieure à 10 minutes.
- Si l'utilisateur ne dispose pas de fichiers similaires aux fichiers Bord NetCDF (s'il ne dispose pas de données d'attitude externe datées avec le GPS), il peut néanmoins créer manuellement un fichier « Derives ». Celui-ci peut contenir, par exemple, la dérive de l'ADCP en début de campagne et la dérive de l'ADCP en fin de campagne.
- Même si il y a eu plusieurs recalages de l'horloge de l'ADCP au cours de la campagne, l'utilisateur ne doit créer qu'un seul fichier « Derives ».
- Le roulis et le tangage internes à l'ADCP sont dans le repère de l'ADCP. Le roulis et le tangage externes sont dans le repère navire. Aussi, pour pouvoir comparer les 2, il faut passer le roulis et le tangage externes du repère navire au repère ADCP. Pour ce faire, on applique la formule fournie par GENAVIR :
  Sin(tangage_adcp) = [sin(tangage_navire) – sin(roulis_navire)]*sqrt(2)/2
  Sin(roulis_adcp) = [sin(tangage_navire) + sin(roulis_navire)]*sqrt(2)/2
  Cette formule est correcte avec la convention GENAVIR pour le roulis (convention inverse à CASCADE).

Pour le reste du traitement, on utilise le **roulis et le tangage externes dans le repère navire**.

ad1_cal_poly.m :

Ce script calcule le polynôme P tel que : *P(heure_adcp) = vraie_derive*
Pour ce faire, on calcule le polynôme P1 tel que : P1(heure_adcp) = dérive estimée (dérive + temps de transfert). On applique P1 aux heures ADCP pour lesquelles la vraie dérive a été déterminée. En comparant le résultat avec les vraies dérives, on déduit le temps de transfert moyen (TT_M).

TT_M = mean(P1(heure_adcp) - derive vraie(heure_adcp)).

On détermine ensuite le polynôme final P(heure_adcp) passant par les vraies dérives et par les (dérives estimées - TT_M). On trace sur un même graphique le polynôme calculé et les points qui ont permis de le déterminer (les vraies dérives et les [dérives GPS estimées - TT_moyen]).

Remarque :

- S'il le souhaite, l'utilisateur peut éliminer des points qui lui semblent aberrants pour déterminer au mieux le polynôme.

- Le calcul du polynôme s'effectue uniquement sur les heures ADCP comprises dans les fichiers en cours à traiter et non pas sur toutes les dérives du fichier « Dérives ».
- Si l'horloge ADCP est remise à l'heure en cours de campagne, il faut effectuer un calcul du polynôme par période.
- Par défaut, pour un ensemble de fichiers allant du numéro X au numéro Y, le polynôme P est stocké dans un fichier de nom : *pX_Y.pol*

ad1_corr_heure.m :

Ce script corrige les heures ADCP (variable **TIM**) de leur dérive en appliquant le polynôme P déterminé précédemment. Pour chaque ensemble de chaque fichier ADCP NetCDF, l'heure corrigée se calcule ainsi :

- heure_corrigée_en_jour_decimal = **TIM** + (P(**TIM**)/86400)

Le résultat est mémorisé dans la variable **TGPS** du fichier.

4.2.3Correction de l'attitude (cap, roulis, tangage)

Les fichiers ADCP NetCDF comportent l'attitude issue de la centrale d'attitude (variables **Hdg, Ptch, Roll**). Lors du traitement, on peut souhaiter leur substituer des données d'attitude externe, soit parce que des données de meilleure qualité, telles que celles fournies par le GPS-3D, sont disponibles, soit parce que suite à un problème de transmission entre la centrale d'attitude et l'ADCP, ces données n'ont pas été enregistrées. Pour utiliser ces données d'attitude externe, il faut les fournir aux instants de mesure ADCP.

ad1_corr_att.m :
Ce script permet d'interpoler l'attitude externe (cap, roulis, tangage issus des fichiers NetCDF d'attitude) aux heures ADCP corrigées de la dérive (**TGPS**).
Ces valeurs d'attitude sont mémorisées dans le fichier ADCP NetCDF sous les variables **HdgExt, PtchExt, RollExt.** Pour les visualiser, ***ad1_corr_att.m*** appelle ***ad1_atti_pl0.m***.

A ce niveau, les fichiers ADCP NetCDF sont similaires aux fichiers ADCP NetCDF générés via ***ad1_cree_nc.m*** si ce n'est que désormais ces fichiers comportent les variables **TGPS, HdgExt, PtchExt, RollExt** qui permettent d'avoir les données ADCP avec une heure et une attitude de meilleure qualité (généralement recalées par rapport au GPS et GPS-3D).

4.3Etape 2: Création des fichiers « processed »

Cette étape permet de créer des fichiers similaires aux fichiers 'processed' générés par le logiciel d'acquisition TRANSECT. Dans ce fichier, les vitesses relatives au navire sont exprimées dans le repère terrestre. Les ensembles des fichiers 'raw' (correspondant à un ou plusieurs pings) sont moyennés sur un nombre *nb_ens_moy* choisi par l'utilisateur. Les informations de navigation (positions et vitesses du navire) sont ajoutées aux dates des ensembles moyens. Enfin, il est possible à ce niveau de calculer les angles de désalignement et le facteur d'amplitude. Cette étape comprend donc trois opérations:

5. Transformation de la mesure suivant 4 faisceaux en une mesure de vitesse dans le repère terrestre et moyennage des ensembles successifs
6. Ajout de la navigation
7. Calcul des angles de désalignement et du facteur d'amplitude

4.3.1Installation de l'ADCP, définition des angles

Les angles utilisés pour les calculs adoptent la convention trigonométrique directe. Les systèmes d'axes suivent la même convention:

- *X* positif vers la droite
- *Y* positif vers l'avant
- *Z* positif suivant la verticale ascendante

Le logiciel est prévu pour un ADCP de type concave, installé sur le navire avec un angle : Δ = ***angle_adcp*** par rapport au cap navire (cet angle est généralement de (+/-) 45 °).
Dans le plan horizontal, on a entre chaque faisceau un angle de 90 °. La numérotation des faisceaux est indiquée figure 1.
Le repère ADCP est défini, après croisement des faisceaux, par les axes suivants:

- *X* va du faisceau 1 au faisceau 2
- *Y* va du faisceau 4 au faisceau 3
- *Z* est défini dans le sens direct par rapport à *X* et *Y*

Chaque faisceau émis forme un angle θ = ***BeamAngle*** avec l'axe Z.

	NB75 Thalassa	BB150 Thalassa	NB75 Atalante	NB75 Entrecasteaux	BB150 Suroit
angle_adcp	+ 45 °	+ 45 °	+ 45 °	- 45 °	+ 45°
BeamAngle	30 °	30 °	30 °	30 °	30°
Fréquence d'émission (kKz)	76.8	153.6	76.8	76.8	153.6

Tableau 1: Paramètres de l'installation de l'ADCP sur différents navires

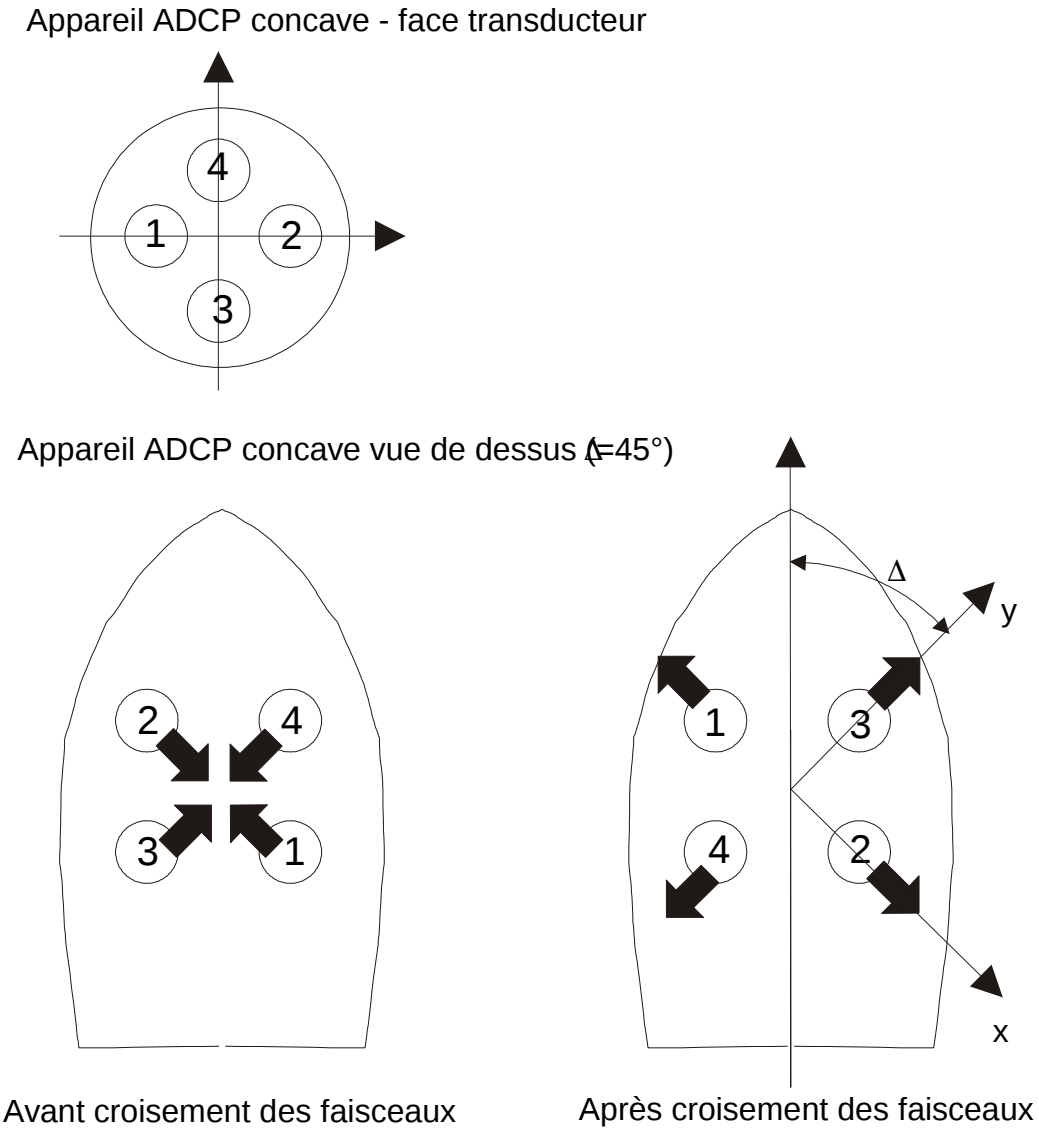


Figure 1. Installation de l'ADCP de coque

4.3.2Création des ensembles moyens (*ad2_ensmoy.m*)

Ce programme traite les fichiers ADCP NetCDF issus de l'étape 1. Les fichiers sont pris en compte les uns après les autres. Ce programme effectue les diverses étapes présentées dans l'annexe C « Calculations and Formulas » de la documentation RDI « Vessel-mounted Acoustic Doppler Current Profiler : Technical Manual ». Le programme effectue 5 opérations successives:

1. Il réaffecte les bins afin de compenser pour le roulis-tangage (***ad2_bincorr.m***).
   Pour ce faire, c'est le **roulis et le tangage externes** qui sont utilisés.
2. Il calcule la vitesse radiale correspondant à la vitesse du son au transducteur.
3. Pour chaque profil, il transforme les vitesses radiales suivant 4 faisceaux en vitesse relative au navire, exprimée dans le repère terrestre suivant trois composantes: (vitesse zonale (U), méridionale (V) et verticale (W)) (***ad2_Beam2geo.m)***. Les vitesses supérieures, en valeur absolue, à une vitesse (**vit_max**) fixée par l'utilisateur sont éliminées.

4. Ces vitesses sont ensuite nettoyées des valeurs aberrantes (***ad2_calmoy.m*** et ***ad2_Net_dat.m***) et moyennées sur un nombre d'ensembles (**nb_ens_moy**) consécutifs (nombre défini par l'utilisateur) (***ad2_calmoy.m***).
5. Ensuite un fichier Processed ADCP NetCDF est créé afin de mémoriser les résultats (***ad2_creat_cdf_p.m***). La structure de ce fichier est présentée en annexe.

4.3.2.1Réaffectation des bins

Cette opération consiste à recaler la profondeur des cellules pour compenser l'inclinaison du navire sous l'effet du roulis-tangage. Les mesures du bin *i*, censées représenter la couche *i* quand le navire est horizontal, représentent en fait la couche *j*. Il faut donc réaffecter chaque mesure au bin qui convient (figure 2).

Si l'angle θ devient θ', $i \cos(\theta') = j \cos(\theta)$, par conséquent, la mesure du bin *i* doit être placée dans le bin *j* qui vérifie:

$$j = i \cos(\theta')/\cos(\theta)$$

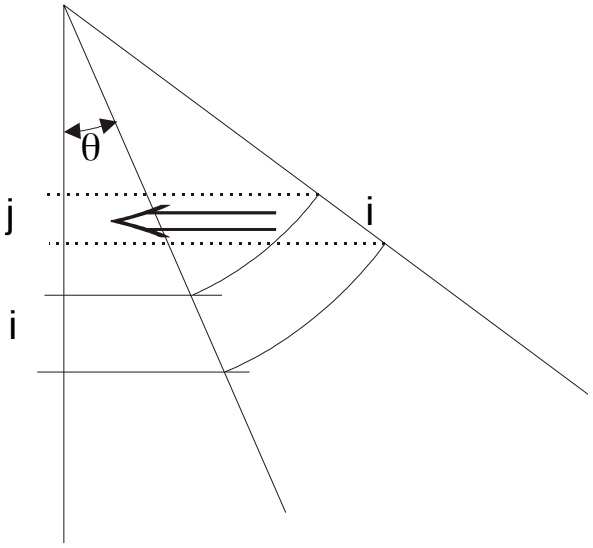


Figure 2. Réaffectation des bins

Dans le cas d'un ADCP installé à 45°, la modification de θ sous l'effet du roulis-tangage peut être décomposée en un effet du roulis, qui affecte les plans latéraux du navire contenant deux faisceaux, et un effet de tangage qui affecte les plans avant et arrière, contenant chacun deux faisceaux. La figure 3 définit l'angle γ du plan des faisceaux avec la verticale. Si l'on prend le cas du plan latéral tribord, le roulis transforme γ en $\gamma' = \gamma + \delta$.

La relation entre γ et θ est: $\tan\theta' = \sqrt{2}\tan\gamma'$

soit $\theta' = Atan(\sqrt{2}\tan(\gamma + \delta))$,

γ est donné par : $\gamma = Atan(\tan(\theta)/\sqrt{2})$,

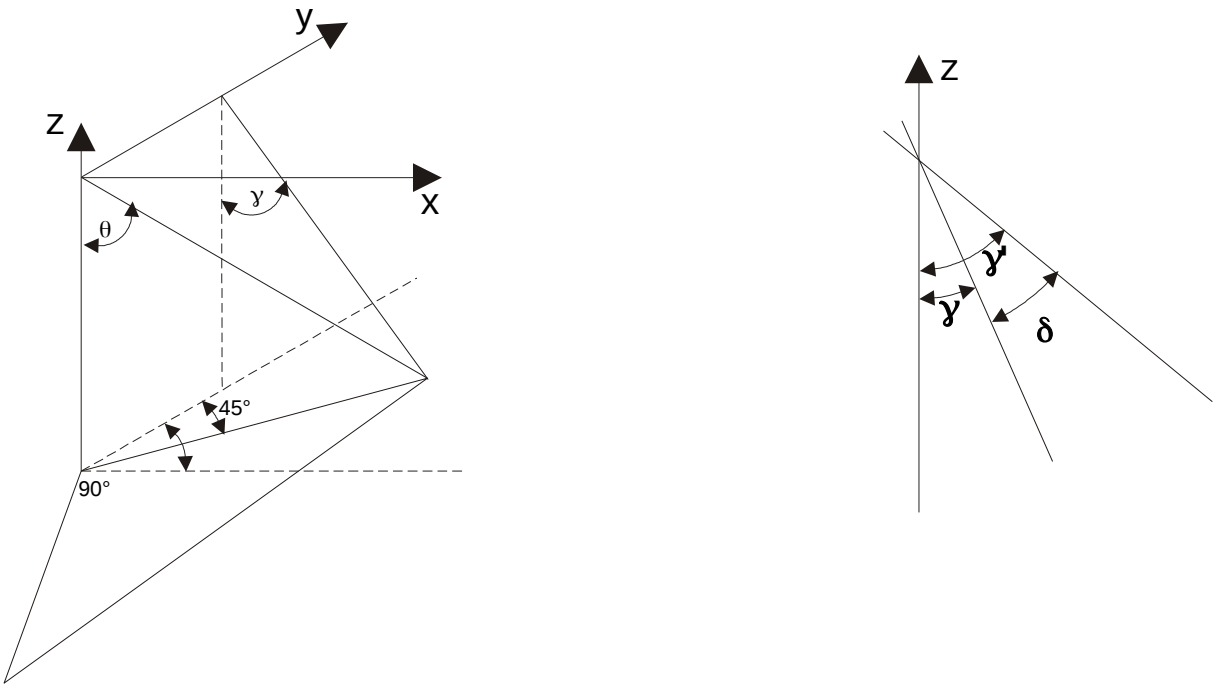


Figure 3: Orientation du plan contenant deux faisceaux dans le cas d'une installation à 45 °

4.3.2.2Calcul des vitesses radiales corrigées de la vitesse du son

Cette partie du programme calcule le facteur de conversion qui sera appliqué à la mesure ADCP. Les vitesses sont exprimées en cm/s.

- Pour un NB 75, la mesure est un « compte » Doppler, la vitesse radiale est :
  $V_i = F_d * C_t/(2*F)$
- Pour le BB150, la mesure est une vitesse en mm/s, calculée à partir d'une vitesse du son C0. Le facteur de conversion est alors :
  $V_i = V_{i0} * C_t / C_0$
  - V_i : Vitesse radiale corrigée
  - F_d : Fréquence Doppler (Hz)
  - C_t : Vitesse du son au transducteur (m/s)
  - F : Fréquence d'émission de l'ADCP (Hz)
  - V_{i0} : Vitesse radiale calculée avec une vitesse du son C_0
  - C_0 : Vitesse du son interne à l'ADCP (m/s)

4.3.2.3Transformation des vitesse radiales en vitesse dans le repère terrestre (ad2_Beam2geo.m)

Transformation en vitesse dans le repère ADCP:

Cette transformation se fait en deux étapes. La première consiste à transformer les 4 vitesses radiales en vitesses suivant trois composantes relatives à l'ADCP exprimées dans le repère de l'ADCP (figure 1).

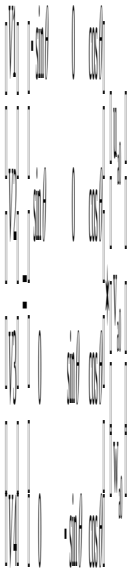
Chaque vitesse radiale V_i représente la projection de la vitesse $Ua0 = (u_{a0}, v_{a0}, w_{a0})$ sur la direction du faisceau *i*

$$V_i = \sin\theta\,(u_{a0}\cos\varphi_i + v_{a0}\sin\varphi_i) + w_{a0}\cos\theta$$

Selon les faisceaux, les valeurs de $cos\varphi_i$, $sin\varphi_i$ sont:

faisceau	φ_i	$\cos\varphi_i$	$\sin\varphi_i$
1	π	-1	0
2	0	1	0
3	$\pi/2$	0	1
4	$-\pi/2$	0	-1

On peut donc définir le système linéaire suivant:



Soit Y = G X en notation matricielle. La solution moindre carrés (celle qui minimise la norme des résidus de l'équation: $\varepsilon = Y - GX$) est donnée par :

$$X = (G^T G)^{-1} G^T Y$$

soit :

$$U_{ao} = \begin{bmatrix} \frac{-V_1}{2\sin\theta} + \frac{V_2}{2\sin\theta} \\ \frac{V_3}{2\sin\theta} - \frac{V_4}{2\sin\theta} \\ \frac{1}{4\cos\theta} * (V_1 + V_2 + V_3 + V_4) \end{bmatrix}$$

L'estimation de l'erreur correspondante est :

$$e = \frac{1}{4}(V_1 + V_2 - V_3 - V_4)$$

Remarque : Cas 3 faisceaux
Si 1 des 4 faisceaux est en panne, les composantes U, V et W sont déterminées à partir de 3 équations, il n'est plus possible d'estimer l'erreur qui est alors fixée à une valeur standard : la valeur max - 1 (**err_max - 1)**.
Soit le faisceau *i* en panne. La matrice G est identique à celle définie ci-dessus ôtée de la ligne *i*. G est une matrice carrée et la solution X est alors donnée par la formule : $X = G^{-1} Y$.

Transformation en vitesse dans le repère géographique*:*
Cette transformation s'opère en effectuant des rotations successives:

- Une première rotation de **angle_adcp** autour de l'axe Z pour se placer dans le repère navire (décrite par la matrice M_{off})
- Une rotation autour de l'axe des X pour compenser du tangage (décrite par la matrice M_P)
- Une rotation autour de l'axe des Y pour compenser du roulis (décrite par la matrice M_R)
- Une rotation autour de l'axe des Z pour placer l'axe Y en direction du nord (décrite par la matrice M_H)

$$U_{ro} = M_H * M_R * M_P * M_{off} * U_{ao}$$

Remarque : Attention aux rotations à appliquer :
Le roulis et le tangage sont donnés dans le sens trigonométrique direct, le cap et l'angle de l'ADCP par rapport à l'axe du navire sont donnés dans le sens trigonométrique indirect.

- Les données de cap et de (roulis/tangage) proviennent des données externes ajoutées au cours de la correction de l'attitude (***ad1_corr_att.m***).

Remarque : Calcul avec 3 ou 4 faisceaux
Pour diverses raisons, il se peut qu'un des faisceaux de l'ADCP ne fonctionne pas correctement ou soit en panne. Aussi, ***ad2_Beam2geo.m*** fait appel à l'un des scripts suivants :

- ***ad2_Beam2ADCP_0.m*** : Ce script est appelé dans le cas où les 4 faisceaux fonctionnent. Néanmoins, la portée d'un faisceau pouvant éventuellement être moindre, pour chaque bin de chaque profil, ce script essaie en premier lieu de déterminer une solution à 4 faisceaux. Si c'est impossible, le script tente alors de trouver une solution à 3 faisceaux. Ainsi, pour un même profil, on peut éventuellement avoir, en fonction de la profondeur, une solution à 3 ou 4 faisceaux.
- ***ad2_Beam2ADCP_1.m*** : Ce script est appelé dans le cas où le faisceau 1 ne contient pas de donnée.
- ***ad2_Beam2ADCP_2.m*** Ce script est appelé dans le cas où le faisceau 2 ne contient pas de donnée.
- ***ad2_Beam2ADCP_3.m*** : Ce script est appelé dans le cas où le faisceau 3 ne contient pas de donnée.
- ***ad2_Beam2ADCP_4.m*** : Ce script est appelé dans le cas où le faisceau 4 ne contient pas de donnée.

