GESTION DES NAMELISTS DE MITRAILLETTE, D'OLIVE/VORTEX ET DE L'OPERATIONNEL: CYCLE 46.

YESSAD K. (METEO-FRANCE/CNRM/GMAP/ALGO)

April 9, 2018

Résumé:
Cette documentation est un memorandum récapitulant les principales actions à effectuer concernant la maintenance des namelists de "mitraillette", ainsi que celles de l'opérationnel, qui vont dans les environnements OLIVE et VORTEX.

Contents

1 Introduction.

Cette documentation est un memorandum récapitulant les principales actions à effectuer concernant la maintenance des namelists de “mitraillette”, ainsi que celles de l’opérationnel, qui vont dans les environnements d’OLIVE et de VORTEX. Cela concerne:

- Toutes les namelists de “mitraillette”.
- Toutes les namelists d’OLIVE et de VORTEX, et de l’environnement oper/double.

Il y a des remises à jour de namelists à faire dans les cas suivants:

- achèvement d’un nouveau cycle principal ou intermédiaire: il faut alors adapter les jeux de namelists à ce nouveau cycle.
- lancement d’une chaîne en double: il faut alors créer les jeux de namelists adéquats.
- basculement opérationnel d’une chaîne en double.

2 Normes de présentation des namelists.

∗ **Normes:** Afin de pouvoir comparer facilement des namelists, les mettre à jour par des outils automatiques, et assurer leur portabilité entre différentes machines il est important que les namelists respectent un certain nombre de normes de présentation, qu’on peut lister comme suit:

- règle 01: tous les éléments sont dans l’ordre alphabétique (celui qui est obtenu à l’issue de l’application de la procédure “xpnam”).
- règle 02: tous les éléments existants sont références (y compris tous les éléments en NEM...), même s’ils ne sont pas lus dans la configuration courante.
- règle 03: chaque élément référencé n’apparaît qu’une seule fois.
- règle 04: il n’y a pas de référence à des éléments obsolètes.
- règle 05: pour les variables logiques, on écrit toujours .TRUE. ou .FALSE.; ainsi on écrit “LREGETA=.FALSE.,”, mais jamais “LREGETA=.false.,”, “LREGETA=.f.,”, “LREGETA=.F.,”, “LREGETA=F,”.
- règle 06: on n’utilise que des majuscules; ainsi on écrit “NRADFR=-1,”, mais jamais “nradfr=-1,”.
- règle 07: chaque ligne s’achève par une virgule.
- règle 08: jamais de blanc avant ou après le “égal”, ni avant la virgule finale; ainsi on écrit “NRADFR=-1,”, mais jamais “NRADFR =-1,”, “NRADFR= -1,” ou “NRADFR=-1 ,” (cela peut poser des problèmes de portabilité et cela complique la gestion semi-automatique de la remise à jour des namelists).
- règle 09: dans chaque élément, il y a une et une seule variable par ligne.
- règle 10: chaque variable référencée n’apparaît qu’une seule fois.
- règle 11: aucune variable obsolète ne doit apparaître.
- règle 12: les caractères blancs sont permis devant le nom des variables, mais toutes les variables doivent être alignées; la norme actuelle est d’indenter d’un blanc devant “&NAM...” et “slash”, et de 3 blancs devant chaque variable.
- règle 13: pour les tableaux dont on remplit les éléments un à un (exemple NFPRGRI), les éléments apparaissent dans l’ordre croissant et de façon contiguë (en gros on n’a pas NFPRGRI(2) entre NFPRGRI(29) et NFPRGRI(30), et on ne trouve pas d’autre variable entre NFPRGRI(29) et NFPRGRI(30)).
- règle 14: concerne ce qui est autorisé ou interdit quand on remplit des tableaux.
  Les syntaxes suivantes sont autorisées:

```
- syntax 1
 NARRAY=1,2,3,9,
- syntax 2
 NARRAY(1:4)=1,2,3,9,
- syntax 3
 NARRAY(1:2)=1,2,
 NARRAY(3:4)=3,9,
- syntax 4
```

```
NARRAY(1)=1,
NARRAY(2)=2,
NARRAY(3)=3,
NARRAY(4)=9,
```

En revanche la syntaxe suivante est interdite:

```
NARRAY=1,2,
       3,9,
```

- règle 15: la valeur affectée à une variable doit correspondre au type de la variable. Par exemple, TSTEP étant un réel, on écrit TSTEP=600. (avec un point).
- règle 16: Variables réelles avec des décimales: les décimales ne s'arrêtent jamais sur un zéro. Ainsi par exemple on écrit 1. et non 1.00; 0. et non 0.000; 0.394 et non 0.394000, 3.14 et non 3.1400; 1.D+07 et non 1.00D+07; 4.07D+03 et non 4.070D+03. Pour des variables dont l'ordre de grandeur s'éloigne notamment de 1. on utilise une notation exponentielle. On écrit 7.4D+07 et non 74000000.; 1.D-09 et non 0.000000001.

