Code d'évolution stellaire CESAM2*k*

Aide mémoire: Installation, Utilisation

Pierre Morel
OCA Observatoire de Nice, BP. 4229, 06304 Nice Cedex 4
UMR CNRS 6529

Version 1.0.0.0, March 17, 2005

Qund un jour, tôt ou tard, il faut qu'on disparaisse,
Quand on a plus ou moins vécu, souffert, aimé
Il ne reste de soi que les enfants qu'on laisse
Et le champ de l'Effort où l'on aura semé.

Ch. De Gaulle

Contents

List of Figures

List of Tables

Index

Chapter 1

Installation

On n'exécute pas tout ce qui se propose

Et le chemin est long du projet à la chose.

Le Tartuffe.

1.1 CESAM2*k*

Le principe de l'exploitation de CESAM2*k* est différent de celui utilisé dans les versions précédentes de CESAM. L'exécutable, cesam2k.out, est créé une fois pour toutes. C'est dans le fichier de données, mon_modele.don, que la physique à utiliser est indiquée conjointement aux paramètres. Il n'y a donc plus à créer de module exécutable adapté à chaque utilisation comme tel était le cas avec les versions précédentes de CESAM. Il est toutefois possible de modifier la plupart des options retenues dans la source grâce à des fichiers de "réglage" décrits au § 3 (Page 47) qui permettent d'adapter CESAM aux moyens de calcul dont on dispose et, dans une large mesure, aux particularités des modèles à calculer.

CESAM2*k* est programmé en `FORTRAN95`, ce qui permet un dimensionnement dynamique des tableaux adapté à chaque cas particulier. Les compilateurs suivants ont été utilisés avec succés :

- sous LINUX : `ifc` de INTEL (gratuit), Lahey-Fujitsu `lf95`,
- sous UNIX : compilateur UNIX des machines ALPHA
- sous WINDOWS : Compact Visual Fortran

PB `Le compilateur gratuit ifc de INTEL n'a intégré les recommandations 1998 de la norme qu'après mars 2004. Des difficultés de compilations peuvent apparaître avec les versions antérieures,` *cf.* § B.1 (Page 69).

PB CESAM2*k* `utilise le logiciel de dessin PGPLOT dont le fichier de configuration n'est pas prévu pour le compilateur ifc. Thierry Corbard, Observatoire de la Côte d'Azur a construit les procédures` nécessaires que l'on trouvera § B.2 (Page 70) et dans le sous-directory SCRIPTS.

1.2 Contenu de la distribution

Le package CESAM2k[1] contient les sous-directory :

- EXPLOIT : ensemble de programmes et d'exemples de fichiers pour l'exploitation.
- NOTICE : sources LaTeX et fichiers postcripts de la notice d'utilisation[2] et du présent aide-mémoire.
- SCRIPTS : procédures UNIX/LINUX pour l'installation et l'exploitation.
- SOURCE : ensemble des routines, modules et programme de la source.
- SUN_STAR_DATA : données physiques, tables d'EOS par exemple.
- TESTS : programmes de test[3].

1.2.1 Avertissement

La notice générale décrivant les algorithmes de CESAM2*k* est indisponible car en cours d'élaboration. Des informations, hélas incomplètes, sur la physique et les méthodes numériques utilisées peuvent être trouvées dans la notice de CESAM4.

1.3 Installation

On décrit l'installation sous UNIX/LINUX en utilisant le C-shell.

PB Avec un compilateur dont la version n'est pas à jour de la norme F95, (comme certaines versions du compilateur d'INTEL) il y aura refus d'effectuer une allocation dans une routine autre que celle où est déclarée l'allocation. Il y a lieu de remplacer dans les routines concernées, les instructions ALLOCATABLE par POINTER pour les tableaux incriminés *cf.* § B.1 (Page 69).

1. Dézipper et détarer la distribution :

   ```
   tar -xzvf CESAM2k.tar.gz
   ```

   Le sous-directory CESAM2k est créé. Il contient 6 sous-directory tarés et compressés :

[1] 2k pour "2000".
[2] En cours d'élaboration.
[3] Certains de ces programmes sont à adapter à la présente version de CESAM2*k*.

```
SOURCE.tar.gz
EXPLOIT.tar.gz
SUN_STAR_DATA.tar.gz
SCRIPTS.tar.gz
TESTS.tar.gz
NOTICE.tar.gz
```

qu'il convient de décompresser et détarer au fur et à mesure des besoins. Cette disposition permet de placer ces sous-directory aux endroits les mieux adaptés et ainsi facilite les mises à jour qui, la plupart du temps, n'affectent que le seul sous-directory `SOURCE`.

2. Dans le sous-directory `SCRIPT`, adapter au compilateur F95 utilisé les paramètres des procédures de compilation `compile2k` et d'exécution `exe2k`; y indiquer aussi les chemins de la source et des bibliothèques. Rendre les procédures exécutables : `chmod 700 *`.