4.3.2.4Moyennage des ensembles successifs (ad2_calmoy)

Le moyennage des ensembles (de profils) est effectué par la partie ***ad2_calmoy.m*** qui opère de la façon suivante pour chaque paquet de **nb_ens_moy** ensembles consécutifs:

- Calcul des dates (***PTIM***, ***PTGPS***), de la température et de l'attitude moyennes des ensembles
  (PTIM : date ADCP 'brute' , PTGPS : date ADCP corrigée de la dérive par rapport au GPS)
- Tri des données de vitesse acceptables. Pour ce faire, on utilise l'erreur estimée par ***ad2_Beam2geo***. Si cette erreur est inférieure au seuil **err_max**, la donnée est considérée bonne. On effectue le même test sur la vitesse verticale.
  Remarque 1: si les vitesses ont été estimées avec trois faisceaux seulement, le calcul de l'erreur est impossible, celle-ci a été arbitrairement fixée à **err_max – 1**
  Remarque 2 : **err_max** doit être typiquement de l'ordre de (2.5*écart-type du bruit annoncé par le constructeur)
- Suppression des mesures hors norme sur critère de variance. Les vitesses (U, V et W) associées à un bin d'un ensemble donné sont éliminées si l'une d'entre elles s'écarte de plus de **n_std** écart-type de sa moyenne calculée sur les **nb_ens_moy** ensembles. Ce test d'écart à la moyenne est réitéré **nitmax** fois (n_std et nitmax fixés par l'utilisateur). Ce test est effectué via l'appel à ***ad2_Net_dat.m***.
- Calcul des moyennes. Sont calculées: les trois composantes de la vitesse et leur écart RMS, l'intensité d'écho rétrodiffusé (ECI) par faisceaux et moyenne sur les 4 faisceaux, le pourcentage de valeurs de vitesse correctes. Ce pourcentage remplace le « percent good » de RDI. En effet, si on travaille en mono-ping, ce qui est recommandé pour CASCADE, le percent-good des fichiers RDI vaut 0 ou 100 et ne constitue pas une information de qualité mais seulement d'absence ou de présence de donnée.
- La moyenne des vitesses (U, V et W) sur les **nb_ens_moy** ensembles de chaque bin est conservée si et seulement si cette moyenne a été calculée avec plus de **pg_min** % de valeurs de vitesse.

4.3.3Ajout de la navigation (*ad2_adnav.m)*

Le script ***ad2_adnav.m*** rajoute dans chaque fichier « processed » ADCP NetCDF les données de navigation (latitude, longitude, vitesse zonale du navire, vitesse méridienne du navire) correspondantes. Ces données sont issues du fichier NetCDF d'attitude et position ou du fichier NetCDF navigation TRINAV et interpolées aux heures ADCP (***PTGPS***).

Remarque :
Si les données de navigation sont échantillonnées à plus haute fréquence que les données ADCP, elles sont filtrées et rééchantillonées aux dates de mesure ADCP par interpolation. On interpole entre 2 valeurs 'correctes', ces 2 valeurs pouvant être éloignées si plusieurs données sont consécutivement mauvaises.
A l'opposé, si les données de navigation sont moins fréquentes que les données d'ADCP, un message d'avertissement apparaît. Dans ce cas, les vitesses navire ne seront pas comparables aux vitesse ADCP, car moyennées sur des périodes plus longues.

4.3.4Calcul du désalignement

4.3.4.1Effet des désalignements et de l'erreur d'amplitude

Soit $\vec{U}_r$ la vitesse ADCP relative au navire exprimée dans le repère géographique. La vitesse absolue du courant $\vec{U}_c$ sera la somme de la vitesse relative au navire et de la vitesse du référentiel en mouvement (le navire):

$$\vec{U}_c = \vec{U}_r + \vec{U}_n$$

Si les axes du repère ADCP ne sont pas parfaitement alignés avec le marbre navire auquel sont rapportées les mesures d'attitude (à la rotation ***angle_adcp*** près), les écarts d'angles introduisent une erreur dans le calcul de la vitesse relative au navire $\vec{U}_r$. Les erreurs d'alignement qui importent sont celles sur le cap et le tangage. L'erreur due au roulis a un effet d'ordre supérieur. Si l'on effectue les rotations pour tenir compte de ces erreurs d'alignement dans le cap et l'assiette, la vitesse relative au navire s'exprime:

$$\vec{U}_r = M_H * M_R * M_P * M_{P'} * M_{H'} * M_{off} * \vec{U}_a$$

où

$M_{H'}$ et $M_{P'}$ sont respectivement les matrices de rotation correspondant aux angles de désalignement de cap et de tangage.

D'autre part, l'amplitude du courant mesuré par l'ADCP peut elle aussi être entachée d'un biais, et si A est le facteur d'erreur sur l'amplitude.:

$$\vec{U}_{ao} = A * \vec{U}_a$$

Si l'on note M_G la matrice de rotation du repère navire vers le repère géographique:

$$M_G = M_H * M_R * M_P$$

et M_{corr} la matrice de correction:

$$M_{corr} = M_{P'} * M_{H'}$$

alors la vitesse relative au navire calculée à l'étape 2 s'écrit

$$\vec{U}_{ro} = A * M_G * M_{off} * \vec{U}_a$$

et la vitesse relative correcte est donnée par :

$$\vec{U}_r = A^{-1} * M_G * M_{corr} * M_G^{-1} * \vec{U}_{ro}$$

Pour corriger les vitesses relatives avant de leur ajouter la vitesse navire, il faut donc connaître le cœfficient A et la matrice M_{corr}. Les matrices de correction pour chacun des angles sont:

$$M_{H'} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_{P'} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix}$$

A l'ordre 1, en supposant α et β petits, $\cos\alpha = \cos\beta = 1$, $\sin\alpha = \alpha$ et $\sin\beta = \beta$. Il en découle que $Mcorr = \begin{bmatrix} 1 & \alpha & 0 \\ -\alpha & 1 & \beta \\ 0 & -\beta & 1 \end{bmatrix}$

De même, A peut s'écrire A = 1 + a, où a << 1.

La vitesse non corrigée du courant relative au navire est :

$$\vec{U}_{ro} = M_G M_{off} \vec{U}_{ao} \ ,$$

La vitesse correcte qui tient compte des erreurs de désalignement et du facteur d'amplitude est :

$$U_r = A^{-1} M_G M_{corr} M_{off} U_{a0} \ .$$

Si l'on écrit ces relations dans un repère fixe, mais aligné avec les axes navires on obtient:

$$U^n_{r0} = M_{off} U_{a0} = U^n_{c0} - U^n_n \ ; \ U^n_r = A^{-1} M_{corr} M_{off} U_{a0} = U^n_c - U^n_n \ .$$

La relation entre le courant vrai U^n_c et le courant modifié par les erreurs d'alignement et d'amplitude U^n_{c0} s'exprime :

$$U^n_c - U^n_n = A^{-1} M_{corr} \left(U^n_{c0} - U^n_n \right).$$

Si l'on développe cette relation pour les 2 composantes horizontales, en négligeant les vitesses verticales, les termes de second ordre et la vitesse navire transverse à la route devant la vitesse parallèle à la route on obtient:

$$u^n_c = u^n_{c0} + a u^n_n$$

$$v^n_c = v^n_{c0} + \alpha u^n_n$$

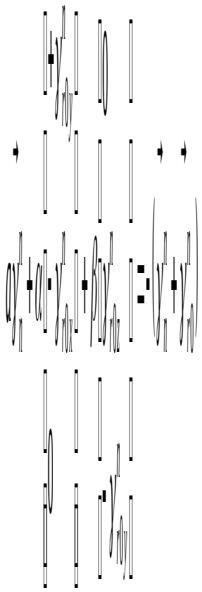
L'erreur sur l'amplitude affecte principalement la composante du courant parallèle au navire alors que l'erreur d'alignement dans le plan horizontal affecte la composante transverse du courant.

4.3.4.2Méthode de calcul des angles de désalignement et du facteur d'amplitude

La mesure du courant absolu est inconnue et variable. Afin d'éliminer cette inconnue du système, on dérive l'équation pour la vitesse absolue dans le repère navire et on néglige les accélérations du courant. On obtient alors l'équation:

$$M_{corr} \ \gamma^n_{r0} + (1 + a) \ \gamma^n_n = 0$$

que l'on peut réarranger en:



soit le système linéaire

$$GX = Y$$

où :

$$G = \begin{bmatrix} \gamma^n_{nx} & \gamma^n_{r0y} & 0 \\ \gamma^n_{ny} & -\gamma^n_{r0x} & \gamma^n_{r0z} \\ \gamma^n_{nz} & 0 & -\gamma^n_{r0y} \end{bmatrix}, \qquad X = \begin{bmatrix} a \\ \alpha \\ \beta \end{bmatrix}, \qquad Y = -\left(\gamma^n_{r0} + \gamma^n_n\right)$$

Dont la solution moindre carrés est donnée par la formule : $X = (G^T G)^{-1} G^T Y$

4.3.4.3ad2_headmis.m

Le programme effectue les étapes suivantes:

8. Calcul de la vitesse U_{r0} moyennée sur la couche de référence (ensemble de bins consécutifs défini par l'utilisateur (variable **iref**)) à partir des vitesses ADCP relatives au navire issues du fichier « processed » ADCP NetCDF (généré via ***ad2_ensmoy.m***)
9. Elimination des données dont les vitesse moyennes vérifient l'un au moins des critères suivants
   - Une seule des composantes (U ou V) est estimée.
   - Une des composantes de la vitesse absolue correspondante est supérieure à un seuil fixé (**vit_max_adcp**) par l'utilisateur.
   - La vitesse du navire (issue du fichier de navigation NetCDF) est inférieure à un seuil fixé par l'utilisateur (variable **vit_min**)
10. Calcul des accélérations relatives γ_{r0} et des accélérations navire γ_n.
11. Elimination des données dont les accélérations vérifient les critères suivants:
    - Les accélérations absolues sont supérieures à une valeur fixée par l'utilisateur (variable **acc_max_adcp**)
    - Le changement de cap est supérieur à un seuil fixé par l'utilisateur (variable **max_dcap**)
12. appel à ***ad2_headestim.m*** qui applique la méthode d'estimation de a, α, et β exposée précédemment.

4.4Etape 3: Calcul du courant absolu

Cette tâche est effectuée par le programme ***ad3_absolu.m***. A partir des fichiers « processed » ADCP NetCDF et des fichiers de navigation NetCDF, ce script détermine les vitesses absolues du courant.
Le calcul de la vitesse absolue est effectué en deux étapes: dans un premier temps, les vitesses relatives du courant par rapport au navire sont corrigées de l'erreur sur l'angle de l'ADCP par rapport à l'axe du navire (**angle_desalignement**), du facteur d'amplitude (**amplitude**) et de l'erreur sur le tangage (**pitch_biais**) déterminés via ***ad2_headmis.m*** et ***ad2_headestim.m***. Ceci est effectué via le script ***ad3_VGcor.m*** . Ensuite, la vitesse du navire est ajoutée à la vitesse du courant relative à l'ADCP afin d'obtenir les vitesses absolues. Le script ***ad3_creat_cdf_c.m*** crée en sortie un fichier « campagne » au format NetCDF dont la structure est présentée en annexe. Le script ***ad3_plt.m***, appelé par ***ad3_absolu.m***, permet de visualiser les vitesses absolues calculées.

ad3_VGcor.m dont le principe est le suivant :

Dans les fichiers « processed », la vitesse qui est mémorisée est la vitesse du courant relative au navire convertie en coordonnées géographiques. Cette conversion a été effectuée via la formule (cf. 4.3.2.3):

$$U_{ro} = MH * MR * MP * Moff * U_{ao}$$

où :

MH : correction du cap
MR : correction par rapport au roulis
MP : correction par rapport au tangage
Moff : correction de l'angle de l'ADCP par rapport à l'axe du navire
(correction permettant de passer du repère ADCP au repère du navire)
U_{ao} : vitesse de l'ADCP par rapport au navire dans le repère de l'ADCP mesurée par l'ADCP

S'il existe des désalignements (erreur sur l'angle de l'ADCP par rapport à l'axe du navire et erreur sur le tangage), la formule à appliquer est la suivante :

$$U_g = MH * MR * MP * MPdes * MHdes * Moff * U_a$$

où :

MPdes : correction due à l'erreur sur le tangage
MHdes : correction due à l'erreur sur l'angle de désalignement
U_a : « vrai » vecteur de la vitesse de l'ADCP par rapport au navire dans le repère de l'ADCP ($U_{ao} = A * U_a$ avec A : facteur d'amplitude)

ad3_VGcor.m détermine donc U_g («vraie » vitesse du courant relative au navire prenant en compte les éventuelles erreurs) à partir de U_{ro} (issu des fichiers « processed »). En notant MG = MH * MR * MP et Mcorr = MPdes * MHdes, on a : $U_g = A^{-1} * MG * Mcorr * MG^{-1} * U_{ro}$

4.5Définition des variables utiles aux étapes de traitement

Il est à noter que tous les scripts Matlab précédemment cités font appel au fichier ***conf.mat*** qui permet d'initialiser toutes les variables utiles au traitement (répertoires, valeurs seuils, ..etc.), à savoir :

- les variables indiquant si :
  - les informations campagne ont été renseignées (**info_camp_ok**=1 si oui, info_camp_ok = 0 sinon)
  - les informations indiquant si les informations utiles au bon déroulement de l'étape 1 ont été renseignées (**info_etape1_ok** = 1 si oui, info_etape1_ok = 0 sinon)
  - les informations indiquant si les informations utiles au bon déroulement de l'étape 2 ont été renseignées (**info_etape2_ok** = 1 si oui, info_etape2_ok = 0 sinon)
  - les informations indiquant si les informations utiles au bon déroulement de l'étape 3 ont été renseignées **(info_etape3_ok** = 1 si oui, info_etape3_ok = 0 sinon)
- ***nom_camp*** : nom de la campagne
- **nom_adcp** : type de l'ADCP
- **nom_navire** : nom du navire
- **nom_id_att** : identificateur de l'attitude dans les fichiers navigation RDI
- **adcp_depth** : profondeur des transducteurs
- les variables (**degrad** et **radeg**) permettant de transformer les degrés en radians et inversement.
- les variables **sens_roulis_interne** et **sens_tangage_interne**. Ces variables sont à 1 si le roulis et le tangage respecte la convention :
  - roulis positif lorsque le navire gîte à bâbord
  - tangage positif lorsque le navire lève le nez

  Elles valent -1 dans le cas contraire.
- la variable **exist_att_externe** qui vaut 1 si des données d'attitude externe existent, 0 sinon.
- l'année de référence (**ref_year**) et le jour julien d'origine associé (**jourjul_ref**) afin de visualiser les dates en jour julien décimal relativement à ce jour de référence.
  Dans le logiciel, les jours juliens débutent à minuit. Pour ce faire, le logiciel fait appel à ***jul_0h.m*** (resp. ***greg_0h.m***) pour transformer les jours grégoriens en jours juliens (resp. pour transformer les jours juliens en jours grégoriens). ***jul_0h.m*** fait appel à ***hms2h.m*** pour convertir les (heures,minutes,secondes) en heure décimale.
- le nom et l'année de la campagne (**nom_camp**, **cruise_year**).
- *les répertoires* :
  - des données ADCP brutes (**rep_adcp_bru**)
  - où seront stockés tous les fichiers ADCP NetCDF créés au cours du traitement (**rep_adcp_nc**) ainsi que les polynômes permettant de recaler l'heure ADCP par rapport au GPS (1 polynôme de nom pX_Y.pol par ensembles de fichiers de numéro X à Y).
    Plus exactement, sous ce répertoire, on trouvera :
    - le fichier journal
    - le fichier des vraies dérives (fichier « Derives »)

- les polynômes de recalage de l'horloge interne de l'ADCP (fichier ASCII et Postcript)
- le fichier Postcript associé au tracé de la différence entre les heures ADCP origines et les heures ADCP recalées par rapport au GPS.
- le fichier relatif à l'état d'avancement
- le fichier Postcript associé au tracé de ***ad2_headestim.m***
- le sous-répertoire **ncr** où seront stockés les fichiers ADCP NetCDF créés et modifiés par l'étape1
- le sous-répertoire **ncp** où seront stockés les fichiers ADCP NetCDF créés et modifiés par l'étape2
- le sous-répertoire **ncc** où seront stockés les fichiers ADCP NetCDF créés et modifiés par l'étape3
- le sous-répertoire **nce** où seront stockés les fichiers ADCP NetCDF créés par la partie exploitation
- le sous-répertoire **plot** où seront stockés les fichiers Postcripts générés par la partie exploitation

- où se situent les fichiers Bord d'attitude externe NetCDF (**rep_mru**)
- où seront créés tous les fichiers temporaires (**rep_concat**).

- l'identificateur des fichiers Bord d'attitude externe NetCDF (**id_mru**)
- l'identificateur des fichiers bruts ADCP (**id_adcp**)
- le nom (moins l'extension) du fichier de navigation TRINAV NetCDF (**nom_fic_nav**)
- *les constantes instrumentales* :
  - angle des faisceaux par rapport à la verticale (**angle_beam**)
  - angle de désalignement de l'ADCP par rapport à l'axe du navire (**angle_adcp**)
  - **range** : variable indiquant si les valeurs brutes ont été enregistrées en mode 'LOW' range ou 'HIGH' range. En fonction est déterminé le facteur d'échelle **fac_ech0**.
  - facteur d'échelle permettant de passer des valeurs faisceaux en vitesses (**fac_ech0**)
  - fréquence réelle de l'ADCP (**xfreq**)
- *les constantes de traitement* :
  - flag indiquant si l'utilisateur souhaite effectuer la correction due au roulis/tangage ou non (**icort_rt**).
    icor_rt = 1 ==> on effectue la correction
    icor_rt = 0 ==> on ne l'effectue pas.
  - nombre d'ensembles consécutifs à moyenner pour créer les fichiers « processed » ADCP NetCDF (**nb_ens_moy**)
  - indice précisant si on traite tout le fichier ou non (**trait_tout_fic**)
  - nombre de moyennes à calculer par fichiers (**nb_moyenne**).
    nb_moyenne = N ==> ***ad2_ensmoy.m*** calcule N moyennes par fichier
    nb_moyenne = 0 ==> ***ad2_ensmoy.m*** traite tout le fichier.
  - erreur maximale acceptée sur l'erreur de vitesse ainsi que sur la vitesse verticale (**err_max** utilisée dans ***ad2_calmoy.m)***
  - vitesse ADCP maximale tolérée (**vit_max**)

- variable (**n_std**) utile au test d'écart à la moyenne utilisée dans ***ad2_Net_dat.m.***

  si (U - mean(U)) < n_std * ecart_type) alors U OK.
- variable (**nit_max**) indiquant le nombre d'itérations à effectuer sur le test d'écart à la moyenne dans ***ad2_Net_dat.m***
- le pourcentage minimum (**pg_min**) de données correctes qu'il faut pour calculer une moyenne. Sur 'X' données, on ne garde la moyenne que si celle-ci peut être calculée avec au moins (**pg_min** % * X) de données validées.
- la couche de référence (**iref**) utilisée dans ***ad2_adnav.m***.
- les variables utilisées (en cm/s) dans ***ad2_headmis.m*** :
  - seuil maximal des vitesses du courant (**vit_max_adcp**)
  - seuil maximal des accélérations du courant (**acc_max_adcp**)
  - seuil minimal pour la vitesse du navire (**vit_min**)
  - seuil maximal pour le changement de cap (**max_dcap**)
  - variable (**fac_ect**)
  - variable (**ictmax**). L'angle de désalignement est estimé de manière itérative par les moindres carrés. **ictmax** représente le nombre maximal d'itérations.
  - variable (**estim_amplitude**) indiquant si l'utilisateur souhaite estimer juste l'angle de désalignement ou aussi l'amplitude.
- les variables utilisées dans ***ad3_absolu.m*** :
  - l'erreur sur l'angle de désalignement (**angle_desalignement**) estimée par ***ad2_headmis.m***/***ad2_headestim.m***
  - l'amplitude (**amplitude**) estimée par ***ad2_headmis.m***/***ad2_headestim.m***
  - l'erreur sur l'assiette du navire (sur le tangage) (**pitch_biais**) estimée par ***ad2_headmis.m/ad2_headestim.m***.

.

4.6Exploitation des vitesses absolues du courant

Cette étape prend en entrée le fichier NetCDF campagne créé par ***ad3_absolu.m***. Elle permet de mettre un flag sur les données, de créer des fichiers NetCDF section et des fichiers NetCDF station. Elle permet également d'effectuer des tracés de contourage, de vecteurs et de profils.

4.6.1 Nettoyage des données

net_vit_cour.m part du fichier campagne généré par ***ad3_absolu.m*** et comportant, entre autres, les vitesses U, V et W du courant. Ce script permet de 'flagguer' les bonnes données à 0, les données douteuses à 1 et les mauvaises données à une valeur supérieure ou égale à 2. Ce script commence par mettre tous les flags à 0.
Ensuite, le 1er 'flagguage' est basé sur un test d'écart à la médiane. Les vitesses (U, V) de chaque cellule sont comparées à la médiane MU1 et MV1 des 'X' cellules encadrant (horizontalement) de part et d'autre cette cellule en cours. Sur chaque segment de longueur (2X+1), on calcule les médianes MU2 et MV2 de |U-MU1| et |V-MV1|. Le point central Uc et Uv des segments de U et V est alors 'flaggué' à 2 si |Uc-MU1| (ou |Vc-MV1|) est supérieur à 1.15*X2*MU2 (ou 1.15*X2*MV2). Il est 'flaggué' à 1 si |Uc-MU2| (ou |V-MV2|) est supérieur à X2*MU2 (ou X2*MV2). Sinon, il est 'flaggué' à 0. X et X2 sont fixés par l'utilisateur.
Le 2nd 'flagguage' est basé sur le cisaillement différentiel. Les cellules dont l'une des composantes horizontales a un cisaillement vertical différentiel supérieur à une valeur V fixée par l'utilisateur sont mises à 3. Un histogramme des cisaillements est affiché pour aider l'utilisateur à choisir sa valeur V. Le nettoyage est particulièrement utile pour les points en fin de profil.
Le 3nd 'flagguage' est basé sur la vitesse verticale. Les cellules dont la vitesse verticale ou l'écart-type de l'erreur est supérieure à une valeur fournie par l'utilisateur sont mises à 4.
Le 4ième 'flagguage' est basé sur les vitesses absolues. Les cellules dont l'une des vitesses absolues horizontales est supérieure à 4m/s sont mises à 5.
Le 5ième 'flagguage' est basé sur l'existence des données. Les cellules sans données sont mises à 6.
Le 6ième 'flagguage' est basé sur la détection du fond. Les cellules sous le fond sont mises à 7 via l'utilisation du 'range' fourni par le bottom-ping.
Enfin, en ne considérant que les bonnes données, les singletons et doublons isolés sont mis à 1. De même pour les profils dont moins de la moitié des données de la couche de référence sont bonnes. La couche de référence est fournie par l'utilisateur via le script ***demande_iref.m***.

Pour un fichier campagne donné, ***flag_8.m*** permet de flagguer à très mauvais (flag 8) des données comprises entre 2 dates fournies par l'utilisateur.

efface_net_vit_cour.m part également du fichier campagne généré par ***ad3_absolu.m***. Ce script annule l'effet des nettoyages ci-dessus et force les données à bonnes via la remise à 0 des flags.

4.6.2Comparaison vitesse navire/vitesse du courant

cmp_vitnav_vitcour.m permet de comparer les vitesses du navire aux vitesses du courant. L'utilisateur peut vérifier que les deux ne sont pas liées. Dans le cas contraire, si

les vitesses du courant sont affectées par les accélérations et/ou décélérations du navire, l'utilisateur peut modifier dans l'environnement de travail l'angle de désalignement et/ou l'amplitude jusqu'à ce que les vitesses du courant ne soient plus 'corrélées' avec les vitesses du navire.