Il y a quelques namelists à usage particulier qui ont leurs propres éléments de namelists.

Il faut noter que toutes les namelists cy46 et al46 sur BEAUFIX sont censées respecter ces normes et que dorénavant on ne doit plus importer de namelists qui ne seraient pas aux normes: si une nouvelle application requiert une nouvelle namelist spécifique celle-ci devra partir d'une namelist existante aux normes.

∗ **Outils de normalisation:** Un certain nombre d'outils de gestion de namelists sont décrits dans la documentation (IDMITS). Voir également la section 5 du présent document. La procédure `tnt.py` (sous VORTEX) qui permet de mettre à jour les namelists fait également la mise aux normes sur presque toutes les règles citées ci-dessus.

3 Quelques informations supplémentaires sur certaines variables.

Ces recommandations ne prétendent pas être exhaustives, mais elles sont utiles au vu d'erreurs ou d'oublis souvent constatés dans les namelists d'OLIVE et de l'environnement opérationnel.

3.1 Physique: pour une physique donnée, les mêmes variables dans le même ordre partout.

Même s'il n'y a aucune règle instituant un ordre particulier pour mettre les variables (cela pourrait être par exemple l'ordre alphabétique), il est souhaitable que toutes les namelists utilisant la même physique spécifient les mêmes variables rangées dans le même ordre.

Les éléments de namelist concernés sont: NAERAD, NAMARPHY, NAMCVMNH, NAMPARAR, NAMPHMSE, NAMPHY, NAMPHY0, NAMPHY1, NAMPHY2, NAMPHY3, NAMPHYDS, NAMRCOEF, NAMSIMPHL, NAMTOPH. NAMPARAR n'est utilisée que pour une micro-physique type AROME; NAMPHMSE n'est utilisée que si la physique de surface utilise SURFEX.

Pour des tâches adiabatiques:

- NAERAD doit contenir l'information suivante: LRRTM=.FALSE., LSRTM=.FALSE., NMCICA=0
- NAMARPHY doit contenir l'information suivante: LMPA=.FALSE., LMSE=.FALSE.
- NAMPHY doit contenir l'information suivante: LMPHYS=.FALSE.
- NAMCVMNH, NAMPHMSE, NAMPHY0, NAMPHY1, NAMPHY2, NAMPHY3, NAMPHYDS, NAMRCOEF, NAMSIMPHL, NAMTOPH doivent rester vides.

Pour des tâches de configuration 601 utilisant de la physique simplifiée dite de Buizza, NAMRCOEF et NAMSIMPHL doivent rester vides.

3.2 Quelques informations sur certaines variables utilisées dans le noyau dynamique.

∗ NAMARG:

Lorsqu'on tourne avec NSUPERSEDE=1, ce qui est généralement le cas à METEO-FRANCE, cet élément de namelist doit être décrit de la façon la plus exhaustive possible, et on doit au moins y trouver (si possible dans le même ordre dans toutes les namelists), les variables suivantes: CNMEXP, NCONF, LELAM, LECMWF, LSLAG, NSUPERSEDE.

∗ Espacement des niveaux η:

Ils sont contrôlés par les variables LREGETA et LVFE_REGETA dans NAMCT0. Lorsque l'une de ces deux variables est spécifiée, les deux doivent obligatoirement l'être.

∗ Discrétisation verticale:

- L'information est fournie par les variables LVERTFE, NVFE_TYPE, LAPRXPK, NDLNPR, réparties entre NAMCVER et NAMDYNA.
- Jobs hydrostatiques utilisant des différences finies sur la verticale: LVERTFE=.FALSE., NVFE_TYPE=0, LAPRXPK=.FALSE., NDLNPR=0 .
- Jobs hydrostatiques utilisant des éléments finis sur la verticale: LVERTFE=.TRUE., NVFE_TYPE=3, LAPRXPK=.TRUE., NDLNPR=0 .
- Jobs non-hydrostatiques (NHEE, cad pleinement élastiques) utilisant des différences finies sur la verticale: LVERTFE=.FALSE., NVFE_TYPE=0, NDLNPR=1 (inutile de spécifier LAPRXPK dans ce cas).