4.6.3Comparaison vitesse du courant en route/vitesse du courant en station

cmp_sta_route.m permet de comparer les vitesses du courant en station avec les vitesses du courant en route. Si les vitesses en station ne sont pas représentatives des vitesses en route, cela signifie que la vitesse du navire n'est pas totalement éliminée. L'utilisateur peut jouer sur l'angle de désalignement et/ou l'amplitude pour y remédier. Si la vitesse verticale en route n'est pas centrée sur 0, l'utilisateur peut alors jouer sur le paramètre **pitch_biais** qui représente l'erreur sur l'assiette du navire (sur le tangage).

4.6.4Information fichier campagne

Diverses opérations peuvent être effectuées sur le fichier campagne.

•1 **Ajout de la marée (cf. *ajout_maree.m*/ *calcul_maree.m*)**

Ce script permet de calculer la marée pour chaque point du fichier campagne. Ce script fait appel à un exécutable C rapatrié via le WEB. Le calcul de la marée est basé sur un modèle global de marée (cf. annexes). Ce modèle permet de calculer les vitesses (U, V) et la hauteur (H) de la marée pour toutes les longitudes [0 360] et les latitudes [-80 70]. Néanmoins, près des côtes, H n'est pas déterminée. Et il est à noter que pour une liste de positions données, ***dès que l'on atteint une position pour laquelle la hauteur de marée ne peut pas être calculée, le programme C de calcul de H indique que la position est 'out of range' et s'arrête (le calcul de la hauteur de marée des positions suivantes n'est donc pas effectué).***

A l'issue de cette étape, dans le fichier campagne, les variables suivantes sont rajoutées :

- o1 U_maree, V_maree : vitesses U et V associées à la marée en cm/s
- o2 H_maree : Hauteur de la marée en m
- o3 U_corrigee et V_corrigee : Vitesses U et V du ficher campagne corrigées de la marée.

•2 **Informations générales** (***info_fic_camp.m***)

Ceci permet de visualiser l'état du fichier campagne sélectionné par l'utilisateur. Cet état comprend :

- le nom du fichier
- les estimations des erreurs considérées pour calculer les vitesses absolues du courant et créer le fichier, à savoir :
  - erreur sur l'angle d'alignement de l'ADCP par rapport au navire
  - erreur sur l'assiette (tangage) du navire
  - erreur sur l'amplitude
- la moyenne de la vitesse verticale pour les données de flags 0 afin de vérifier l'assiette du navire
- le nombre d'ensembles consécutifs moyennés par valeur (cf. ***ad2_ensmoy.m***)
- le nombre de points pour chacun des flags créés par ***net_vit_cour.m***

- **Tracés**
  - **Tracé de la dérive (*trace_derive_fic_camp.m*)**
    Ce script permet de tracer la dérive de l'horloge ADCP en fonction du temps pour toute la durée de la campagne. Si l'utilisateur le souhaite, il a la possibilité de tracer un polynôme associé à cette dérive.
  - **Tracé de variable 1D (cf. *trace_fic_camp.m*)**
    La liste de toutes les variables à 1 dimension du fichier campagne est présentée à l'utilisateur. Celui-ci peut en choisir 4 au maximum. S'en suit une figure comprenant 1 tracé par variable sélectionnée. Les variables sont tracées selon le temps.
  - **Tracé de variable 2D (cf. *trace_fic_camp.m*)**
    La liste de toutes les variables à 2 dimensions du fichier campagne est présentée à l'utilisateur. Celui-ci peut en choisir 4 au maximum. S'en suit une figure comprenant 1 tracé par variable sélectionnée. Les variables sont tracées selon le temps en X et selon la profondeur en Y.
- **Filtrage (cf. *appel_filtre.m*, *filtre.m*)**
  Cette étape permet de filtrer les vitesses U et V, corrigées ou non de la marée, du fichier campagne. L'utilisateur a la possibilité de choisir les flags de qualité à prendre en compte pour le filtrage. Généralement, on prend les données correctes (flag 0) et éventuellement les données douteuses (flag 1).Le filtre est basé sur 3 points consécutifs.
  Soient 3 points A, B, C consécutifs:
  - Cas général : Le résultat du filtre appliqué à B est : 0.25*A+0.5*B+0.25*C
  - Cas des bords :
    - Le résultat du filtre appliqué à A est : 2/3*A + 1/3*B
    - Le résultat du filtre appliqué à C est : 1/3*B + 2/3*C

  L'utilisateur peut effectuer un filtrage :
  - Vertical : selon la profondeur
  - Horizontal : selon le temps
  - Vertical et horizontal

  Le fichier résultat créé comporte :
  - Les vitesses filtrées pour toutes les positions du fichier fourni en entrée
  - Un flag de qualité :
    - 0 : si les vitesses ont été filtrées avec au moins 2 données correctes (flagguées à 0)
    - 1 : si les vitesses ont été filtrées avec au moins 2 données douteuses (flagguées à 1)
    - 1 : si les vitesses ont été interpolées. C'est le cas des éventuels trous du fichier initial qui ont été comblés via le filtrage.
    - 6 : si les vitesses n'ont pas être déterminées.

4.6.5Création de fichiers

cree_sec.m permet à l'utilisateur de créer un fichier section NetCDF. Il part du fichier campagne généré par ***ad3_absolu.m*** et d'un fichier section ASCII fourni par l'utilisateur. Ce fichier doit avoir une extension « **.sec** » et comporter une ligne par section comprenant :

- le numéro de section
- la date de début de section (JJ/MM/AAAA HH:MIN:SEC)

- la date de fin de section (JJ/MM/AAAA HH:MIN:SEC).

Via le script ***demande_distance.m***, l'utilisateur indique sur quelle distance (en km) il souhaite moyenner les sections. Le fichier NetCDF section *résultat comporte pour chaque section un ensemble de points pour lesquels les vitesses du courant (corrigées ou non de la marée selon le choix de l'utilisateur) sont moyennées sur cette distance*. Selon le choix de l'utilisateur, les fichiers NetCDF section prennent en compte ou non les données associées aux stations. Le fichier est généré sous le sous-répertoire **nce** du répertoire des fichiers NetCDF défini par l'utilisateur. Son nom a la structure suivante : *section_X_yy.nc* où X est la distance et yy correspond à 'st' ou 'at' selon que les stations sont ou non considérées. Ainsi, le fichier section_2_st.nc correspond à un fichier section où les sections sont moyennées environ tous les 2 kms sans prendre en compte les stations.

cree_sta.m permet à l'utilisateur de créer un fichier station NetCDF. Il part du fichier campagne généré par ***ad3_absolu.m*** et d'un fichier station ASCII fourni par l'utilisateur. Ce fichier doit avoir une extension « **.sta** » et comporter une ligne par station comprenant :

- le numéro de station
- la date de début de station (JJ/MM/AAAA HH:MIN:SEC)
- la date de fin de station (JJ/MM/AAAA HH:MIN:SEC)

Via le script ***demande_duree.m***, l'utilisateur indique sur quelle durée (en seconde) chaque station doit être moyennée. Le fichier NetCDF station *résultat comporte pour chaque station un ensemble de points pour lesquels les vitesses du courant sont moyennées sur cette durée*. L'utilisateur a la possibilité de moyenner chaque station sur toute la durée de la station. Le fichier est généré sous le sous-répertoire **nce** du répertoire des fichiers NetCDF défini par l'utilisateur. Son nom a la structure suivante : *station_X.nc* où X est la durée. Ainsi, le fichier station_600.nc correspond à un fichier station comprenant un point par station toutes les 10 minutes. Le fichier station_0.nc correspond au fichier NetCDF station comprenant un seul point par station moyenné sur toute la durée de la station.

La structure des fichiers NetCDF créés par ***cree_sec.m*** et ***cree_sta.m*** est présentée en annexe.

4.6.6Tracés

contour_sec.m permet d'effectuer un contourage de section en fonction de la latitude ou de la longitude. L'utilisateur sélectionne un fichier NetCDF section généré via ***cree_sec.m***. Toutes les sections s'affichent et l'utilisateur doit cliquer sur la section qui l'intéresse. Un premier tracé par défaut est effectué. Via le script ***demande_contour.m***, l'utilisateur peut définir son propre tracé en imposant la valeur minimale et maximale à contourer ainsi que le pas de contourage.

contour_sta.m permet d'effectuer un contourage de station en fonction du temps. L'utilisateur sélectionne un fichier NetCDF station généré via ***cree_sta.m***. Le contourage étant effectué pour chaque station en fonction du temps, le choix d'un fichier station moyenné sur toute la durée de la station est impossible.

vecteur_sec.m permet de tracer les vecteurs de vitesses de courant pour une profondeur donnée à partir du fichier NetCDF section sélectionné par l'utilisateur. L'utilisateur peut tracer sur le même graphique les vecteurs associés à une ou plusieurs sections. Il sélectionne les sections qui l'intéressent en cliquant dessus. La fin de la sélection est déterminée par un clic sur le mot 'Longitude' du graphe représentant toutes les sections du fichier. Via le script ***demande_cadre.m***, l'utilisateur est invité à saisir les informations

relatives au cadre géographique du tracé. Un premier tracé par défaut est effectué. Via le script ***demande_echelle.m***, l'utilisateur peut jouer sur l'échelle des vecteurs.

vecteur_sta.m permet de tracer les vecteurs de vitesses de courant pour une profondeur donnée à partir du fichier NetCDF station sélectionné par l'utilisateur. Via le script ***demande_cadre.m***, l'utilisateur est invité à saisir les informations relatives au cadre géographique du tracé. Un premier tracé par défaut est effectué. Via le script ***demande_echelle.m***, l'utilisateur peut jouer sur l'échelle des vecteurs.

profil_sta.m permet de tracer les profils de vitesses U, V et W en fonction de la profondeur pour chaque station. L'utilisateur sélectionne un fichier NetCDF station généré par ***cree_sta.m***. Le choix d'un fichier station moyenné sur une durée autre que la durée de la station n'est pas autorisé.

profil_deb_fin_sta.m permet de tracer les profils de vitesses U, V et W en fonction de la profondeur pour le début et la fin de chaque station. L'utilisateur sélectionne un fichier NetCDF station généré par ***cree_sta.m***. Le choix d'un fichier station moyenné sur toute la durée de la station n'est pas autorisé.

4.6.7Création de fichiers pour l'inversion

cree_fic_inv.m permet à l'utilisateur de créer 1 fichier de données par jour à partir d'un fichier section (issu à partir d'un fichier campagne généralement filtré). Ces fichiers journaliers seront utilisés dans le programme d'inversion avec filtre de kalman. Ils contiennent notamment la position des données, les vitesses U et V en cm/s et une erreur sur les données fixée à 5 cm/s.

5.L'interface utilisateur

Afin de faciliter l'utilisation des divers scripts Matlab mentionnés précédemment, une interface graphique a été développée. Celle-ci repose sur divers scripts succinctement présentés ci-après.

5.1L'interface principale

Pour appeler le logiciel de traitement de données d'ADCP de coque, l'utilisateur doit appeler l'interface principale en tapant ***cascade*** à partir du prompt Matlab.
Ensuite, par simple clic, il peut accéder aux différentes étapes du traitement (cf. 4.1.5).
L'interface principale est présentée ci-après.

La fonction 'Aide' affiche un message rappelant la structure des fichiers « **.sec** » et « **.sta** » à fournir par l'utilisateur pour la partie 'exploitation'.

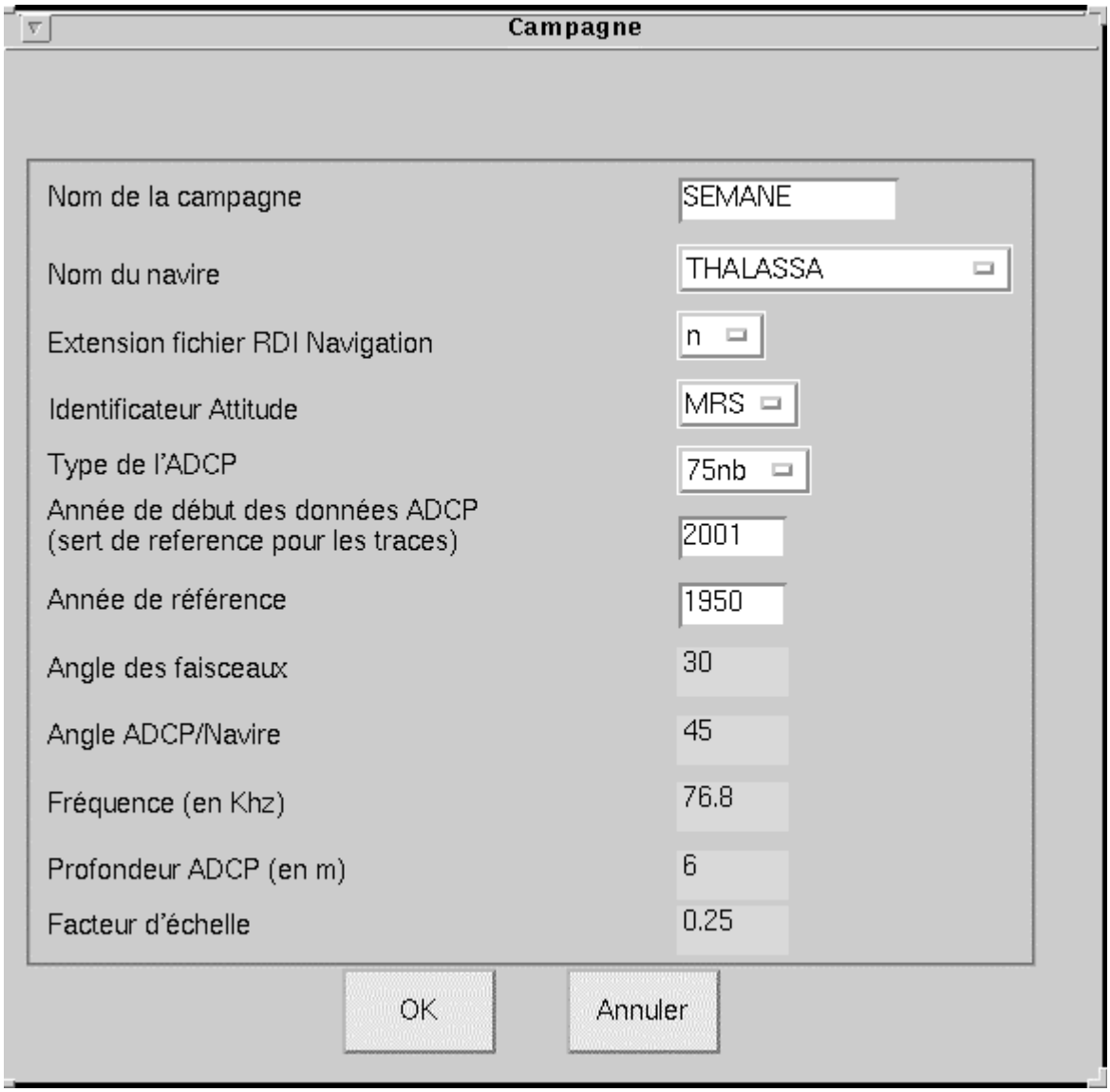
5.2Environnement de travail : Définition - Sauvegarde - Chargement

Avant toute chose, l'utilisateur doit définir son « Environnement de travail ». Il doit impérativement renseigner les « Informations Campagne ». Ensuite, avant chaque étape, il doit saisir les informations utiles au bon déroulement de l'étape en question.

- A l'exécution, si aucun fichier ***conf.mat*** n'existe sous le répertoire courant, une configuration de base est chargée à partir du fichier ***conf.mat*** situé sous le répertoire de base /home3/doelan/chemon/soft_adcp/adcp_lpo/global. Ces informations, après modifications éventuelles et validation, sont ensuite mémorisées dans le fichier ***conf.mat*** sous le répertoire courant (à partir duquel Matlab a été lancé). Toutes les autres étapes du traitement sont basées sur ce fichier ainsi que sur le fichier ***sauv_etat.mat*** (cf. 4.1.3).

La configuration de l'environnement de travail s'effectue via l'appel aux interfaces : ***info_camp.m, info_etape1.m, info_etape2.m, info_etape2_2.m*** et ***info_etape3.m.***

info_camp.m



Remarques :

•1 L'extension du fichier RDI navigation (lettre 'n') est en minuscule ou en majuscule selon les campagnes. Cette extension RDI diffère sans raison apparente.

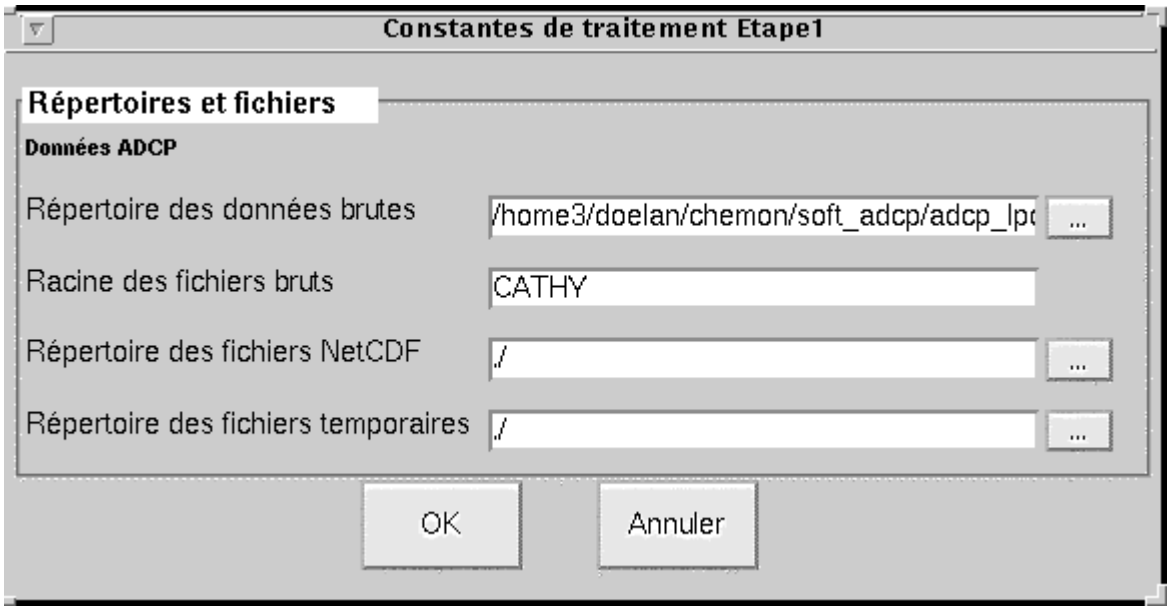
•2 L'identificateur Attitude correspond aux 3 lettres avant les données d'attitude dans le message ADCP temps réel des fichiers navigation RDI. On peut avoir : HDM, MRU, MRS, …etc.

- **La profondeur ADCP, l'angle des faisceau** et **l'angle de l'ADCP** par rapport au navire ne sont pas saisis par l'utilisateur mais **fonction du navire et de l'ADCP**. Pour ce faire, CASCADE est basé sur le fichier **info_navire**.
- **La fréquence** est **fonction du type d'ADCP.** Pour ce faire, CASCADE est basé sur le fichier info_adcp.
- **Le facteur d'échelle** est **fonction du type d'ADCP** et du '**velocity_range**'. Le 'velocity_range' est directement lu dans les fichiers bruts ADCP RDI. En conséquence, CASCADE détermine le facteur d'échelle via le fichier info_adcp. Il est à noter que le programme ***ad2_ensmoy.m*** lit la valeur du « range » dans les fichiers NetCDF ADCP. Si

ce « range » lu diffère du « range » préalablement initialisé, un message apparaît à l'écran et le facteur d'échelle est redéterminé en conséquence (via la relecture du fichier info_adcp).

- L'année de référence (**ref_year**) permet de déterminer le jour julien de référence (qui est le 01 janvier de **ref_year**). Ce jour julien est utilisé pour la création des fichiers campagne, section et station. Dans ces fichiers, le jour julien mémorisé est le jour julien relatif à ce jour julien de référence. Afin de ne pas avoir de jour julien négatif, l'année de référence ne peut pas être supérieure à l'année de début des données ADCP (**cruise_year**). L'année de début des données ADCP sert à visualiser la date relativement au 1er janvier de cette année dans divers tracés.

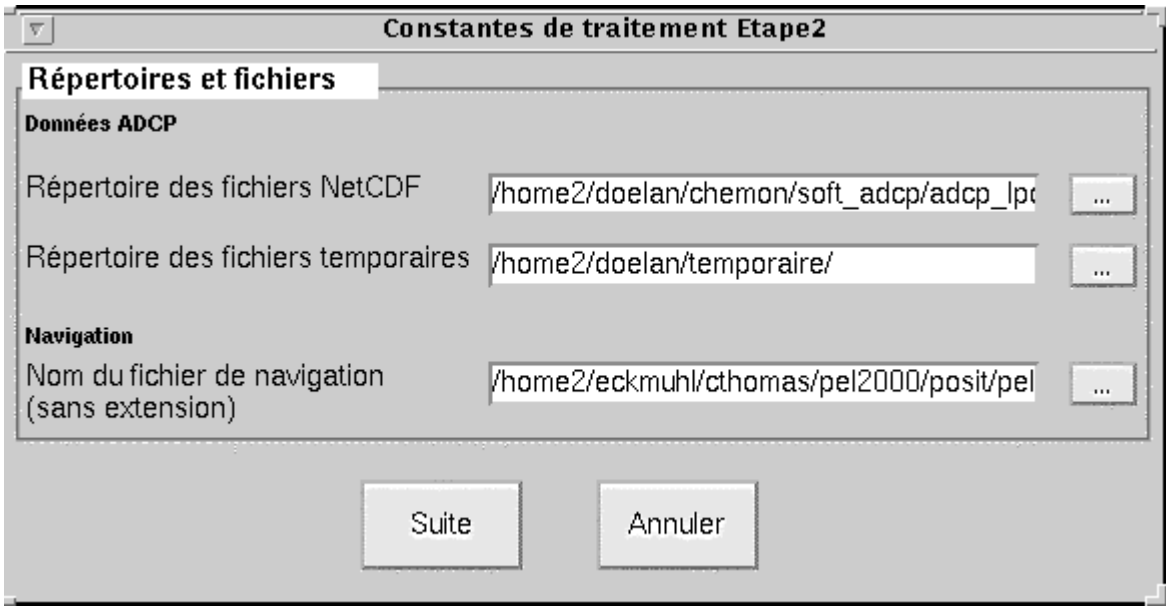
info_etape1.m



Remarques :

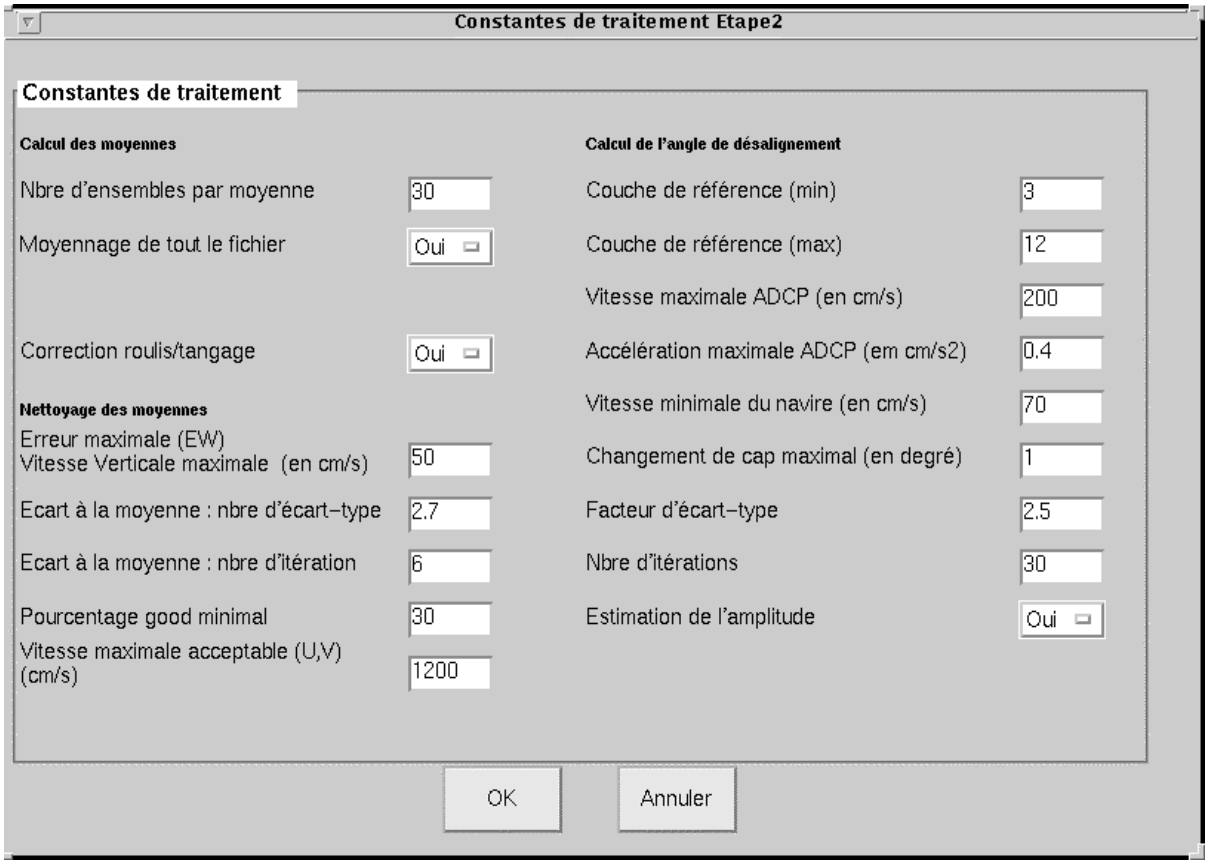
- Les fichiers ASCII et Postcript (fichiers journal, « Derive », polynômes et état d'avancement) sont générés sous le répertoire des fichiers NetCDF donné par l'utilisateur.
- Les fichiers NetCDF ADCP générés par le logiciel sont créés dans des sous-répertoires du répertoire des fichiers NetCDF donné par l'utilisateur.
  - les fichiers issus de ***ad1_cree_nc.m*** sont créés sous le sous-répertoire **ncr**.
  - les fichiers issus de ***ad2_ensmoy.m*** sont créés sous le sous-répertoire **ncp**.
  - les fichiers issus de ***ad3_absolu.m*** sont créés sous le sous-répertoire **ncc**.
  - les fichiers générés par la partie 'exploitation' sont créés sous le sous-répertoire **nce**.
  - les fichiers Postscript générés par la partie 'exploitation' sont créés sous le sous-répertoire **plot**.

info_etape2.m



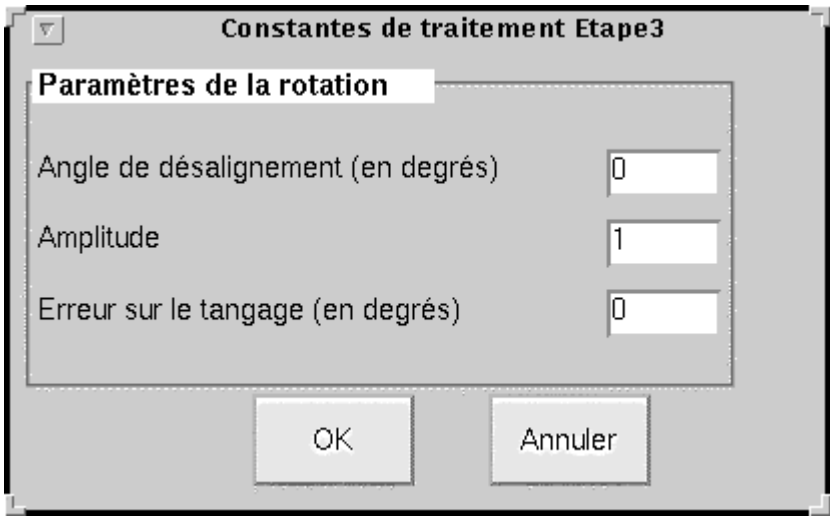
Remarque :
Si l'utilisateur souhaite utiliser la navigation temps réel pour le traitement des données, il ne doit rien entrer pour le fichier de navigation.

info_etape2_2.m



Les valeurs ci-dessus, chargées par défaut par le logiciel CASCADE, sont celles généralement utilisées. Néanmoins, s'il le souhaite, l'utilisateur peut les modifier.

info_etape3.m



Les 2 dernières interfaces permettent à l'utilisateur de saisir les constantes de traitement utiles aux étapes 2 et 3, constantes explicitées précédemment.

Pour chaque interface, le bouton « OK » permet de sauvegarder ces informations dans le répertoire courant sous un fichier nommé ***conf.mat*** via le script ***sauv_var.m***.
Par la suite, pour chacune des diverses étapes, le fichier ***conf.mat*** est automatiquement chargé via le script ***charge_conf.m***.

5.3Appel aux différentes étapes

Les callbacks de la plupart des menus de ***cascade.m*** ne sont en fait que de simple appel aux étapes du traitement détaillées dans les paragraphes 4.2 à 4.4. Ces appels se font via le script ***appel.m*** qui permet :

- de demander à l'utilisateur de saisir les numéros de fichier sur lesquels il souhaite travailler via l'appel à ***demande_numfic.m***.
- de charger l'environnement de travail via l'appel à ***charge_conf.m***.
- de lancer le script ou fonction en paramètre.

Ainsi, les **sous-menus de l'étape1** font respectivement appel à :

- appel('ad1_cree_nc')
- appel('ad1_estim_derive_TT')
- appel('ad1_cal_vraie_derive')
- appel('ad1_cal_poly')
- appel('ad1_corr_heure')
- appel('ad1_trace_derive')
- appel('ad1_corr_att')
- appel('ad1_avancement')

ad1_trace_derive et ***ad1_avancement*** ne sont pas décrits dans le paragraphe 4 car ce ne sont pas des scripts de traitement des données. Ces scripts sont en fait des scripts d'information.

ad1_trace_derive.m :

Ce script permet de visualiser la dérive qui a été appliquée sur les fichiers en cours en traçant le graphe :

f(temps_ADCP_origine) = temps_ADCP_origine - Temps_ADCP_corrigé_du_GPS

Le fichier Postcript associé est automatiquement généré sous le répertoire des fichiers ADCP NetCDF. Ce fichier se nomme : *derive_X_Y.ps*, X et Y étant les numéros du premier et du dernier fichier à considérer.

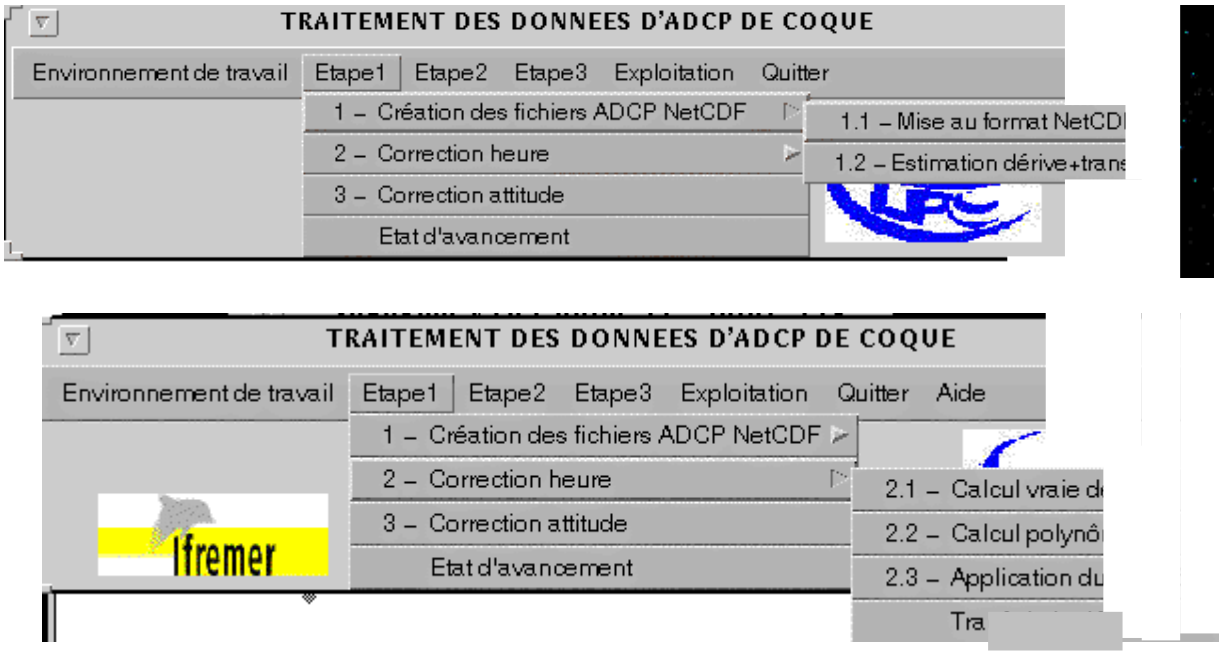
ad1_avancement.m :

Ce script permet de visualiser l'état d'avancement du traitement dans l'étape1 pour les numéros de fichier entrés par l'utilisateur. Cet état d'avancement peut être sauvegardé sous le répertoire des fichiers ADCP NetCDF.

Exemple de sortie :

Fichier Nb_ens date_début date_fin dérive_estim Corr_heure Corr_attitude

camb011 20581 1998/04/18 16:51:54 1998/04/19 11:43:49 0.868 Oui Non
...
camb020 10303 1998/04/23 12:01:05 1998/04/23 23:43:49 NaN Non Non



Remarque :

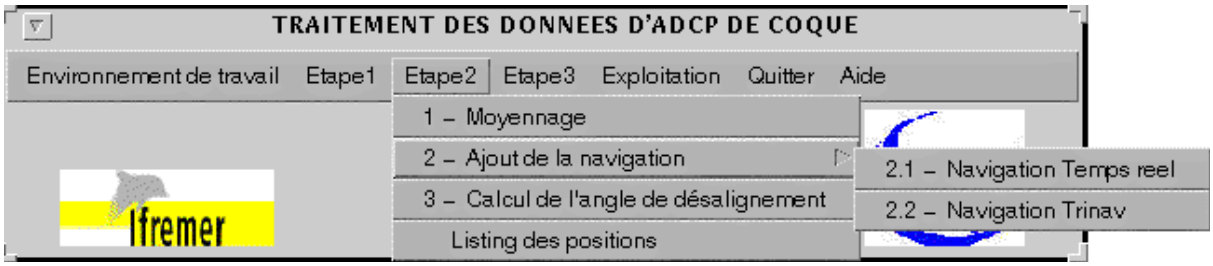
- Les parties « Mise au format NetCDF » et « Estimation dérive+transfert » peuvent s'effectuer en routine au fur et à mesure de l'acquisition. En effet, elles sont basées uniquement sur les fichiers bruts ADCP (issus du logiciel « Transect ») et ne requièrent aucun autre fichier extérieur.

Les **sous-menus de l'étape2** font respectivement appel à :

- appel('ad2_ensmoy')
- appel('ad2_adnav') avec la variable '**utilise_trinav'** qui indique si on souhaite ajouter la navigation temps réel ou la navigation TRINAV.
- appel('ad2_headmis')
- appel('ad2_listpos')

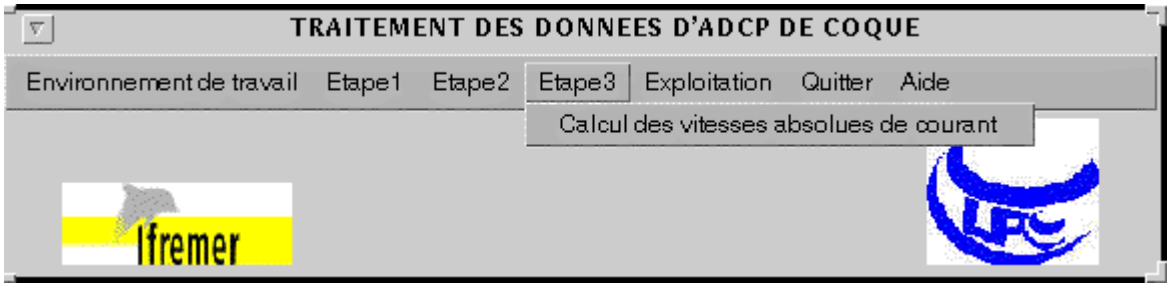
ad2_listpos.m n'est pas un script de traitement des données proprement dit. Il permet simplement de générer un fichier ASCII comportant la date en jour julien, la date

grégorienne associée et la position (latitude, longitude) à partir de fichiers NetCDF ‘processed’.



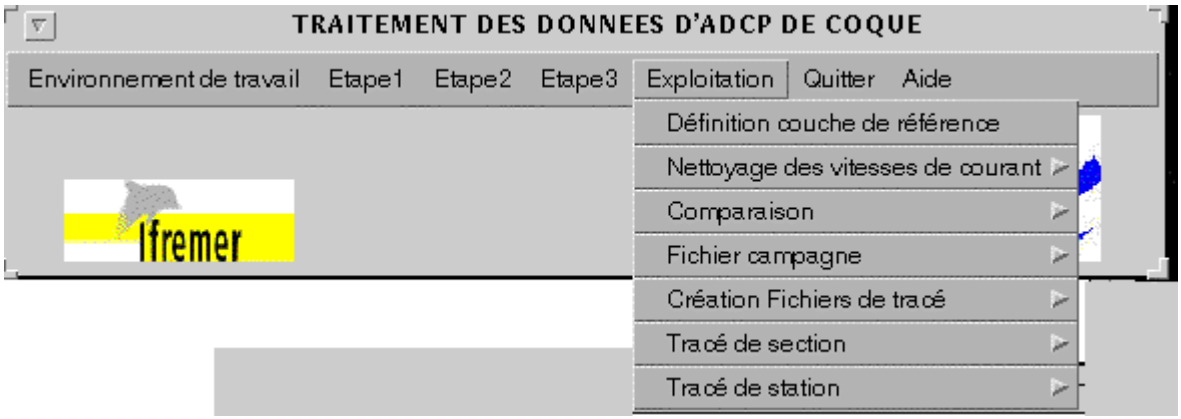
Le **sous-menu de l’étape3** fait appel à :

- appel(‘ad3_absolu’)



Le **sous-menu de la partie exploitation** fait appel à :

- demande_iref
- net_vit_cour
- efface_net_vit_cour
- cmp_vitnav_vit_cour
- cmp_sta_route
- ajout_maree
- info_fic_camp
- trace_derive_fic_camp
- trace_fic_camp
- appel_filtre
- cree_sec
- cree_sta
- contour_sec
- vecteur_sec
- contour_sta
- vecteur_sta
- profil_sta



5.4Autres scripts et interfaces

Les scripts et interfaces présentés ci-après sont utilisés pour afficher des messages d'informations aux utilisateurs, pour effectuer des tests sur les valeurs saisies ou encore pour demander à l'utilisateur divers renseignements.

appel.m
Ce script permet de charger les variables et constantes utiles au traitement (via ***charge_conf.m***) puis de demander à l'utilisateur sur quels numéros de fichiers il souhaite travailler (via ***demande_numfic.m***).

appel_filtre.m
Ce script permet d'appeler filtre.m de manière différente selon le type de filtrage (vertical ou horizontal).

Cal_vit_ref.m
Ce script permet de calculer la vitesse moyenne sur une couche de référence donnée.

const_adcp.m
Ce script définit toutes les constantes, à savoir :

- le répertoire de base où se situent :
  - le fichier ***conf.mat*** pris par défaut si le fichier conf.mat n'existe pas dans le répertoire courant
  - le fichier ***sauv_etat.mat***
  - le fichier ***info_adcp*** contenant 1 ligne par type d'ADCP
  - le fichier ***info_navire***

  Ces fichiers sont explicités dans le paragraphe 4.1.3.
- l'année minimale autorisée
- l'année en cours (déterminée automatiquement)
- les couleurs, textes et messages utilisés dans les diverses interfaces.

choix_fic.m
Ce script permet à l'utilisateur de sélectionner un fichier. Le fichier est mémorisé sans son extension.

choix_repertoire.m
Ce script permet à l'utilisateur de sélectionner un répertoire.

demande_date.m
Ce script permet à l'utilisateur de fournir les dates entre lesquelles les données d'un fichier campagne doivent être flaguées très mauvaises (flag 8).

demande_cadre.m
Ce script est appelé par ***vecteur_sec.m*** et ***vecteur_sta.m***. Il permet à l'utilisateur de saisir les coordonnées géographiques (Latitude et Longitude minimale et maximale) pour les tracés de vecteurs.

demande_contour.m
Ce script est appelé par ***contour_sec.m***. Il permet à l'utilisateur de saisir la valeur minimale M1 et maximale M2 à contourer ainsi que le pas P. Les valeurs contourées sont alors M1, M1+P, M1+2P, ..., M2.

demande_decal.m
Ce script est appelé par ***ad1_cal_vraie_derive.m***. Il permet à l'utilisateur d'indiquer le nombre de secondes à ajouter à l'heure ADCP afin que les 2 graphes attitude=f(Tps GPS) et attitude=f(tps ADCP) se superposent et que l'heure ADCP coïncide avec l'heure GPS.

demande_degre.m
Ce script est appelé par ***ad1_cal_poly.m***. Il permet à l'utilisateur de saisir le degré du polynôme à calculer.

demande_distance.m
Ce script est appelé par ***cree_sec.m***. Il permet à l'utilisateur d'indiquer sur quelle distance (en km) il souhaite moyenner les données (U, V, W, ...etc.) afin de créer le fichier NetCDF section.

demande_douteux.m
Il est appelé par plusieurs des scripts de la partie exploitation. Il permet à l'utilisateur d'indiquer s'il souhaite prendre en compte les données flagguées à douteuses ou non.

demande_duree.m
Il est appelé par ***cree_sta.m***. Il permet à l'utilisateur de saisir sur quelle durée (en seconde) il souhaite moyenner chaque station afin de créer les fichiers NetCDF station.

demande_echelle.m
Il est appelé par ***vecteur_sec.m*** et ***vecteur_sta.m***. Il permet à l'utilisateur de saisir un facteur d'échelle pour les tracés de vecteurs.

demande_flag.m
Il est appelé par cree_sec.m. Il permet à l'utilisateur de sélectionner les flags à prendre en compte dans le fichier section à créer.

demande_info_contour_sec.m
Il est appelé par ***contour_sec.m***. Il permet à l'utilisateur de préciser les caractéristiques du tracé (type de tracé : contourage ou image (pcolor), axe des X : latitude ou longitude).

demande_info_contour_sta.m
Il est appelé par ***contour_sta.m***. Il permet à l'utilisateur de préciser s'il souhaite un tracé de contourage ou une image (pcolor). L'axe des X sera le temps.

demande_lon_lat.m
Il est appelé par ***cmp_sta_route.m***. Il permet à l'utilisateur de préciser s'il souhaite un tracé en longitude ou en latitude.

demande_nettoie.m
Il est appelé par ***net_vit_cour.m***. Il permet à l'utilisateur de saisir les valeurs utilisées lors des diverses phases de 'flagguage' des données (cf. 4.6.1-***net_vit_cour.m***).

demande_nom_diary.m
Ce script est appelé par ***cascade.m*** afin de demander à l'utilisateur un nom pour le fichier journal. Le fichier journal est un fichier dans lequel MATLAB mémorise toutes les informations affichées dans la fenêtre principale au cours d'une session.

demande_nom_etat.m
Ce script est appelé par ***ad1_avancement.m*** afin de demander à l'utilisateur un nom pour le fichier de sauvegarde de l'état d'avancement.

demande_nom_fic_camp.m

Ce script est appelé par ***ad3_absolu.m*** afin de permettre à l'utilisateur de donner un nom au fichier campagne où les vitesses absolues seront stockées (cf. annexe : Structure NetCDF du fichier campagne)

demande_numfic.m
Ce script permet à l'utilisateur de saisir les numéros de fichiers à traiter.

demande_pcolor.m
Ce script est appelé par ***contour_sec.m***. Il permet à l'utilisateur de saisir la valeur minimale et maximale à colorier.

demande_postcript.m
Ce script est appelé par plusieurs des scripts d'exploitation. Il permet à l'utilisateur d'indiquer s'il souhaite générer les fichiers Postcript associés aux divers tracés en cours.

demande_profondeur.m
Ce script est appelé par ***vecteur_sec.sec*** et ***vecteur_sta.m***. Il permet à l'utilisateur de saisir sur quelle tranche (profondeur minimale et maximale en m) les vecteurs à tracer doivent être moyennés.

demande_trace_cap.m
Il permet à l'utilisateur de préciser s'il souhaite visualiser le cap entre -180 et 180 ou entre 0 et 360.

demande_zoom.m
Ce script est appelé par ***ad1_cal_vraie_derive.m***. Il permet à l'utilisateur d'indiquer pour quelles dates il souhaite visualiser les graphes attitude=f(Tps GPS) et attitude=f(Tps_ADCP). Plus exactement, il demande à l'utilisateur de saisir :

- le jour décimal de début du zoom
- la largeur du zoom en minute

efface_trace.m
Ce script permet d'effacer toues les éventuelles fenêtres de tracé.

est_un_entier.m
Ce script est appelé par diverses interfaces afin de vérifier que la valeur saisie par l'utilisateur est un entier. Un message d'erreur est affiché dans le cas contraire.

est_un_fichier.m
Ce script est appelé par diverses interfaces afin de vérifier que la valeur saisie par l'utilisateur est un fichier existant. Un message d'erreur est affiché dans la cas contraire.

est_un_nombre.m
Ce script est appelé par diverses interfaces afin de vérifier que la valeur saisie par l'utilisateur est un nombre. Un message d'erreur est affiché dans la cas contraire.

est_un_repertoire.m
Ce script est appelé par diverses interfaces afin de vérifier que la valeur saisie par l'utilisateur est un répertoire existant. Un message d'erreur est affiché dans la cas contraire.

est_une_annee.m

Ce script est appelé par diverses interfaces afin de vérifier que la valeur saisie par l'utilisateur est une année valide. Un message d'erreur est affiché dans la cas contraire.
Pour l'année de la campagne, elle doit se situer entre 1900 et l'année en cours.
Pour l'année de référence, elle doit se situer entre 1900 et l'année de la campagne.

global_adcp.m
Ce script définit toutes les variables globales, c'est-à-dire communes à diverses étapes et interfaces.

greg_0h.m
Ce script permet de transformer un jour julien en un jour grégorien.

hms2h.m :
Ce script convertit une heure donnée en (heure, minute, secondes) en fraction d'heure.

jul_0h.m
Ce script permet de transformer les jours grégoriens en jours juliens. Dans CASCADE, les jours juliens débutent à minuit.

lit_info_adcp.m
Ce script est appelé par ***info_camp.m, ad1_brut2cdf_nb.m, ad1_brut2cdf_bb.m*** et ***ad2_ensmoy.m.***. En fonction du type d'ADCP sélectionné par l'utilisateur, il récupère dans le fichier **info_adcp** (sous /home3/doelan/chemon/soft_adcp/adcp_lpo/global)

- la fréquence réelle de l'ADCP (en Khz)
- le facteur d'échelle (Doppler Shift) intervenant dans ***ad2_ensmoy.m*** pour passer des valeurs de fréquence Doppler en vitesses.

lit_info_adcp_navire.m
Ce script est appelé par ***info_camp.m, ad1_cal_vraie_derive.m*** et ***ad1_corr_att.m***. Il permet de récupérer dans le fichier **info_navire** l'angle de l'ADCP par rapport à l'axe du navire, l'angle des faisceaux par rapport à la verticale ainsi que les éventuelles informations sur les noms de variables et conventions de signe associées aux attitudes externes disponibles.

mat_navgeo.m
Ce script permet de définir la matrice de passage des vitesses dans le repère du navire aux vitesses dans le repère terrestre.

message.m
Cette interface est appelée par diverses autres interfaces. Elle permet d'afficher un message à l'utilisateur. Cette interface est inspirée de msgbox.m de Matlab. La différence réside dans le fait que dans message.m, le bouton « OK » n'existe pas.

quit_adcp.m
Cette interface est appelée par ***cascade.m*** dans le callback du menu 'quitter'. Il permet à l'utilisateur de quitter l'application de traitement des données d'ADCP de coque après confirmation.

sauv_var.m

Ce script permet de sauvegarder les variables et constantes de traitement dans le fichier ***conf.mat.***

s2hms.m
Ce script permet de convertir une heure donnée en secondes en une heure sous la forme (heure, minute, seconde).

test_numfic.m
Ce script est appelé par ***demande_numfic.m*** pour vérifier que le numéro du dernier fichier à traiter est supérieur ou égal au premier numéro de fichier à traiter.

test_valid_fic.m
Ce script est appelé par ***demande_numfic.m*** afin de vérifier que les fichiers à traiter sont conformes au logiciel, c'est-à-dire qu'ils sont :

- relatifs à un ADCP concave
- relatifs à un ADCP regardant vers le bas
- enregistrés en mode « BEAM ».