∗ Tâches utilisant un schéma d'advection de type SL2TL (schéma semi-Lagrangien à deux niveaux temporels):

- NAMARG doit spécifier LSLAG=.TRUE.
- NAMCT0 doit spécifier LTWOTL=.TRUE.
- NAMDYNA doit spécifier les valeurs de LNESC, LNESCT et LNESCV.
- NAMDYNA doit spécifier les valeurs de LSETTLS, LSETTLST et LSETTLSV.
- Les valeurs des LSETTLS.. et LNESC.. s'excluent (par exemple on ne peut pas avoir à la fois LSETTLS=.TRUE. et LNESC=.TRUE.) et dépendent de si on met un schéma itératif (valeur de LPC_FULL) ou non.
- En hydrostatique, où on n'active pas de schéma itératif (LPC_FULL=.FALSE.), on a toujours LSETTLS=.TRUE., LSETTLST=.TRUE., LSETTLSV=.TRUE., LNESC=.FALSE., LNESCT=.FALSE. et LNESCV=.FALSE..
- Lorsqu'on active LPC_FULL=.TRUE. avec LPC_CHEAP=.FALSE., on a toujours LSETTLS=.FALSE., LSETTLST=.FALSE., LSETTLSV=.FALSE., LNESC=.TRUE., LNESCT=.TRUE. et LNESCV=.TRUE. .
- Lorsqu'on active LPC_FULL=.TRUE. avec LPC_CHEAP=.TRUE., on a toujours LSETTLS=.FALSE., LSETTLST=.TRUE., LSETTLSV=.TRUE., LNESC=.TRUE., LNESCT=.FALSE. et LNESCV=.FALSE. .
- En complément des six variables précédentes il est recommandé de spécifier LELTRA=.FALSE. dans NAMDYNA.
- NAMDYN doit spécifier NTLAG=3, NVLAG=3, NWLAG=3 (et aussi NSVDLAG=3, NSPDLAG=3 en non-hydrostatique).
- En général on utilise LADVF=.TRUE., donc NAMDYN doit spécifier cela.
- NAMDYN doit spécifier la valeur de RW2TLFF.
- En général on utilise RCMSLP0=1., donc NAMDYN doit spécifier cela.
- Il est recommandé de spécifier VESL=0. et XIDT=0. dans NAMDYN, pour les tâches qui n'activent pas de décentrage.
- Il n'est pas indispensable de spécifier les valeurs (nulles) de REPS1, REPS2, REPSM1, REPSM2, REPSP1.

∗ **Tâches n'utilisant pas de schéma d'advection:**

- On doit au moins trouver les informations suivantes dans les namelists correspondantes: LSLAG=.FALSE., LTWOTL=.FALSE.
- On n'a pas besoin de référencer les autres variables mentionnées au paragraphe précédent.

3.3 Gestion des GFL.

- Cela concerne essentiellement l'élement de namelist NAMGFL.
- Certaines incohérences vont générer un ABOR1 en set-up.
- Les attributs LPT et LPC ne doivent JAMAIS être mentionnés dans NAMGFL.
- NCOUPLING différent de 0, ainsi que la spécification de REFVALC, n'ont de sens que dans un modèle à domaine limité.
- Pour les GFL point de grille, LADV=.TRUE. ne peut être activé qu'en semi-lagrangien.
- LQM=.TRUE., LQMH=.TRUE., LSLHD=.TRUE., LCOMAD=.TRUE. n'ont de sens qu'en semi-lagrangien et pour des variables advectées (LADV=.TRUE.).
- LSLHD_GFL=.TRUE. dans NAMDYNA si et seulement si l'un au moins des GFL a LSLHD=.TRUE. dans NAMGFL.
- LCOMAD_GFL=.TRUE. dans NAMDYNA si et seulement si l'un au moins des GFL a LCOMAD=.TRUE. dans NAMGFL.
- LINTLIN=.TRUE. impose de mettre LSLHD=.FALSE. .

4 Jeux de namelists.

4.1 Namelists de "mitraillette".

Tout est sous l'environnement "mitraillette". Il faut utiliser la version v042018 de "mitraillette" pour le cycle cy46.

Concernant les namelists:

- Elles sont sur BEAUFIX, sur le répertoire /home/gmap/mrpm/yessad/mitraille/namelist/cy46.
- Elles sont également archivées sous GIT, sous le répertoire validation/mitraille/namelist.
- Le modèle global utilise les namelists en "G[...].nam", les modèles à aire limitée les namelists en "L[...].nam".
- La mise à disposition d'un nouveau cycle ou d'un cycle intermédiaire de développement impose une remise à jour de ces namelists.
- Un changement de version opérationnelle d'ARPEGE, d'ALADIN ou d'AROME peut imposer une réactualisation de ces namelists: par exemple physique, passage sur une situation météorologique plus récente: en effet il faut chercher à valider avec mitraillette des configurations de modèles utilisées actuellement ou récemment en opérationnel et non des configurations datant d'il y a 10 ans.