Dans le cas contraire, un message est affiché à l'utilisateur et le traitement suspendu.

6.Annexes

6.1Fichier info_adcp

```
%
% Fichier (issu de la doc. RDI : "VM_ADCP  Technical manual" (sept.91))
% permettant d'associer aux types d'adcp la frequence et le facteur
% d'echelle.
%
% Structure du fichier :
%       modele_adcp frequence fact_echelle(high range) fact_echelle(low range).
%
%
% C. Kermabon : 7/12/98.
%

75nb        76.8        0.25        0.25
150nb       153.6       0.5           0.25
300nb       307.2       1.0           0.5
600nb       614.4       2.0           1.0
1200nb      1228.8      4.0           2.0
150bb       153.6       1.0           1.0
```

6.2Fichier info_navire

```
%
% Fichier precisant les informations d'attitude externes
% associe aux differents navire.
%
%
% Structure du fichier :
%       1 ligne par navire et par type ADCP indiquant :
%   Nom_navire Type_ADCP prof_adcp AngleFaisceau AngleADCP/Navire
convention_roulis_interne convention_tangage_interne
% La convention roulis : 1 ==> sens trigo direct (roulis>0 sens trigo)
%                       -1 ==> sens trigo indirect.
%
% La convention tangage : 1 ==> tangage positif quand le navire leve le nez
%                        -1 ==> tangage negatif quand le navire leve le nez.
%
% C. Kermabon : 19/09/2002
%
ATALANTE            75NB    6 20 45  -1 1
THALASSA            150BB   6 30 45  -1 1
THALASSA            75NB   6 30 45  -1 1
SUROIT                  75NB    6 30 45 -1 1
ENTRECASTEAUX           75NB    6 30 -45 -1 1
```

6.3Extrait du fichier RDI navigation

```
ENSEMBLE  5179 PCTIME 7608037
2,281.83,281.33,MRS,282.98,-00.64,+00.33,+00.29
$CADCP,16/11/01,21:07:56.075,N,48,20.14780,W,005,35.07250,274.40,+11.99,N,4
8,20.14897,W,005,35.07430,282.33,281.67,MRS,283.50,-01.11,+00.18,+00.50
$CADCP,16/11/01,21:07:57.075,N,48,20.14786,W,005,35.07751,274.20,+11.99,N,4
8,20.14933,W,005,35.07924,282.67,282.17,MRS,283.96,-01.15,-00.15,+00.49

ENSEMBLE  5180 PCTIME 7608251
$CADCP,16/11/01,21:07:58.075,N,48,20.14804,W,005,35.08271,273.90,+12.07,N,4
8,20.14971,W,005,35.08417,283.00,282.33,MRS,284.24,-00.80,-00.21,+00.37
$CADCP,16/11/01,21:07:59.075,N,48,20.14832,W,005,35.08783,274.20,+12.11,N,4
8,20.15011,W,005,35.08910,283.17,282.50,MRS,284.47,-00.40,+00.15,+00.29

ENSEMBLE  5181 PCTIME 7608471
```

6.4Structure NetCDF du fichier d'attitude

```
netcdf att_evh001r {
dimensions:
      dim_brd = 10402 ;
variables:
      double TBRD(dim_brd) ;
            TBRD:units = "decimal days" ;
            TBRD:long_name = "JULIAN DAYS BRD" ;
            TBRD:_FillValue = -999999. ;
            TBRD:valid_min = 0 ;
      float BRD_Hdg(dim_brd) ;
            BRD_Hdg:units = "degrees" ;
            BRD_Hdg:long_name = "BRD Heading" ;
            BRD_Hdg:_FillValue = -999999.f ;
            BRD_Hdg:valid_range = 0.f, 359.99f ;
      float BRD_Ptch(dim_brd) ;
            BRD_Ptch:units = "degrees" ;
            BRD_Ptch:long_name = "BRD Pitch" ;
            BRD_Ptch:_FillValue = -999999.f ;
            BRD_Ptch:valid_range = -180.f, 180.f ;
      float BRD_Roll(dim_brd) ;
            BRD_Roll:units = "degrees" ;
            BRD_Roll:long_name = "BRD Roll" ;
            BRD_Roll:_FillValue = -999999.f ;
            BRD_Roll:valid_range = -180.f, 180.f ;
      float BRD_gdop(dim_brd) ;
            BRD_gdop:units = "gdop" ;
            BRD_gdop:long_name = "BRD gdop" ;
            BRD_gdop:_FillValue = -999999.f ;
      double BRD_Lat(dim_brd) ;
            BRD_Lat:units = "degrees" ;
            BRD_Lat:long_name = "BRD Latitude" ;
            BRD_Lat:_FillValue = -999999. ;
            BRD_Lat:valid_range = -90., 90. ;
      double BRD_Lon(dim_brd) ;
            BRD_Lon:units = "degrees" ;
            BRD_Lon:long_name = "BRD Longitude" ;
            BRD_Lon:_FillValue = -999999. ;
            BRD_Lon:valid_range = -180., 180. ;
      float BRD_VitU(dim_brd) ;
            BRD_VitU:units = "m/s" ;
            BRD_VitU:long_name = "BRD Vitesse U" ;
            BRD_VitU:_FillValue = -999999.f ;
            BRD_VitU:valid_range = -15.f, 15.f ;
      float BRD_VitV(dim_brd) ;
            BRD_VitV:units = "m/s" ;
            BRD_VitV:long_name = "BRD Vitesse V" ;
            BRD_VitV:_FillValue = -999999.f ;
            BRD_VitV:valid_range = -15.f, 15.f ;
      float BRD_AccX(dim_brd) ;
            BRD_AccX:units = "m/s*2" ;
            BRD_AccX:long_name = "BRD Acceleration U" ;
            BRD_AccX:_FillValue = -999999.f ;
            BRD_AccX:valid_range = -0.1f, 0.1f ;
      float BRD_AccY(dim_brd) ;
            BRD_AccY:units = "m/s*2" ;
```

```
            BRD_AccY:long_name = "BRD Acceleration V" ;
            BRD_AccY:_FillValue = -999999.f ;
            BRD_AccY:valid_range = -0.1f, 0.1f ;
      float BRD_Lfilt ;
            BRD_Lfilt:units = "seconds" ;
            BRD_Lfilt:long_name = "BRD Largeur filtre" ;
            BRD_Lfilt:_FillValue = -999999.f ;
            BRD_Lfilt:valid_range = 0.f, 3600.f ;

// global attributes:
            :CREATION_DATE = "14-Mar-2002" ;
            :INST_TYPE = "BORD INFO" ;
            :PROG_CMNT1 = "Converted to netCDF via MATLAB by Cdf_att.m" ;
}
```

6.5Structure NetCDF du fichier de navigation trinav

```
netcdf c98tri {
dimensions:
        dim_trin = 211271 ;
        cte = 1 ;
variables:
        float Lfilt_TRIN ;                  /* Largeur du filtre utilisée sur les données
                                               de navigation. */
                Lfilt_TRIN:units = "seconds" ;
                Lfilt_TRIN:long_name = "Largeur du filtre" ;
                Lfilt_TRIN:_FillValue = -999999.f ;
                Lfilt_TRIN:valid_range = 0.f, 3600.f ;
        double T_TRIN(dim_trin) ;           /* Date de navigation en jour décimal. */
                T_TRIN:units = "decimal days" ;
                T_TRIN:long_name = "JULIAN DAYS TRINAV" ;
                T_TRIN:_FillValue = -999999. ;
                T_TRIN:valid_range = 2450450., 2451450. ;
        double Lat_TRIN(dim_trin) ;         /* Latitude. */
                Lat_TRIN:units = "decimal Degrees" ;
                Lat_TRIN:long_name = "LATITUDE TRINAV" ;
                Lat_TRIN:_FillValue = -999999. ;
                Lat_TRIN:valid_range = -90., 90. ;
        double Lon_TRIN(dim_trin) ;         /* Longitude. */
                Lon_TRIN:units = "decimal Degrees" ;
                Lon_TRIN:long_name = "LONGITUDE TRINAV" ;
                Lon_TRIN:_FillValue = -999999. ;
                Lon_TRIN:valid_range = -180., 180. ;
        float VitU_TRIN(dim_trin) ;         /* Vitesse zonale du navire. */
                VitU_TRIN:units = "m/s" ;
                VitU_TRIN:long_name = "Vitesse navire filtree (U)" ;
                VitU_TRIN:_FillValue = -999999.f ;
                VitU_TRIN:valid_range = -15.f, 15.f ;
        float VitV_TRIN(dim_trin) ;         /* Vitesse méridienne du navire. */
                VitV_TRIN:units = "m/s" ;
                VitV_TRIN:long_name = "Vitesse navire filtree (V)" ;
                VitV_TRIN:_FillValue = -999999.f ;
                VitV_TRIN:valid_range = -15.f, 15.f ;
        float AccX_TRIN(dim_trin) ;         /* Accélération en X du navire. */
                AccX_TRIN:units = "m/s*2" ;
                AccX_TRIN:long_name = "Vitesse navire filtree (x)" ;
                AccX_TRIN:_FillValue = -999999.f ;
                AccX_TRIN:valid_range = -0.1f, 0.1f ;
        float AccY_TRIN(dim_trin) ;         /* Accélération en Y du navire. */
                AccY_TRIN:units = "m/s*2" ;
                AccY_TRIN:long_name = "Vitesse navire filtree (y)" ;
                AccY_TRIN:_FillValue = -999999.f ;
                AccY_TRIN:valid_range = -0.1f, 0.1f ;
```

// global attributes:
:CREATION_DATE = "02-Nov-1998" ;
:INST_TYPE = "TRINAV" ;
:PROG_CMNT1 = "Converted to netCDF via MATLAB by Cdf_trin_c" ;

6.6Structure du fichier brut NetCDF ADCP

6.6.1Contenu d'un fichier NetCDF NarrowBand

```
netcdf camb028r {
dimensions:
        ensemble = UNLIMITED ; // (6679 currently)
        bin = 50 ;
        cte = 1 ;
variables:
        double TIM(ensemble) ;
                TIM:units = "decimal days" ;
                TIM:long_name = "Internal ADCP time (julian days)" ;
                TIM:_FillValue = -999999. ;
                TIM:epic_code = 627 ;
                TIM:valid_min = 0 ;
        double TGPS(ensemble) ;
                TGPS:units = "decimal days" ;
                TGPS:long_name = "GPS time (julian days)" ;
                TGPS:_FillValue = -999999. ;
                TGPS:valid_min = 0 ;
        float DGPS(cte) ;
                DGPS:units = "seconds" ;
                DGPS:long_name = "mean difference GPS-INST time" ;
                DGPS:_FillValue = -999999.f ;
        float Xoff(cte) ;
                Xoff:units = "meters" ;
                Xoff:long_name = "Transducer offset" ;
                Xoff:_FillValue = -999999.f ;
        float ADCPDe(cte) ;
                ADCPDe:units = "meters" ;
                ADCPDe:long_name = "ADCP Depth" ;
                ADCPDe:_FillValue = -999999.f ;
        int Rec(ensemble) ;
                Rec:units = "counts" ;
                Rec:long_name = "Records" ;
                Rec:_FillValue = -999999 ;
                Rec:epic_code = 1207 ;
                Rec:valid_min = 0 ;
        short Bst(ensemble) ;
                Bst:units = "status" ;
                Bst:long_name = "BIT status" ;
                Bst:_FillValue = -9999s ;
                Bst:valid_range = 0s, 255s ;
        float Hdg(ensemble) ;
                Hdg:units = "degrees" ;
                Hdg:long_name = "Internal Heading" ;
                Hdg:epic_code = 1215 ;
                Hdg:_FillValue = -999999.f ;
```

Hdg:valid_range = 0.f, 360.f ;
float Ptch(ensemble) ;
Ptch:units = "degrees" ;
Ptch:long_name = "Internal Pitch" ;
Ptch:epic_code = 1216 ;
Ptch:_FillValue = -999999.f ;
Ptch:valid_range = -180.f, 180.f ;
float Roll(ensemble) ;
Roll:units = "degrees" ;
Roll:long_name = "Internal Roll" ;
Roll:epic_code = 1217 ;
Roll:_FillValue = -999999.f ;
Roll:valid_range = -180.f, 180.f ;
float STDHdg(ensemble) ;
STDHdg:units = "degrees" ;
STDHdg:long_name = "std. dev. Internal Heading" ;
STDHdg:_FillValue = -999999.f ;
float STDPtch(ensemble) ;
STDPtch:units = "degrees" ;
STDPtch:long_name = "std. dev. Internal Pitch" ;
STDPtch:_FillValue = -999999.f ;
float STDRoll(ensemble) ;
STDRoll:units = "degrees" ;
STDRoll:long_name = "std. dev. Internal Roll" ;
STDRoll:_FillValue = -999999.f ;
float HdgExt(ensemble) ;
HdgExt:units = "degrees" ;
HdgExt:long_name = "External Heading" ;
HdgExt:epic_code = 1215 ;
HdgExt:_FillValue = -999999.f ;
HdgExt:valid_range = 0.f, 360.f ;
float PtchExt(ensemble) ;
PtchExt:units = "degrees" ;
PtchExt:long_name = "External Pitch" ;
PtchExt:epic_code = 1216 ;
PtchExt:_FillValue = -999999.f ;
PtchExt:valid_range = -180.f, 180.f ;
float RollExt(ensemble) ;
RollExt:units = "degrees" ;
RollExt:long_name = "External Roll" ;
RollExt:epic_code = 1217 ;
RollExt:_FillValue = -999999.f ;
RollExt:valid_range = -180.f, 180.f ;
float Tx(ensemble) ;
Tx:units = "degrees" ;
Tx:long_name = "ADCP Transducer Temperature" ;
Tx:epic_code = 3017 ;
Tx:_FillValue = -999999.f ;
Tx:valid_range = -5.f, 45.f ;
float HIv(ensemble) ;
HIv:units = "volts" ;
HIv:long_name = "HI volt in" ;

HIv:_FillValue = -999999.f ;
float LOv(ensemble) ;
LOv:units = "volts" ;
LOv:long_name = "LO volt in" ;
LOv:_FillValue = -999999.f ;
float xmitc(ensemble) ;
xmitc:units = "amps" ;
xmitc:long_name = "XMIT currente" ;
xmitc:_FillValue = -999999.f ;
float BTV1(ensemble) ;
BTV1:units = "count" ;
BTV1:long_name = "Beam 1 bottom track velocity " ;
BTV1:_FillValue = -999999.f ;
BTV1:valid_range = -32768.f, 32767.f ;
float BTV2(ensemble) ;
BTV2:units = "count" ;
BTV2:long_name = "Beam 2 bottom track velocity " ;
BTV2:_FillValue = -999999.f ;
BTV2:valid_range = -32768.f, 32767.f ;
float BTV3(ensemble) ;
BTV3:units = "count" ;
BTV3:long_name = "Beam 3 bottom track velocity »;
BTV3:_FillValue = -999999.f ;
BTV3:valid_range = -32768.f, 32767.f ;
float BTV4(ensemble) ;
BTV4:units = "count" ;
BTV4:long_name = "Beam 4 bottom track velocity " ;
BTV4:_FillValue = -999999.f ;
BTV4:valid_range = -32768.f, 32767.f ;
float BTR1(ensemble) ;
BTR1:units = "meters" ;
BTR1:long_name = "Beam 1 bottom track range, m" ;
BTR1:_FillValue = -999999.f ;
BTR1:valid_range = 0.f, 9999.f ;
float BTR2(ensemble) ;
BTR2:units = "meters" ;
BTR2:long_name = "Beam 2 bottom track range, m" ;
BTR2:_FillValue = -999999.f ;
BTR2:valid_range = 0.f, 9999.f ;
float BTR3(ensemble) ;
BTR3:units = "meters" ;
BTR3:long_name = "Beam 3 bottom track range, m" ;
BTR3:_FillValue = -999999.f ;
BTR3:valid_range = 0.f, 9999.f ;
float BTR4(ensemble) ;
BTR4:units = "meters" ;
BTR4:long_name = "Beam 4 bottom track range, m" ;
BTR4:_FillValue = -999999.f ;
BTR4:valid_range = 0.f, 9999.f ;
float BTPGd1(ensemble) ;
BTPGd1:units = "percent" ;
BTPGd1:long_name = "Beam 1 bottom track percent good" ;

BTPGd1:_FillValue = -999999.f ;
BTPGd1:valid_range = 0.f, 100.f ;
float BTPGd2(ensemble) ;
BTPGd2:units = "percent" ;
BTPGd2:long_name = "Beam 2 bottom track percent good" ;
BTPGd2:_FillValue = -999999.f ;
BTPGd2:valid_range = 0.f, 100.f ;
float BTPGd3(ensemble) ;
BTPGd3:units = "percent" ;
BTPGd3:long_name = "Beam 3 bottom track percent good" ;
BTPGd3:_FillValue = -999999.f ;
BTPGd3:valid_range = 0.f, 100.f ;
float BTPGd4(ensemble) ;
BTPGd4:units = "percent" ;
BTPGd4:long_name = "Beam 4 bottom track percent good" ;
BTPGd4:_FillValue = -999999.f ;
BTPGd4:valid_range = 0.f, 100.f ;
short vel1(ensemble, bin) ;
vel1:units = "count" ;
vel1:long_name = "Beam 1 velocity" ;
vel1:_FillValue = -9999s ;
vel1:valid_range = -2048s, 2047s ;
short vel2(ensemble, bin) ;
vel2:units = "count" ;
vel2:long_name = "Beam 2 velocity" ;
vel2:_FillValue = -9999s ;
vel2:valid_range = -2048s, 2047s ;
short vel3(ensemble, bin) ;
vel3:units = "count" ;
vel3:long_name = "Beam 3 velocity" ;
vel3:_FillValue = -9999s ;
vel3:valid_range = -2048s, 2047s ;
short vel4(ensemble, bin) ;
vel4:units = "count" ;
vel4:long_name = "Beam 4 velocity" ;
vel4:_FillValue = -9999s ;
vel4:valid_range = -2048s, 2047s ;
short ECI1(ensemble, bin) ;
ECI1:units = "counts" ;
ECI1:long_name = "Echo Intensity (ECI) Beam 1" ;
ECI1:epic_code = 1221 ;
ECI1:_FillValue = -9999s ;
ECI1:valid_range = 0s, 255s ;
ECI1:norm_factor = 0.45f ;
short ECI2(ensemble, bin) ;
ECI2:units = "counts" ;
ECI2:long_name = "Echo Intensity (ECI) Beam 2" ;
ECI2:epic_code = 1222 ;
ECI2:_FillValue = -9999s ;
ECI2:valid_range = 0s, 255s ;
ECI2:norm_factor = 0.45f ;
short ECI3(ensemble, bin) ;

```
                ECI3:units = "counts" ;
                ECI3:long_name = "Echo Intensity (ECI) Beam 3" ;
                ECI3:epic_code = 1223 ;
                ECI3:_FillValue = -9999s ;
                ECI3:valid_range = 0s, 255s ;
                ECI3:norm_factor = 0.45f ;
        short ECI4(ensemble, bin) ;
                ECI4:units = "counts" ;
                ECI4:long_name = "Echo Intensity (ECI) Beam 4" ;
                ECI4:epic_code = 1224 ;
                ECI4:_FillValue = -9999s ;
                ECI4:valid_range = 0s, 255s ;
                ECI4:norm_factor = 0.45f ;
        byte PGd1(ensemble, bin) ;
                PGd1:units = "%" ;
                PGd1:long_name = "Percent Good Beam 1" ;
                PGd1:epic_code = 1241 ;
                PGd1:_FillValue = 0b ;
                PGd1:valid_range = 0b, 100b ;
        byte PGd2(ensemble, bin) ;
                PGd2:units = "%" ;
                PGd2:long_name = "Percent Good Beam 2" ;
                PGd2:epic_code = 1242 ;
                PGd2:_FillValue = 0b ;
                PGd2:valid_range = 0b, 100b ;
        byte PGd3(ensemble, bin) ;
                PGd3:units = "%" ;
                PGd3:long_name = "Percent Good Beam 3" ;
                PGd3:epic_code = 1243 ;
                PGd3:_FillValue = 0b ;
                PGd3:valid_range = 0b, 100b ;
        byte PGd4(ensemble, bin) ;
                PGd4:units = "%" ;
                PGd4:long_name = "Percent Good Beam 4" ;
                PGd4:epic_code = 1244 ;
                PGd4:_FillValue = 0b ;
                PGd4:valid_range = 0b, 100b ;

// global attributes:
                :Soft_Version = "Version 2.0" ;
                :Creation_Date = "02-Oct-2000" ;
                :ADCP_Constructor = "RDI" ;
                :ADCP_Type = "75nb  " ;
                :Cruise = "Cambios98" ;
                :Vessel = "THALASSA     " ;
                :Beam_Angle = 30.f ;
                :ADCP_Angle = 45.f ;
                :Real_Frequency_in_Khz = 76.8f ;
                :Last_Update = "02-Oct-2000" ;
                :time_between_ping_groups = 0, 1, 10 ;
                :pings_per_ensemble = 1 ;
                :number_of_bins = 50 ;
```

```
                :bin_length_in_m = 16 ;
                :xmit_pulse_length_in_m = 16 ;
                :blanking_distance_in_m = 8 ;
                :delay_after_blank_in_m = 0 ;
                :hardware_configuration = 141 ;
                :Supposed_Frequency_in_Khz = 75 ;
                :beam_pattern = "concave" ;
                :orientation = "DOWN" ;
                :transform = "BEAM" ;
                :velocity_range = "HIGH" ;
                :SN_threshold_in_db = 6.f ;
                :percent_good = 25 ;
                :Scale_Factor = 0.25f ;
}
```

6.6.2Contenu d'un fichier BroadBand 150 Khz

```
netcdf pel2008r {
dimensions:
        ensemble = UNLIMITED ; // (1666 currently)
        bin = 40 ;
        cte = 1 ;
variables:
        double TIM(ensemble) ;
                TIM:units = "decimal days" ;
                TIM:long_name = "Internal ADCP time (julian days)" ;
                TIM:_FillValue = -999999. ;
                TIM:epic_code = 627 ;
                TIM:valid_min = 0 ;
        double TGPS(ensemble) ;
                TGPS:units = "decimal days" ;
                TGPS:long_name = "GPS time (julian days)" ;
                TGPS:_FillValue = -999999. ;
                TGPS:valid_min = 0 ;
        float DGPS(cte) ;
                DGPS:units = "seconds" ;
                DGPS:long_name = "mean difference GPS-INST time" ;
                DGPS:_FillValue = -999999.f ;
        float Xoff(cte) ;
                Xoff:units = "meters" ;
                Xoff:long_name = "Transducer offset" ;
                Xoff:_FillValue = -999999.f ;
        float ADCPDe(cte) ;
                ADCPDe:units = "meters" ;
                ADCPDe:long_name = "ADCP Depth" ;
                ADCPDe:_FillValue = -999999.f ;
        int Rec(ensemble) ;
                Rec:units = "counts" ;
                Rec:long_name = "Records" ;
                Rec:_FillValue = -999999 ;
                Rec:epic_code = 1207 ;
                Rec:valid_min = 0 ;
        float Hdg(ensemble) ;
                Hdg:units = "degrees" ;
                Hdg:long_name = "Internal Heading" ;
                Hdg:epic_code = 1215 ;
                Hdg:_FillValue = -999999.f ;
                Hdg:valid_range = 0.f, 360.f ;
        float Ptch(ensemble) ;
                Ptch:units = "degrees" ;
                Ptch:long_name = "Internal Pitch" ;
                Ptch:epic_code = 1216 ;
                Ptch:_FillValue = -999999.f ;
                Ptch:valid_range = -180.f, 180.f ;
        float Roll(ensemble) ;
                Roll:units = "degrees" ;
```

Roll:long_name = "Internal Roll" ;
Roll:epic_code = 1217 ;
Roll:_FillValue = -999999.f ;
Roll:valid_range = -180.f, 180.f ;
float STDHdg(ensemble) ;
STDHdg:units = "degrees" ;
STDHdg:long_name = "std. dev. Internal Heading" ;
STDHdg:_FillValue = -999999.f ;
float STDPtch(ensemble) ;
STDPtch:units = "degrees" ;
STDPtch:long_name = "std. dev. Internal Pitch" ;
STDPtch:_FillValue = -999999.f ;
float STDRoll(ensemble) ;
STDRoll:units = "degrees" ;
STDRoll:long_name = "std. dev. Internal Roll" ;
STDRoll:_FillValue = -999999.f ;
float HdgExt(ensemble) ;
HdgExt:units = "degrees" ;
HdgExt:long_name = "External Heading" ;
HdgExt:epic_code = 1215 ;
HdgExt:_FillValue = -999999.f ;
HdgExt:valid_range = 0.f, 360.f ;
float PtchExt(ensemble) ;
PtchExt:units = "degrees" ;
PtchExt:long_name = "External Pitch" ;
PtchExt:epic_code = 1216 ;
PtchExt:_FillValue = -999999.f ;
PtchExt:valid_range = -180.f, 180.f ;
float RollExt(ensemble) ;
RollExt:units = "degrees" ;
RollExt:long_name = "External Roll" ;
RollExt:epic_code = 1217 ;
RollExt:_FillValue = -999999.f ;
RollExt:valid_range = -180.f, 180.f ;
float Tx(ensemble) ;
Tx:units = "degrees" ;
Tx:long_name = "ADCP Transducer Temperature" ;
Tx:epic_code = 3017 ;
Tx:_FillValue = -999999.f ;
Tx:valid_range = -5.f, 45.f ;
float SoundSpeed(ensemble) ;
SoundSpeed:units = "m s-1" ;
SoundSpeed:long_name = "Speed of sound" ;
SoundSpeed:_FillValue = -999999.f ;
float BTV1(ensemble) ;
BTV1:units = "cm s-1" ;
BTV1:long_name = "Beam 1 bottom track velocity, cm/s" ;
BTV1:_FillValue = -999999.f ;
BTV1:valid_range = -32768.f, 32767.f ;
float BTV2(ensemble) ;
BTV2:units = "cm s-1" ;
BTV2:long_name = "Beam 2 bottom track velocity, cm/s" ;

BTV2:_FillValue = -999999.f ;
BTV2:valid_range = -32768.f, 32767.f ;
float BTV3(ensemble) ;
BTV3:units = "cm s-1" ;
BTV3:long_name = "Beam 3 bottom track velocity, cm/s" ;
BTV3:_FillValue = -999999.f ;
BTV3:valid_range = -32768.f, 32767.f ;
float BTV4(ensemble) ;
BTV4:units = "cm s-1" ;
BTV4:long_name = "Beam 4 bottom track velocity, cm/s" ;
BTV4:_FillValue = -999999.f ;
BTV4:valid_range = -32768.f, 32767.f ;
float BTR1(ensemble) ;
BTR1:units = "meters" ;
BTR1:long_name = "Beam 1 bottom track range, m" ;
BTR1:_FillValue = -999999.f ;
BTR1:valid_range = 0.f, 9999.f ;
float BTR2(ensemble) ;
BTR2:units = "meters" ;
BTR2:long_name = "Beam 2 bottom track range, m" ;
BTR2:_FillValue = -999999.f ;
BTR2:valid_range = 0.f, 9999.f ;
float BTR3(ensemble) ;
BTR3:units = "meters" ;
BTR3:long_name = "Beam 3 bottom track range, m" ;
BTR3:_FillValue = -999999.f ;
BTR3:valid_range = 0.f, 9999.f ;
float BTR4(ensemble) ;
BTR4:units = "meters" ;
BTR4:long_name = "Beam 4 bottom track range, m" ;
BTR4:_FillValue = -999999.f ;
BTR4:valid_range = 0.f, 9999.f ;
float BTPGd1(ensemble) ;
BTPGd1:units = "percent" ;
BTPGd1:long_name = "Beam 1 bottom track percent good" ;
BTPGd1:_FillValue = -999999.f ;
BTPGd1:valid_range = 0.f, 100.f ;
float BTPGd2(ensemble) ;
BTPGd2:units = "percent" ;
BTPGd2:long_name = "Beam 2 bottom track percent good" ;
BTPGd2:_FillValue = -999999.f ;
BTPGd2:valid_range = 0.f, 100.f ;
float BTPGd3(ensemble) ;
BTPGd3:units = "percent" ;
BTPGd3:long_name = "Beam 3 bottom track percent good" ;
BTPGd3:_FillValue = -999999.f ;
BTPGd3:valid_range = 0.f, 100.f ;
float BTPGd4(ensemble) ;
BTPGd4:units = "percent" ;
BTPGd4:long_name = "Beam 4 bottom track percent good" ;
BTPGd4:_FillValue = -999999.f ;
BTPGd4:valid_range = 0.f, 100.f ;

short vel1(ensemble, bin) ;
vel1:units = "cm s-1" ;
vel1:long_name = "Beam 1 velocity" ;
vel1:_FillValue = -9999s ;
vel1:valid_range = -2048s, 2047s ;
short vel2(ensemble, bin) ;
vel2:units = "cm s-1" ;
vel2:long_name = "Beam 2 velocity" ;
vel2:_FillValue = -9999s ;
vel2:valid_range = -2048s, 2047s ;
short vel3(ensemble, bin) ;
vel3:units = "cm s-1" ;
vel3:long_name = "Beam 3 velocity" ;
vel3:_FillValue = -9999s ;
vel3:valid_range = -2048s, 2047s ;
short vel4(ensemble, bin) ;
vel4:units = "cm s-1" ;
vel4:long_name = "Beam 4 velocity" ;
vel4:_FillValue = -9999s ;
vel4:valid_range = -2048s, 2047s ;
short COR1(ensemble, bin) ;
COR1:units = "counts" ;
COR1:long_name = "Correlation Magnitude Beam 1" ;
COR1:_FillValue = -9999s ;
COR1:valid_range = 0s, 255s ;
COR1:norm_factor = 0.45f ;
short COR2(ensemble, bin) ;
COR2:units = "counts" ;
COR2:long_name = "Correlation Magnitude Beam 2" ;
COR2:_FillValue = -9999s ;
COR2:valid_range = 0s, 255s ;
COR2:norm_factor = 0.45f ;
short COR3(ensemble, bin) ;
COR3:units = "counts" ;
COR3:long_name = "Correlation Magnitude Beam 3" ;
COR3:_FillValue = -9999s ;
COR3:valid_range = 0s, 255s ;
COR3:norm_factor = 0.45f ;
short COR4(ensemble, bin) ;
COR4:units = "counts" ;
COR4:long_name = "Correlation Magnitude Beam 4" ;
COR4:_FillValue = -9999s ;
COR4:valid_range = 0s, 255s ;
COR4:norm_factor = 0.45f ;
short ECI1(ensemble, bin) ;
ECI1:units = "counts" ;
ECI1:long_name = "Echo Intensity (ECI) Beam 1" ;
ECI1:epic_code = 1221 ;
ECI1:_FillValue = -9999s ;
ECI1:valid_range = 0s, 255s ;
ECI1:norm_factor = 0.45f ;
short ECI2(ensemble, bin) ;

ECI2:units = "counts" ;
ECI2:long_name = "Echo Intensity (ECI) Beam 2" ;
ECI2:epic_code = 1222 ;
ECI2:_FillValue = -9999s ;
ECI2:valid_range = 0s, 255s ;
ECI2:norm_factor = 0.45f ;
short ECI3(ensemble, bin) ;
ECI3:units = "counts" ;
ECI3:long_name = "Echo Intensity (ECI) Beam 3" ;
ECI3:epic_code = 1223 ;
ECI3:_FillValue = -9999s ;
ECI3:valid_range = 0s, 255s ;
ECI3:norm_factor = 0.45f ;
short ECI4(ensemble, bin) ;
ECI4:units = "counts" ;
ECI4:long_name = "Echo Intensity (ECI) Beam 4" ;
ECI4:epic_code = 1224 ;
ECI4:_FillValue = -9999s ;
ECI4:valid_range = 0s, 255s ;
ECI4:norm_factor = 0.45f ;
byte PGd1(ensemble, bin) ;
PGd1:units = "%" ;
PGd1:long_name = "Percent Good Beam 1" ;
PGd1:epic_code = 1241 ;
PGd1:_FillValue = 0b ;
PGd1:valid_range = 0b, 100b ;
byte PGd2(ensemble, bin) ;
PGd2:units = "%" ;
PGd2:long_name = "Percent Good Beam 2" ;
PGd2:epic_code = 1242 ;
PGd2:_FillValue = 0b ;
PGd2:valid_range = 0b, 100b ;
byte PGd3(ensemble, bin) ;
PGd3:units = "%" ;
PGd3:long_name = "Percent Good Beam 3" ;
PGd3:epic_code = 1243 ;
PGd3:_FillValue = 0b ;
PGd3:valid_range = 0b, 100b ;
byte PGd4(ensemble, bin) ;
PGd4:units = "%" ;
PGd4:long_name = "Percent Good Beam 4" ;
PGd4:epic_code = 1244 ;
PGd4:_FillValue = 0b ;
PGd4:valid_range = 0b, 100b ;
short Bst1(ensemble, bin) ;
Bst1:units = "status" ;
Bst1:long_name = "BIT status" ;
Bst1:_FillValue = -9999s ;
Bst1:valid_range = 0s, 1s ;
short Bst2(ensemble, bin) ;
Bst2:units = "status" ;
Bst2:long_name = "BIT status" ;

```
                Bst2:_FillValue = -9999s ;
                Bst2:valid_range = 0s, 1s ;
        short Bst3(ensemble, bin) ;
                Bst3:units = "status" ;
                Bst3:long_name = "BIT status" ;
                Bst3:_FillValue = -9999s ;
                Bst3:valid_range = 0s, 1s ;
        short Bst4(ensemble, bin) ;
                Bst4:units = "status" ;
                Bst4:long_name = "BIT status" ;
                Bst4:_FillValue = -9999s ;
                Bst4:valid_range = 0s, 1s ;

// global attributes:
                :Soft_Version = "Version 2.0" ;
                :Creation_Date = "09-Oct-2000" ;
                :ADCP_Constructor = "RDI" ;
                :ADCP_Type = "150bb " ;
                :Cruise = "CTHOMAS" ;
                :Vessel = "THALASSA     " ;
                :Beam_Angle = 30.f ;
                :ADCP_Angle = 45.f ;
                :Real_Frequency_in_Khz = 153.6f ;
                :Last_Update = "09-Oct-2000" ;
                :Scale_Factor = 1.f ;
                :hardware_configuration = 65 ;
                :frequency_in_KHz = 150 ;
                :beam_pattern = "concave" ;
                :orientation = "DOWN" ;
                :number_of_bins = 40 ;
                :pings_per_ensemble = 1 ;
                :bin_length_in_m = 8 ;
                :blanking_distance_in_m = 4 ;
                :percent_good = 0 ;
                :time_between_ping_groups = 9999, 0, 1 ;
                :transform = "BEAM" ;
                :xmit_pulse_length_in_m = 7.92f ;
                :Transmit_Lag_Distance = 0.47f ;
                :delay_after_blank_in_m = 0 ;
}
```

6.7 Structure NetCDF des fichiers « processed »

```
netcdf sem005p {
dimensions:
      Pensemble = 14 ;
      bin = 50 ;
      cte = 1 ;
variables:
      double PTIM(Pensemble) ;
            PTIM:units = "decimal days" ;
            PTIM:long_name = "Processed JULIAN DAYS ADCP" ;
            PTIM:_FillValue = -999999. ;
            PTIM:valid_min = 0. ;
      double PTGPS(Pensemble) ;
            PTGPS:units = "decimal days" ;
            PTGPS:long_name = "Processed JULIAN DAYS GPS" ;
            PTGPS:_FillValue = -999999. ;
            PTGPS:valid_min = 0. ;
      double PLat(Pensemble) ;
            PLat:units = "degrees" ;
            PLat:long_name = "Latitude" ;
            PLat:_FillValue = -999999. ;
            PLat:valid_range = -90., 90. ;
      double PLon(Pensemble) ;
            PLon:units = "degrees" ;
            PLon:long_name = "Longitude" ;
            PLon:_FillValue = -999999. ;
            PLon:valid_range = -180., 180. ;
      float Unav(Pensemble) ;
            Unav:units = "cm s-1" ;
            Unav:long_name = "Est West Boat Velocity" ;
            Unav:_FillValue = -999999.f ;
      float Vnav(Pensemble) ;
            Vnav:units = "cm s-1" ;
            Vnav:long_name = "Sud North Boat Velocity" ;
            Vnav:_FillValue = -999999.f ;
      float PTx(Pensemble) ;
            PTx:units = "degrees" ;
            PTx:long_name = "Processed ADCP Transducer Temperature" ;
            PTx:_FillValue = -999999.f ;
            PTx:valid_range = -5.f, 45.f ;
      float PHdg(Pensemble) ;
            PHdg:units = "degrees" ;
            PHdg:long_name = "Processed Heading" ;
            PHdg:_FillValue = -999999.f ;
            PHdg:valid_range = 0.f, 360.f ;
      float PPtch(Pensemble) ;
            PPtch:units = "degrees" ;
            PPtch:long_name = "Processed Pitch" ;
            PPtch:_FillValue = -999999.f ;
            PPtch:valid_range = -360.f, 360.f ;
      float PRoll(Pensemble) ;
            PRoll:units = "degrees" ;
            PRoll:long_name = "Processed Roll" ;
            PRoll:_FillValue = -999999.f ;
            PRoll:valid_range = -360.f, 360.f ;
      float EWVel(Pensemble, bin) ;
```

```
      EWVel:units = "cm s-1" ;
      EWVel:long_name = "East/West velocity" ;
      EWVel:_FillValue = -999999.f ;
float NSVel(Pensemble, bin) ;
      NSVel:units = "cm s-1" ;
      NSVel:long_name = "North/South velocity" ;
      NSVel:_FillValue = -999999.f ;
float UDVel(Pensemble, bin) ;
      UDVel:units = "cm s-1" ;
      UDVel:long_name = "Up/Down velocity" ;
      UDVel:_FillValue = -999999.f ;
float EVel(Pensemble, bin) ;
      EVel:units = "cm s-1" ;
      EVel:long_name = "Mean residual velocity" ;
      EVel:_FillValue = -999999.f ;
float PGood(Pensemble, bin) ;
      PGood:units = "percent %" ;
      PGood:long_name = "Percent Good" ;
      PGood:_FillValue = -999999.f ;
float PECI1(Pensemble, bin) ;
      PECI1:units = "counts" ;
      PECI1:long_name = "Echo intensity Beam1" ;
      PECI1:_FillValue = -999999.f ;
float PECI2(Pensemble, bin) ;
      PECI2:units = "counts" ;
      PECI2:long_name = "Echo intensity Beam2" ;
      PECI2:_FillValue = -999999.f ;
float PECI3(Pensemble, bin) ;
      PECI3:units = "counts" ;
      PECI3:long_name = "Echo intensity Beam3" ;
      PECI3:_FillValue = -999999.f ;
float PECI4(Pensemble, bin) ;
      PECI4:units = "counts" ;
      PECI4:long_name = "Echo intensity Beam4" ;
      PECI4:_FillValue = -999999.f ;
float PECI(Pensemble, bin) ;
      PECI:units = "counts" ;
      PECI:long_name = "Mean Echo intensity" ;
      PECI:_FillValue = -999999.f ;
float URMS(Pensemble, bin) ;
      URMS:units = "cm s-1" ;
      URMS:long_name = "Velocity East-West Root Mean Square" ;
      URMS:_FillValue = -999999.f ;
float VRMS(Pensemble, bin) ;
      VRMS:units = "cm s-1" ;
      VRMS:long_name = "Velocity South-North Root Mean Square" ;
      VRMS:_FillValue = -999999.f ;
float WRMS(Pensemble, bin) ;
      WRMS:units = "cm s-1" ;
      WRMS:long_name = "Velocity Up-Down Root Mean Square" ;
      WRMS:_FillValue = -999999.f ;
float ERMS(Pensemble, bin) ;
      ERMS:units = "cm s-1" ;
      ERMS:long_name = "Residual Velocity Root Mean Square" ;
      ERMS:_FillValue = -999999.f ;
float BT_EWVel(Pensemble) ;
      BT_EWVel:units = "cm s-1" ;
      BT_EWVel:long_name = "Bottom track East/West velocity" ;
      BT_EWVel:_FillValue = -999999.f ;
float BT_NSVel(Pensemble) ;
      BT_NSVel:units = "cm s-1" ;
```

```
            BT_NSVel:long_name = "Bottom track North/South velocity" ;
            BT_NSVel:_FillValue = -999999.f ;
      float BT_UDVel(Pensemble) ;
            BT_UDVel:units = "cm s-1" ;
            BT_UDVel:long_name = "Bottom track Up/Down velocity" ;
            BT_UDVel:_FillValue = -999999.f ;
      float BT_PGood(Pensemble) ;
            BT_PGood:units = "percent" ;
            BT_PGood:long_name = "Bottom percent good" ;
            BT_PGood:_FillValue = -999999.f ;
            BT_PGood:valid_range = 0.f, 100.f ;
      float BT_URMS(Pensemble) ;
            BT_URMS:units = "cm s-1" ;
            BT_URMS:long_name = "Bottom Velocity East-West Root Mean
Square" ;
            BT_URMS:_FillValue = -999999.f ;
      float BT_VRMS(Pensemble) ;
            BT_VRMS:units = "cm s-1" ;
            BT_VRMS:long_name = "Bottom Velocity South-North Root Mean
Square" ;
            BT_VRMS:_FillValue = -999999.f ;
      float BT_WRMS(Pensemble) ;
            BT_WRMS:units = "cm s-1" ;
            BT_WRMS:long_name = "Bottom Velocity Up-Down Root Mean
Square" ;
            BT_WRMS:_FillValue = -999999.f ;
      float BTR1(Pensemble) ;
            BTR1:units = "meters" ;
            BTR1:long_name = "Bottom Range Beam 1" ;
            BTR1:_FillValue = -999999.f ;
      float BTR2(Pensemble) ;
            BTR2:units = "meters" ;
            BTR2:long_name = "Bottom Range Beam 2" ;
            BTR2:_FillValue = -999999.f ;
      float BTR3(Pensemble) ;
            BTR3:units = "meters" ;
            BTR3:long_name = "Bottom Range Beam 3" ;
            BTR3:_FillValue = -999999.f ;
      float BTR4(Pensemble) ;
            BTR4:units = "meters" ;
            BTR4:long_name = "Bottom Range Beam 4" ;
            BTR4:_FillValue = -999999.f ;
      float BTR(Pensemble) ;
            BTR:units = "meters" ;
            BTR:long_name = "Bottom Range Beam" ;
            BTR:_FillValue = -999999.f ;

// global attributes:
            :Creation_Date = "19-Jul-2002" ;
            :Soft_Version = "Version 2.0" ;
            :ADCP_Constructor = "RDI" ;
            :ADCP_Type = "75nb  " ;
            :Cruise = "SEMANE" ;
            :Vessel = "THALASSA     " ;
            :Real_Frequency_in_Khz = 76.8f ;
            :Scale_Factor = 0.25f ;
            :Last_Update = "19-Jul-2002" ;
            :Beam_Angle = 30.f ;
            :ADCP_Angle = 45.f ;
            :Bin_Length = 16.f ;
            :Middle_Bin1_Depth = 24.f ;
```

```
            :Xoff = 6.f ;
            :Nb_mean = 30 ;
            :Pgood_lim = 30 ;
            :Corr_PR = 1 ;
            :Count_to_DB = 0.45f ;
}
```