4.2 Namelists d'ARPEGE et AROME oper et dble.

Elles sont utilisées par le script de la chaîne opérationnelle et elles sont susceptibles d'être dans les environnements OLIVE et VORTEX. Toutes les namelists doivent respecter les normes listées en partie 2.

Il y a quelques namelists à usage particulier qui ont leurs propres éléments de namelists.

- ARPEGE METROPOLE: namel_ana_surfex, namel_cplsurf_def, namel_ext_sst, namelistcris331, namelistiasi314, namel_previ_surfex.
- ARPEGE PEARP: namcombi.
- AROME-METROPOLE:
  namel_ana_surfex, namel_cplsurf_def, namelists en namel_lamflag_..., namelists en namel_previ_surfex..., namel_progrele, namelists en namel_rgb.
- AROME-REUNION, AROME-POLYF, AROME-NCAL, AROME-ANTIG:
  namel_ana_surfex, namel_cplsurf_def, namelistiasi314, namelists en namel_lamflag_..., namelists en namel_previ_surfex..., namel_pseudotraj, namelists en "namel_rgb".
- Les namelists en "select_" utilisées dans FULL-POS LALON doivent contenir les éléments suivants: NAMFPDY2, NAMFPDYF, NAMFPDYH, NAMFPDYI, NAMFPDYP, NAMFPDYS, NAMFPDYT, NAMFPDYV, NAMFPPHY.

5 Réactualisation de namelists.

∗ **Actions:** Il y a un certain nombre de questions à se poser lorsqu'on a à remettre à jour des namelists:

- 1/ Quels sont les éléments nouveaux?
- 2a/ Quels sont les éléments renommés?
- 2b/ Quels sont les éléments qui disparaissent et dont le contenu est transféré dans un élément existant?
- 3a/ Quels sont les variables qui changent de nom sans changer d'élément de namelist?
- 3b/ Quels sont les variables qui changent d'élément de namelist sans changer de nom?
- 3c/ Quels sont les variables qui changent à la fois de nom et d'élément de namelist?
- 4/ Quelles sont les variables qui disparaissent?
- 5a/ Quelles sont les variables nouvelles, qu'il faut spécifier en namelist car leurs défauts ne sont pas ceux qu'on veut utiliser?
- 5b/ Quelles sont les variables existantes, dont les défauts sont ne sont plus corrects, et qu'il faut donc rajouter en namelist?
- 5c/ Quelles sont les variables existantes, dont les défauts sont devenus corrects, et que donc on peut supprimer des namelists?
- 5d/ Quelles sont les variables référencées en namelist dont il faut changer la valeur?
- 6/ Quels sont les éléments qui disparaissent?

Il faut faire un recensement de tous ces points précis, qu'on peut traiter par la procédure `tnt.py` (voir ci-dessous).

Il faut remarquer que certains de ces points nécessitent d'avoir une certaine connaissance du code et de la signification des variables de namelist.

∗ **Outils:** Ces outils ont maintenant été rassemblés sous GIT (projet VALIDATION) sous le répertoire validation/mitraille. Quelques uns sont sous l'environnement VORTEX. Une description détaillée en est faite dans la documentation (IDMITS). On peut citer par exemple:

- Un fichier recensant l'historique des modifications de namelists d'un cycle à l'autre, depuis le cycle 36: fichier mitraille/doc/history_difnam.
- Un fichier contenant la liste de tous les élements de namelists, vidés de tout contenu: fichier mitraille/namelist/vide.
- Ces fichiers concernent des cycles de développement. Les cycles opérationnels ou de chaîne en double n'y sont pas référencés explicitement, mais le plus souvent on peut réutiliser l'environnement du cycle de développement associé (par exemple réutiliser l'environnement de cy46 pour cy46_op1).

On recommande fortement d'utiliser maintenant la procédure `tnt.py` (sous VORTEX) pour mettre à jour les namelists tant de MITRAILLETTE que de l'environnement opérationnel. Cette procédure met également les namelists aux normes listées en section 2.

On recommande également de controler les namelists par la procédure `nam_check_consistency.py` pour vérifier certains des points exposés en section 3.

6 Conclusion.

La gestion des namelists n'est pas un travail du tout négligeable, elle requiert non seulement une veille active des évolutions du code et aussi des environnements d'OLIVE, de VORTEX et oper/dble, mais également une bonne connaissance des points principaux du code et de la signification des principales variables de namelist.

Les outils récemment développés et archivés sous VORTEX ou sous GIT devraient sensiblement faciliter ce travail.

7 Références.

- (IDMITS) Yessad, K., 2018: MITRAILLETTE: environnement files and user's guide (version v042018 valid for cycle 46). Internal note, 8pp, available on "http://www.umr-cnrm.fr/gmapdoc/".