6.8Structure du fichier campagne NetCDF

```
netcdf cathy {
dimensions:
      Pensemble = 14 ;
      bin = 50 ;
      cte = 1 ;
      DATE_TIME = 14 ;
variables:
      char REFERENCE_DATE_TIME(DATE_TIME) ;
            REFERENCE_DATE_TIME:long_name = "Date of the Julian day 0" ;
            REFERENCE_DATE_TIME:conventions = "YYYYMMDDHHMISS" ;
      double JULD(Pensemble) ;
            JULD:units = "GPS Decimal days since REFERENCE_DATE_TIME" ;
            JULD:long_name = "JULIAN DAYS" ;
            JULD:_FillValue = -999999. ;
            JULD:valid_min = 0 ;
      double JULD_ADCP(Pensemble) ;
            JULD_ADCP:units = "ADCP Decimal days since REFERENCE_DATE_TIME"
;
            JULD_ADCP:long_name = "JULIAN DAYS" ;
            JULD_ADCP:_FillValue = -999999. ;
            JULD_ADCP:valid_min = 0 ;
      char DATE_TIME_UTC(Pensemble, DATE_TIME) ;
            DATE_TIME_UTC:long_name = "ASCII GREGORIAN DAYS AND TIME" ;
            DATE_TIME_UTC:conventions = "YYYYMMDDHHMMSS" ;
      double PLat(Pensemble) ;
            PLat:units = "degrees" ;
            PLat:long_name = "Latitude" ;
            PLat:_FillValue = -999999. ;
            PLat:valid_range = -90., 90. ;
      double PLon(Pensemble) ;
            PLon:units = "degrees" ;
            PLon:long_name = "Longitude" ;
            PLon:_FillValue = -999999. ;
            PLon:valid_range = -180., 180. ;
      double Unav(Pensemble) ;
            Unav:units = "cm/s" ;
            Unav:long_name = "vitesse navire composante est" ;
            Unav:_FillValue = -999999. ;
      double Vnav(Pensemble) ;
            Vnav:units = "cm/s" ;
```

```
      Vnav:long_name = "vitesse navire composante nord" ;
      Vnav:_FillValue = -999999. ;
float Depth(bin) ;
      Depth:units = "m" ;
      Depth:long_name = "Depth (m)" ;
      Depth:_FillValue = -999999.f ;
float PTx(Pensemble) ;
      PTx:units = "degrees" ;
      PTx:long_name = "Processed ADCP Transducer Temperature" ;
      PTx:_FillValue = -999999.f ;
      PTx:valid_range = -5.f, 45.f ;
float PHdg(Pensemble) ;
      PHdg:units = "degrees" ;
      PHdg:long_name = "Processed Heading" ;
      PHdg:_FillValue = -999999.f ;
      PHdg:valid_range = -360.f, 360.f ;
float PPtch(Pensemble) ;
      PPtch:units = "degrees" ;
      PPtch:long_name = "Processed Pitch" ;
      PPtch:_FillValue = -999999.f ;
      PPtch:valid_range = -360.f, 360.f ;
float PRoll(Pensemble) ;
      PRoll:units = "degrees" ;
      PRoll:long_name = "Processed Roll" ;
      PRoll:_FillValue = -999999.f ;
      PRoll:valid_range = -360.f, 360.f ;
short flag_courant(Pensemble, bin) ;
      flag_courant:units = "flag" ;
      flag_courant:long_name = "flag" ;
      flag_courant:_FillValue = -9999s ;
      flag_courant:valid_min = 0s ;
float NSVelAbs(Pensemble, bin) ;
      NSVelAbs:units = "cm s-1" ;
      NSVelAbs:long_name = "Absolute North/South velocity" ;
      NSVelAbs:_FillValue = -999999.f ;
float EWVelAbs(Pensemble, bin) ;
      EWVelAbs:units = "cm s-1" ;
      EWVelAbs:long_name = "Absolute East/West velocity" ;
      EWVelAbs:_FillValue = -999999.f ;
float UDVelAbs(Pensemble, bin) ;
      UDVelAbs:units = "cm s-1" ;
      UDVelAbs:long_name = "Absolute Up/Down velocity" ;
      UDVelAbs:_FillValue = -999999.f ;
float EVel(Pensemble, bin) ;
      EVel:units = "cm s-1" ;
      EVel:long_name = "Mean residual velocity" ;
      EVel:_FillValue = -999999.f ;
float PGood(Pensemble, bin) ;
      PGood:units = "percent %" ;
      PGood:long_name = "Percent Good" ;
      PGood:_FillValue = -999999.f ;
float URMS(Pensemble, bin) ;
      URMS:units = "cm s-1" ;
      URMS:long_name = "Velocity East-West Root Mean Square" ;
      URMS:_FillValue = -999999.f ;
float VRMS(Pensemble, bin) ;
      VRMS:units = "cm s-1" ;
      VRMS:long_name = "Velocity South-North Root Mean Square" ;
      VRMS:_FillValue = -999999.f ;
float WRMS(Pensemble, bin) ;
      WRMS:units = "cm s-1" ;
```

```
            WRMS:long_name = "Velocity Up-Down Root Mean Square" ;
            WRMS:_FillValue = -999999.f ;
      float ERMS(Pensemble, bin) ;
            ERMS:units = "cm s-1" ;
            ERMS:long_name = "Residual Velocity Root Mean Square" ;
            ERMS:_FillValue = -999999.f ;
      float PECI(Pensemble, bin) ;
            PECI:units = "counts" ;
            PECI:long_name = "Mean Echo intensity" ;
            PECI:_FillValue = -999999.f ;
      float BT_NSVel(Pensemble) ;
            BT_NSVel:units = "cm s-1" ;
            BT_NSVel:long_name = "Bottom track North/South velocity" ;
            BT_NSVel:_FillValue = -999999.f ;
      float BT_EWVel(Pensemble) ;
            BT_EWVel:units = "cm s-1" ;
            BT_EWVel:long_name = "Bottom track East/West velocity" ;
            BT_EWVel:_FillValue = -999999.f ;
      float BT_UDVel(Pensemble) ;
            BT_UDVel:units = "cm s-1" ;
            BT_UDVel:long_name = "Bottom track Up/Down velocity" ;
            BT_UDVel:_FillValue = -999999.f ;
      float BTR(Pensemble) ;
            BTR:units = "meters" ;
            BTR:long_name = "Bottom Range Beam" ;
            BTR:_FillValue = -999999.f ;
      float U_maree(Pensemble) ;
            U_maree:units = "cm/s" ;
            U_maree:long_name = "U tide" ;
            U_maree:_FillValue = -999999.f ;
            U_maree:valid_min = 0.f ;
      float V_maree(Pensemble) ;
            V_maree:units = "cm/s" ;
            V_maree:long_name = "V tide" ;
            V_maree:_FillValue = -999999.f ;
            V_maree:valid_min = 0.f ;
      float H_maree(Pensemble) ;
            H_maree:units = "m" ;
            H_maree:long_name = "H tide" ;
            H_maree:_FillValue = -999999.f ;
            H_maree:valid_min = 0.f ;
      float U_corrigee(Pensemble, bin) ;
            U_corrigee:units = "cm s-1" ;
            U_corrigee:long_name = "Absolute velocity tide corrected" ;
            U_corrigee:_FillValue = -999999.f ;
      float V_corrigee(Pensemble, bin) ;
            V_corrigee:units = "cm s-1" ;
            V_corrigee:long_name = "Absolute velocity tide corrected" ;
            V_corrigee:_FillValue = -999999.f ;

// global attributes:
            :Creation_Date = "19-Jul-2002" ;
            :Soft_Version = "Version 2.0" ;
            :ADCP_Constructor = "RDI" ;
            :ADCP_Type = "75nb  " ;
            :Cruise = "SEMANE" ;
            :Vessel = "THALASSA     " ;
            :Real_Frequency_in_Khz = 76.8f ;
            :Scale_Factor = 0.25f ;
            :Last_Update = "19-Jul-2002" ;
            :Beam_Angle = 30.f ;
```

```
                :ADCP_Angle = 45.f ;
                :Bin_Length = 16.f ;
                :Middle_Bin1_Depth = 24.f ;
                :Xoff = 6.f ;
                :Corr_PR = 1 ;
                :Nb_ens_moy = 30. ;
                :HeadMis = 0. ;
                :PitchMis = 0. ;
                :Amplitude = 1. ;
                :Nb_a_Moyenner = 10.f ;
                :Nb_Ecart_Moyenne = 50.f ;
}
```

6.9Structure du fichier section NetCDF

```
      netcdf section_2_ast_f0 {
dimensions:
      Pensemble = 1885 ;
      bin = 50 ;
      cte = 1 ;
      nbre_section = 10 ;
      DATE_TIME = 14 ;
variables:
      double INDICE(nbre_section) ;
            INDICE:units = "indice" ;
            INDICE:long_name = "indice" ;
               indique le début des données associées aux diverses sections
            INDICE:_FillValue = -999999. ;
            INDICE:valid_min = 0 ;
      char REFERENCE_DATE_TIME(DATE_TIME) ;
            REFERENCE_DATE_TIME:long_name = "Date of the Julian day 0" ;
            REFERENCE_DATE_TIME:conventions = "YYYYMMDDHHMISS" ;
      double JULD(Pensemble) ;
            JULD:units = "Decimal days since REFERENCE_DATE_TIME" ;
            JULD:long_name = "JULIAN DAYS" ;
            JULD:_FillValue = -999999. ;
            JULD:valid_min = 0 ;
      double SecLat(Pensemble) ;
            SecLat:units = "degrees" ;
            SecLat:long_name = "Latitude" ;
            SecLat:_FillValue = -999999. ;
            SecLat:valid_range = -90., 90. ;
      double SecLon(Pensemble) ;
            SecLon:units = "degrees" ;
            SecLon:long_name = "Longitude" ;
            SecLon:_FillValue = -999999. ;
            SecLon:valid_range = -180., 180. ;
      float Depth(bin) ;
            Depth:units = "m" ;
            Depth:long_name = "Depth (m)" ;
            Depth:_FillValue = -999999.f ;
      float NSVelAbs(Pensemble, bin) ;
            NSVelAbs:units = "cm s-1" ;
            NSVelAbs:long_name = "Absolute North/South velocity" ;
            NSVelAbs:_FillValue = -999999.f ;
      float EWVelAbs(Pensemble, bin) ;
            EWVelAbs:units = "cm s-1" ;
            EWVelAbs:long_name = "Absolute East/West velocity" ;
            EWVelAbs:_FillValue = -999999.f ;
      float UDVelAbs(Pensemble, bin) ;
            UDVelAbs:units = "cm s-1" ;
            UDVelAbs:long_name = "Absolute Up/Down velocity" ;
            UDVelAbs:_FillValue = -999999.f ;
      float URMS(Pensemble, bin) ;
            URMS:units = "cm s-1" ;
            URMS:long_name = "Velocity East-West Root Mean Square" ;
            URMS:_FillValue = -999999.f ;
      float VRMS(Pensemble, bin) ;
            VRMS:units = "cm s-1" ;
```

```
            VRMS:long_name = "Velocity South-North Root Mean Square" ;
      float WRMS(Pensemble, bin) ;
            WRMS:units = "cm s-1" ;
            WRMS:long_name = "Velocity Up-Down Root Mean Square" ;
            WRMS:_FillValue = -999999.f ;
      float PECI(Pensemble, bin) ;
            PECI:units = "countsse" ;
            PECI:long_name = "Mean Echo intensity" ;
            PECI:_FillValue = -999999.f ;

// global attributes:
            :CREATION_DATE = "05-Jun-2002" ;
            :INST_TYPE = "RD Instruments ADCP" ;
            :INST_MODEL = "narrow band" ;
            :PROG_CMNT1 = "Section file" ;
            :delta_distance = 2. ;
}
```

6.10Structure du fichier station NetCDF

```
netcdf station_0 {
dimensions:
      Pensemble = 66 ;
      bin = 50 ;
      cte = 1 ;
      nbre_station = 66 ;
      DATE_TIME = 14 ;
variables:
      double INDICE(nbre_station) ;
            INDICE:units = "indice" ;
            INDICE:long_name = "indice" ;
            Indique le début des données associées aux diverses stations
            INDICE:_FillValue = -999999. ;
            INDICE:valid_min = 0 ;
      char REFERENCE_DATE_TIME(DATE_TIME) ;
            REFERENCE_DATE_TIME:long_name = "Date of the Julian day 0" ;
            REFERENCE_DATE_TIME:conventions = "YYYYMMDDHHMMSS" ;
      double JULD(Pensemble) ;
            JULD:units = "Decimal days since REFERENCE_DATE_TIME" ;
            JULD:long_name = "JULIAN DAYS" ;
            JULD:_FillValue = -999999. ;
            JULD:valid_min = 0 ;
      double StaLat(Pensemble) ;
            StaLat:units = "degrees" ;
            StaLat:long_name = "Latitude" ;
            StaLat:_FillValue = -999999. ;
            StaLat:valid_range = -90., 90. ;
      double StaLon(Pensemble) ;
            StaLon:units = "degrees" ;
            StaLon:long_name = "Longitude" ;
            StaLon:_FillValue = -999999. ;
            StaLon:valid_range = -180., 180. ;
      float Depth(bin) ;
            Depth:units = "m" ;
            Depth:long_name = "Depth (m)" ;
            Depth:_FillValue = -999999.f ;
      float NSVelAbs(Pensemble, bin) ;
            NSVelAbs:units = "cm s-1" ;
            NSVelAbs:long_name = "Absolute North/South velocity" ;
            NSVelAbs:_FillValue = -999999.f ;
      float EWVelAbs(Pensemble, bin) ;
            EWVelAbs:units = "cm s-1" ;
            EWVelAbs:long_name = "Absolute East/West velocity" ;
            EWVelAbs:_FillValue = -999999.f ;
      float UDVelAbs(Pensemble, bin) ;
            UDVelAbs:units = "cm s-1" ;
            UDVelAbs:long_name = "Absolute Up/Down velocity" ;
            UDVelAbs:_FillValue = -999999.f ;
      float URMS(Pensemble, bin) ;
            URMS:units = "cm s-1" ;
            URMS:long_name = "Velocity East-West Root Mean Square" ;
            URMS:_FillValue = -999999.f ;
      float VRMS(Pensemble, bin) ;
            VRMS:units = "cm s-1" ;
            VRMS:long_name = "Velocity South-North Root Mean Square" ;
```

```
      float WRMS(Pensemble, bin) ;
            WRMS:units = "cm s-1" ;
            WRMS:long_name = "Velocity Up-Down Root Mean Square" ;
            WRMS:_FillValue = -999999.f ;
      float PECI(Pensemble, bin) ;
            PECI:units = "countsse" ;
            PECI:long_name = "Mean Echo intensity" ;
            PECI:_FillValue = -999999.f ;

// global attributes:
            :CREATION_DATE = "22-Jul-2002" ;
            :INST_TYPE = "RD Instruments ADCP" ;
            :INST_MODEL = "narrow band" ;
            :PROG_CMNT1 = "Station file" ;
            :duree_moyenne = 0. ;
}
```

6.11Calcul de la marée

TPXO.3
The OSU TOPEX/Poseidon Cross-Over Global Inverse Solution, Ver. 3.1

TPXO.3 is a global model of ocean tides, which best-fits, in a least-squares sense, the Laplace Tidal Equations and cross-over data from the first 40 TOPEX/Poseidon orbit cycles. The methods used to compute the model, are described in detail by Egbert, Bennett, and Foreman [1994; EBF]. TPXO.3 is an improved version of TPXO.1 (TPXO.2), the model originally presented in EBF. The tides are provided as complex amplitudes of earth-relative sea-surface elevation for eight primary harmonic constituents (M2, S2, N2, K2, K1, O1, P1, Q1), on a 512x256 grid between latitudes of 79.71 degrees S and 69.71 degrees N.

A tidal synthesis program ("tide"), which computes tidal corrections for a given time and location, is provided with the model file. This program applies bilinear interpolation to the gridded model file of complex harmonic constants, at the desired location. The tidal synthesis program uses interpolation of the tidal admittances in the diurnal and semi-diurnal bands to include 9 additional minor constituents (2N2,MU2,NU2,L2,T2,J1,NO1,OO1,RHO1). The synthesis program also adds the long period constituents MF,MM,SSA using the standard equilibrium forms. All of the usual nodal corrections have been applied. Note that the tides output by "tide" are geocentric, or altimetric tides and are suitable for direct use as corrections for TPOEX/Poseidon or other altimetric data.

TPXO.3 reduced the RMS TP cross-over difference from 0.4524 m (no tidal correction), to 0.0937 m. For comparison, the RMS cross-over difference for the other tidal corrections on the TP GDR were 0.1199 m for SCH80, and 0.1120 for CR91. TPXO.3 thus resulted in variance reductions of 39% and 30% relative to SCH80 and CR91, respectively. Note that all models were compared at the same set of cross-overs, and that the quoted RMS values are unweighted. Smaller RMS values are obtained for all models when the comparison is restricted to open ocean (e.g., greater than 1000 m depth) cross-overs. For the 78 pelagic validation gauges proposed by Cartwright and Ray, the RMS misfits of TPXO.3 for the four constituents were 0.0275 m (M2); 0.0165 m (S2); 0.0125 m (O1); 0.0161 m (K1).

Ref: Egbert, G.D., A.F. Bennett, and M.G.G. Foreman,
TOPEX/POSEIDON Tides Estimated Using a Global Inverse Model,
J. Geophys. Res., In press.

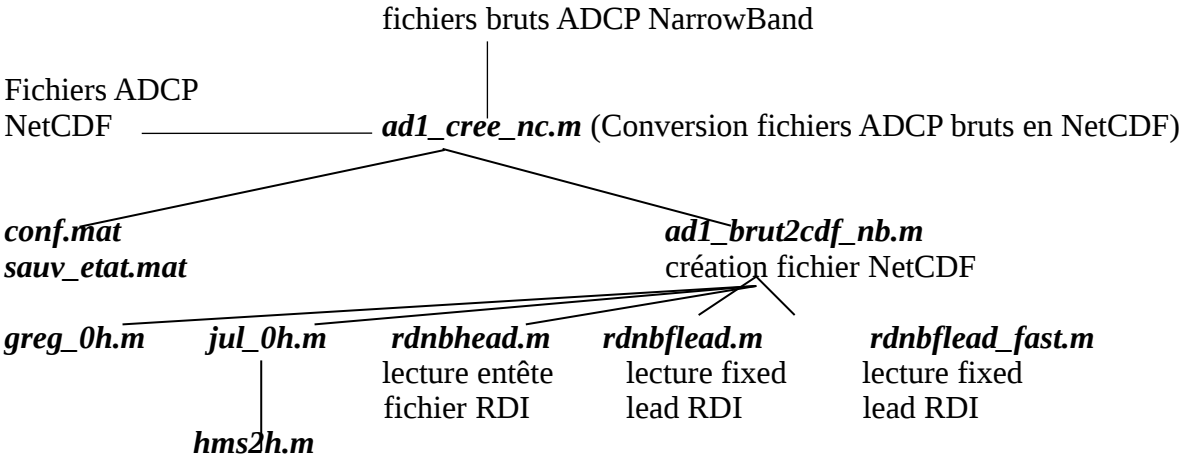
----------> email contact:: egbert@oce.orst.edu <----------

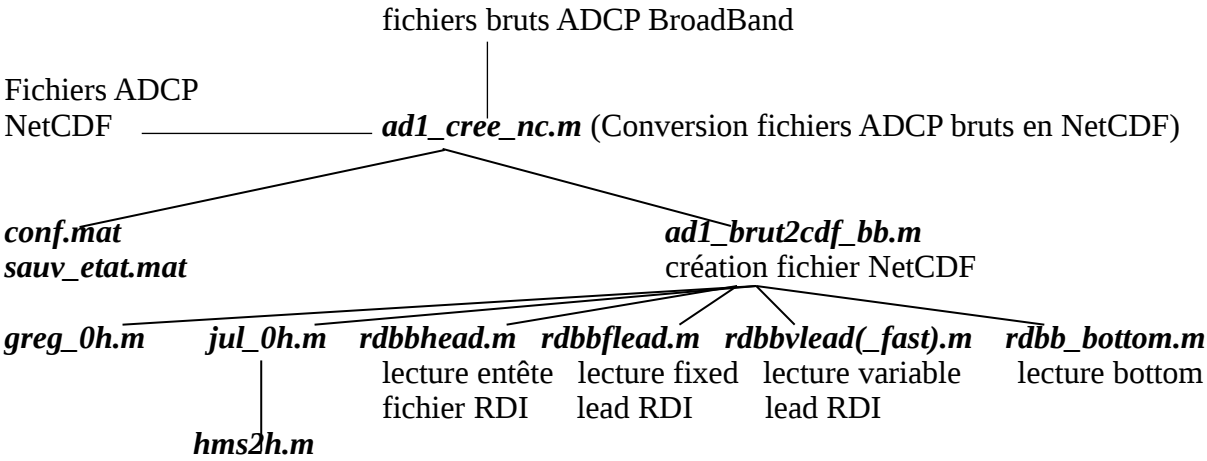
7.Organigrammes

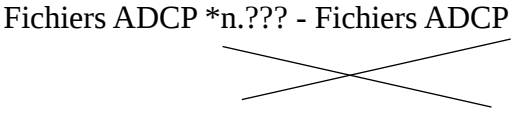
Tous les scripts faisant appel aux fichiers NetCDF utilisent la librairie NetCDF ainsi que mexcdf.

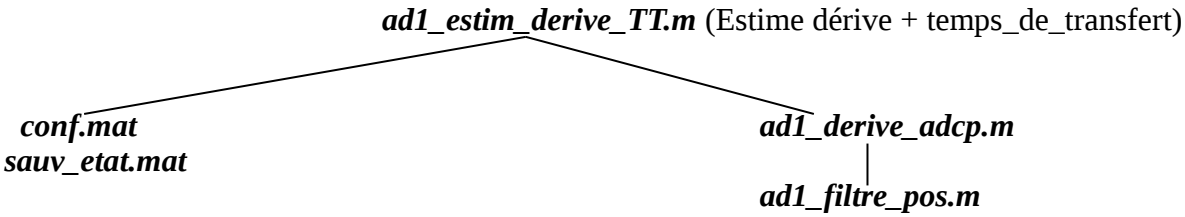
Syntaxe

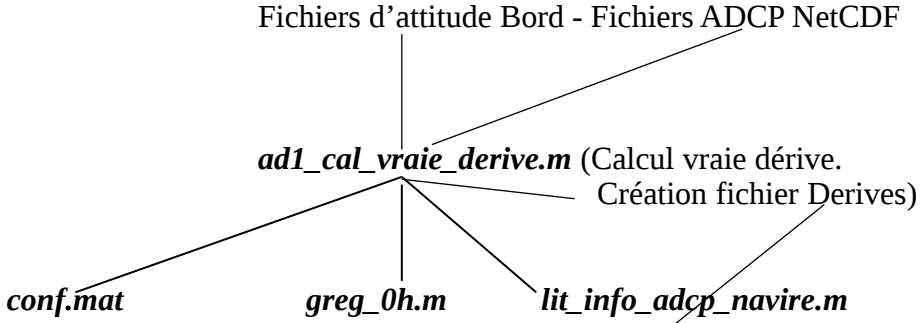
script1.m ——— script2 .m : Le script1.m fait appel au script2.m.
fichier ——— script.m : Le script.m utilise fichier.
fichier ——— script.m : Le script.m crée fichier.
fichier ——— script.m : Le script.m utilise et modifie fichier.

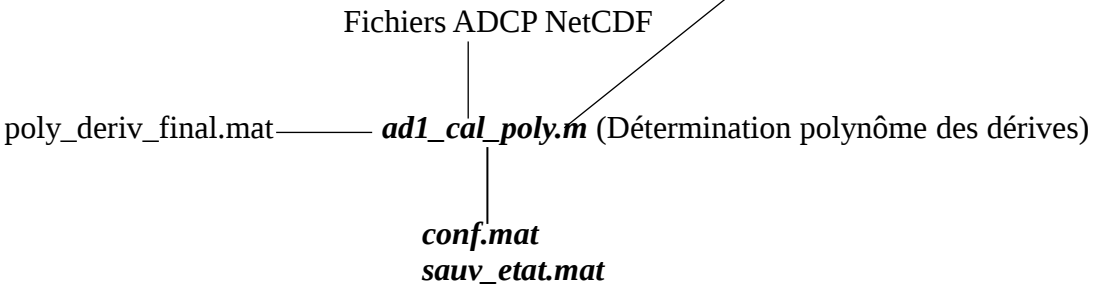


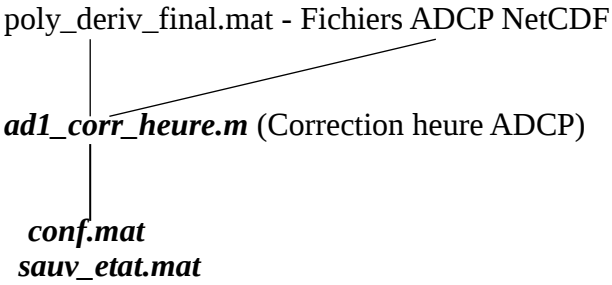


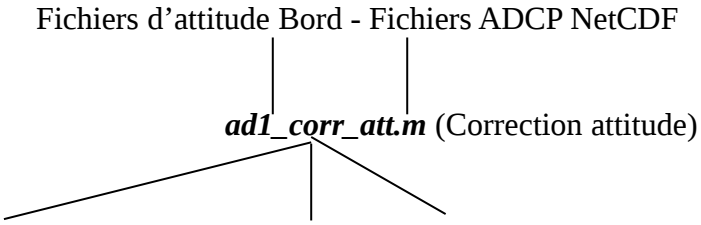



ad1_estim_derive_TT.m (Estime dérive + temps_de_transfert)
conf.mat
sauv_etat.mat
ad1_derive_adcp.m
ad1_filtre_pos.m

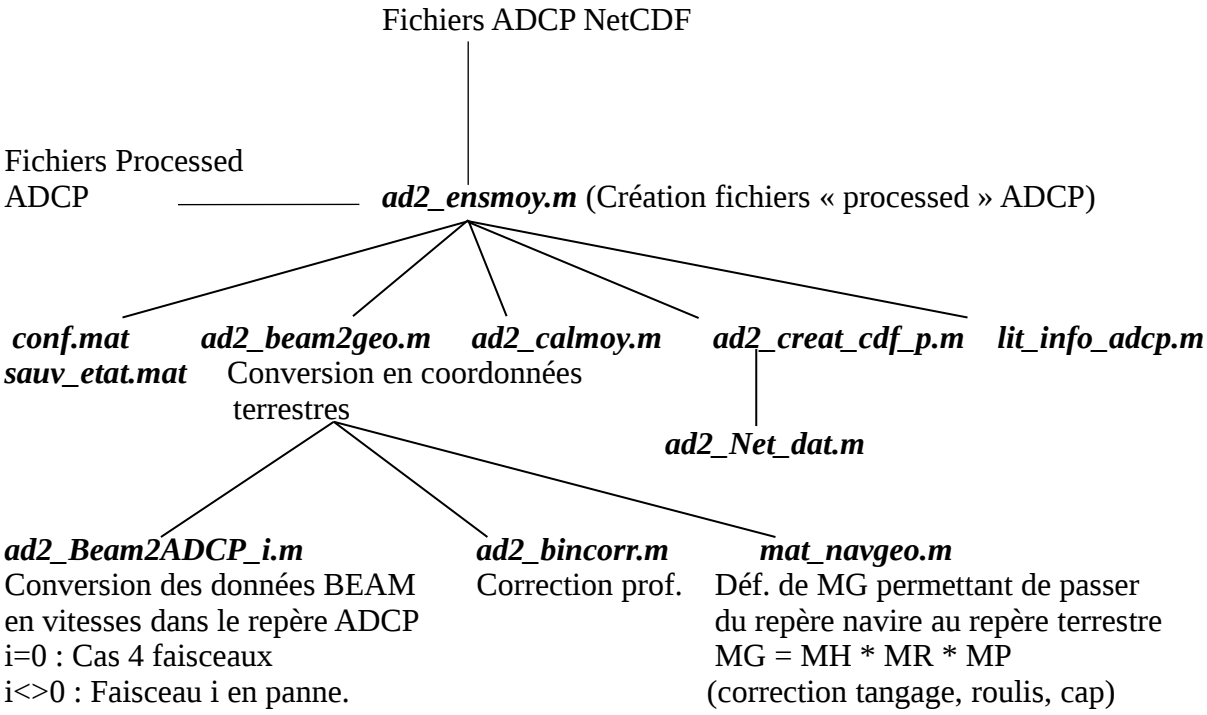

Fichiers d'attitude Bord - Fichiers ADCP NetCDF
ad1_cal_vraie_derive.m (Calcul vraie dérive.
Création fichier Derives)
conf.mat
greg_0h.m
lit_info_adcp_navire.m

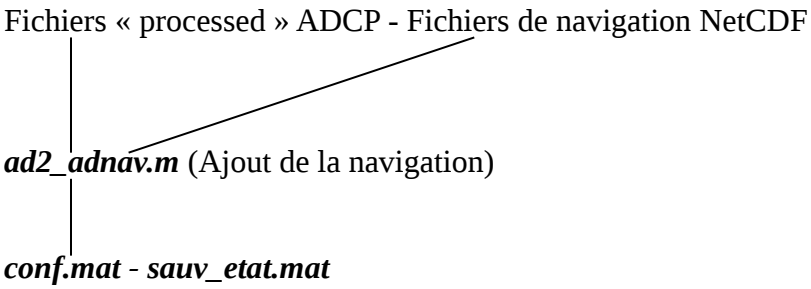

Fichiers ADCP NetCDF
poly_deriv_final.mat
ad1_cal_poly.m (Détermination polynôme des dérives)
conf.mat
sauv_etat.mat

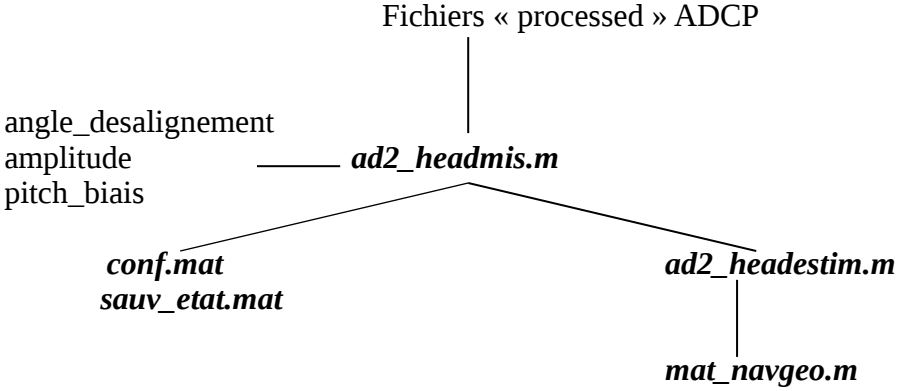

poly_deriv_final.mat - Fichiers ADCP NetCDF
ad1_corr_heure.m (Correction heure ADCP)
conf.mat
sauv_etat.mat

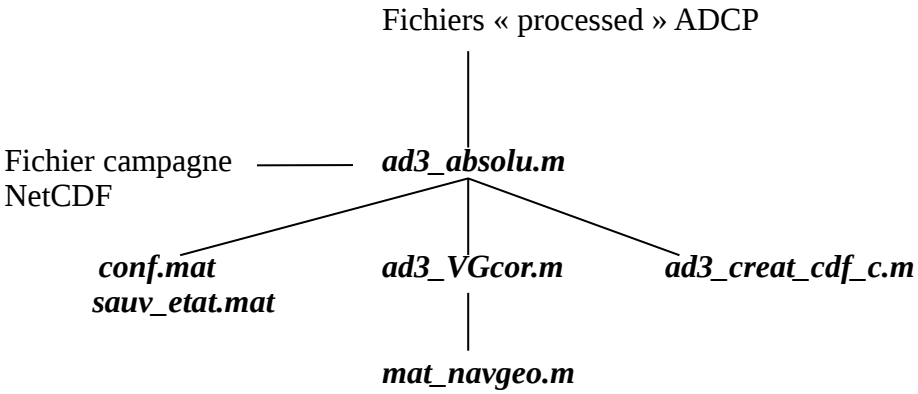

Fichiers d'attitude Bord - Fichiers ADCP NetCDF
ad1_corr_att.m (Correction attitude)

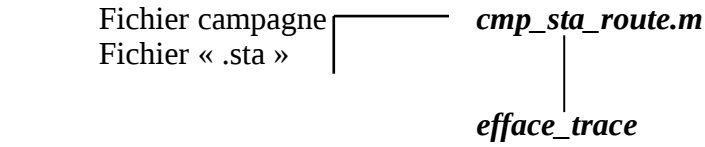
conf.mat ***ad1_atti_pl0.m*** ***lit_info_adcp_navire.m***
sauv_etat.mat

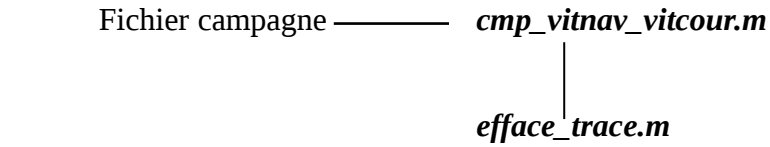






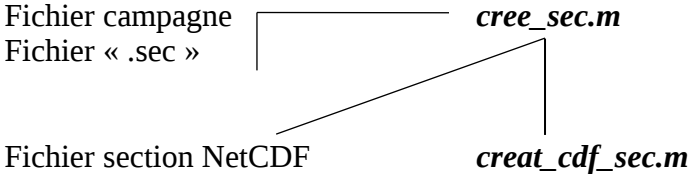

Fichiers « processed » ADCP
Fichier campagne
NetCDF
ad3_absolu.m
conf.mat
sauv_etat.mat
ad3_VGcor.m
ad3_creat_cdf_c.m
mat_navgeo.m


Fichier campagne
Fichier « .sta »
cmp_sta_route.m
efface_trace

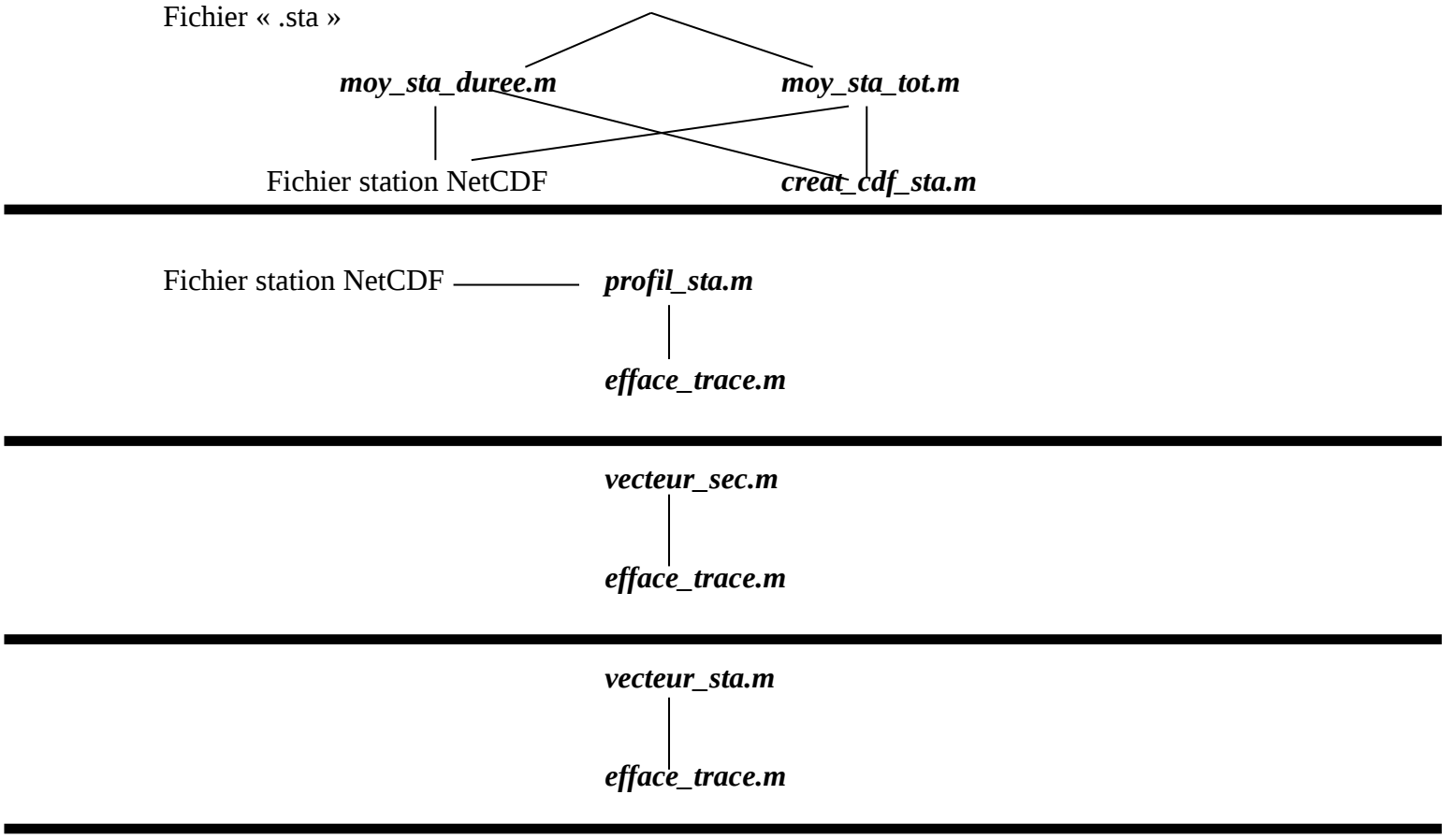

Fichier campagne
cmp_vitnav_vitcour.m
efface_trace.m


Fichier section NetCDF
contour_sec.m
efface_trace.m

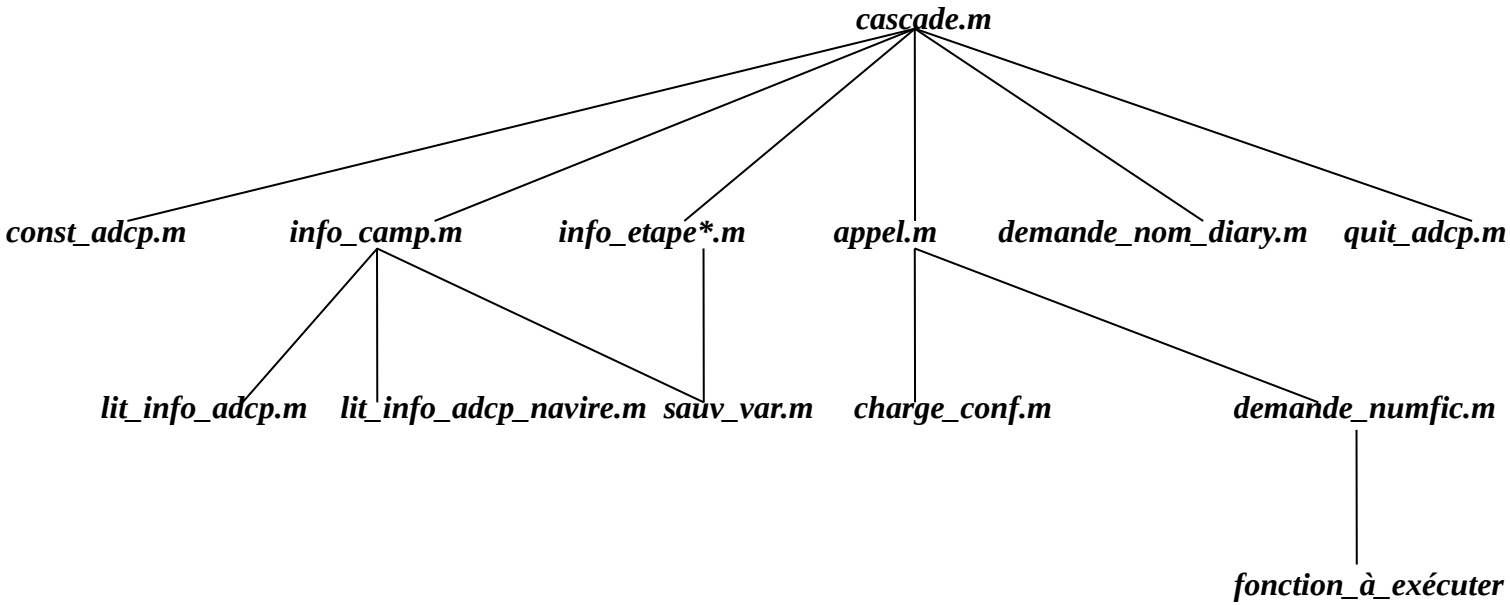

Fichier station NetCDF
contour_sta.m
efface_trace.m


Fichier campagne
Fichier « .sec »
cree_sec.m
Fichier section NetCDF
creat_cdf_sec.m


Fichier campagne
cree_sta.m

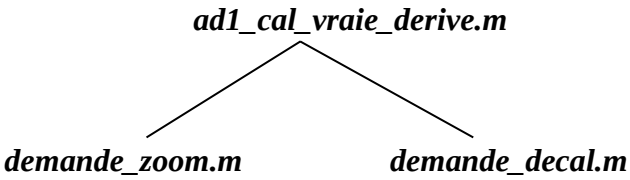

Fichier « .sta »
moy_sta_duree.m
moy_sta_tot.m
Fichier station NetCDF
creat_cdf_sta.m
Fichier station NetCDF
profil_sta.m
efface_trace.m
vecteur_sec.m
efface_trace.m
vecteur_sta.m
efface_trace.m

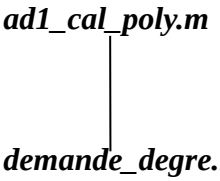
Organigrammes généraux des interfaces

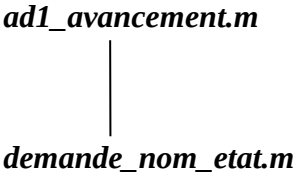


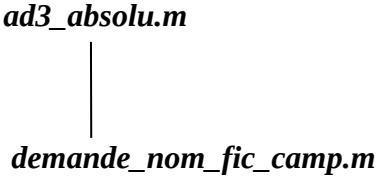
fonction_à_exécuter est un des scripts décrits dans le paragraphe 4 (***ad1_cree_nc.m, ad1_estime_derive_TT.m***, ...etc).
A noter que :

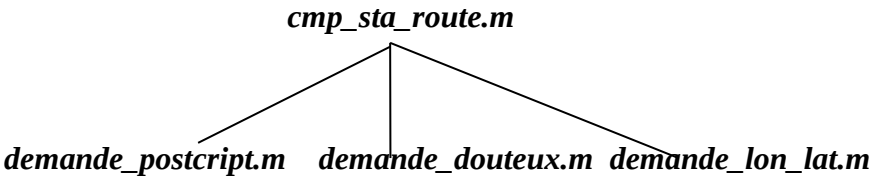
- toutes les interfaces font appel à ***global_adcp.m*** (définition des variables globales).

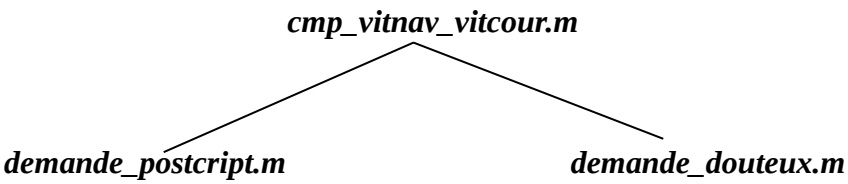

ad1_cal_vraie_derive.m
demande_zoom.m
demande_decal.m

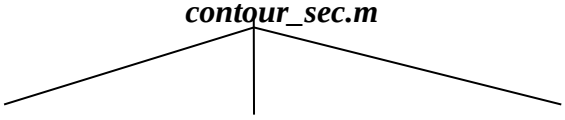

ad1_cal_poly.m
demande_degre.m


ad1_avancement.m
demande_nom_etat.m

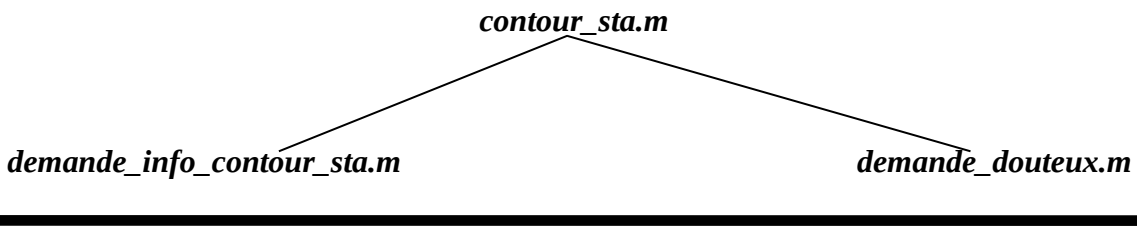

ad3_absolu.m
demande_nom_fic_camp.m

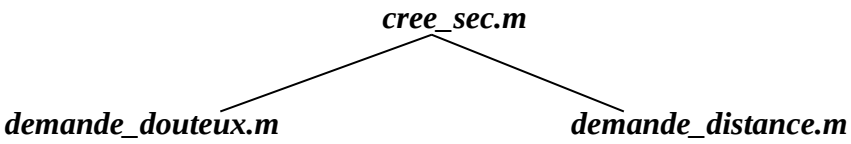

cmp_sta_route.m
demande_postcript.m
demande_douteux.m
demande_lon_lat.m

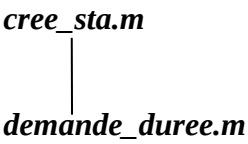

cmp_vitnav_vitcour.m
demande_postcript.m
demande_douteux.m

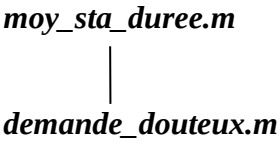

contour_sec.m

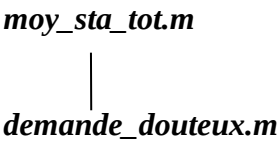
demande_info_contour_sec.m *demande_contour.m* *demande_pcolor.m*

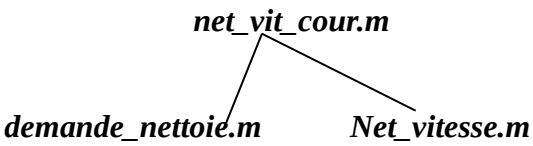


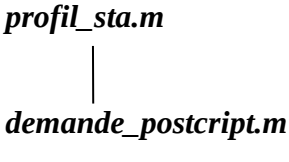






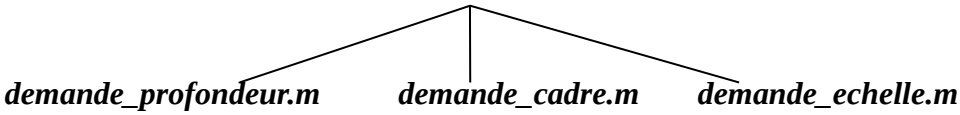


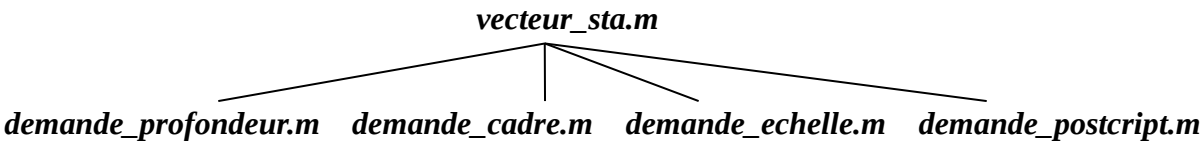




---

vecteur_sec.m


demande_profondeur.m
demande_cadre.m
demande_echelle.m


vecteur_sta.m
demande_profondeur.m
demande_cadre.m
demande_echelle.m
demande_postcript.m