3. Pour une utilisation avec le shell csh ou tcsh, dans le fichier .cshrc ajouter les chemins de la bibliothèque et des exécutables, par exemple :

   `set mypath=(.... ~/CESAM2k/SOURCE ~/CESAM2k/EXPLOIT .....)`

   `set path=(. ~ ~/SCRIPTS $path $mypath) .....)`. Adapter ces recommandations au fichier .bashrc pour une utilisation avec le shell bash.

4. Si ce n'est déjà fait, installer le logiciel de dessin `PGPLOT`[4], et indiquer le chemin de sa bibliothèque dans le fichier .cshrc :

   `setenv PGPLOT_DIR /usr/local/pgplot`. On donne au § B.2 (Page 70) un supplément d'information concernant l'installation de ce logiciel.

   A défaut, de ce logiciel de dessin, dans le sous-directory `SOURCE` :

   (a) Supprimer boite.f, box.f, des_r.f, des_m.f de la liste des `INCLUDE` du module mod_cesam.f.

   (b) Dans la subroutine des.f, ne conserver que `CASE ('no_des')` et, éventuellement, `CASE DEFAULT`.

   (c) Dans la subroutine sortie.f, supprimer `CALL pgend`.

PB `La liste de ces corrections n'est peut être pas complète.`

1.4 Adaptation de la source

Avant de créer le module exécutable, il convient d'adapter certains paramètres libres de la source à l'installation dont on dispose.

[4]Avec son mode d'emploi, le logiciel `PGPLOT` est téléchargeable via `anonymous ftp` à l'adresse `http://astro.caltech.edu/~tpj/pgplot/`.

- Dans la routine des_m, ligne 105, adapter le nom du "device" : `device='/xw'`, éventuellement coder `device='?'` de façon à obtenir la liste des devices disponibles.
- Faire de même dans la routine des_r, ligne 89.

Dans les routines des_m et des_r il conviendra, par la suite, d'adapter la hauteur et la largeur des cadres de façon à optimiser leur centrage dans la fenêtre dont on dispose ou encore de personnaliser le dessin *cf.* § 3.9 (Page 56).

1.5 Création du module exécutable

Se placer dans le sous-directory SOURCE, y lancer la procédure[5] :

genere_cesam2k

Après la compilation on obtient la liste des "object module" de la bibliothèque libcesam2k.a et le menu d'entrée de CESAM :

```
 mod_kind.o
 mod_numerique.o
 mod_donnees.o
 mod_variables.o
 mod_etat.o
 mod_opa.o
 mod_conv.o
 mod_atm.o
 mod_nuc.o
 mod_bp_for_alecian.o
 mod_evol.o
 mod_static.o
 mod_cesam.o
 mod_exploit.o
 Encountered 0 errors, 0 warnings in file cesam2k-dbg.f.

CESAM speaks a bit of english if you include in
the working directory a file named langue with the statements :
&NL_LANGUE
langue='english'
/
cf. aide_mem2k, chapter Personnalisation

Pour arrêter : taper 0 puis RETURN
Pour poursuivre une évolution : taper 1 puis RETURN
Pour initialiser un modèle de ZAMS : taper 2 puis RETURN
Pour initialiser un modèle de PMS : taper 3 puis RETURN
 ARRET
```

[5]Dans la suite le symbole <== indique une instruction à saisir.

Suivant l'installation et les paramètres de compilation retenus, la création de la bibliothèque et du module exécutable nécessitent de l'ordre de 3 à 10 minutes.
De façon similaire, on peut créer un exécutable de debug avec la procédure `genere_cesam2k-dbg` du sous-directory `SCRIPTS`.

1.5.1 Fonctionalités des modules

Les modules de CESAM2k ont les fonctionalités suivantes :

- Module mod_kind : regroupe les types des variables.
- Module mod_numerique : regroupe les routines purement numériques.
- Module mod_donnees : regroupe l'ensemble des quantités fixes au cours de l'évolution.
- Module mod_variables : regroupe l'ensemble des quantités variables au cours de l'évolution.
- Module mod_etat : regroupe les routines concernant l'équation d'état.
- Module mod_opa : regroupe les routines concernant le calcul de l'opacité.
- Module mod_conv : regroupe les routines concernant la convection.
- Module mod_atm : regroupe les routines de restitution de l'atmosphère.
- Module mod_nuc : regroupe les routines concernant les réactions thermonucléaires.
- Module mod_bp_for_alecian. : regroupe les routines concernant le calcul des accélérations radiatives, suivant le formalisme de G.Alécian.
- Module mod_evol : regroupe les routines concernant l'évolution temporelle de la composition chimique.
- Module mod_static : regroupe les routines concernées par la résolution de l'équilbre quasi-statique.
- Module mod_cesam : regroupe les routines concernant la gestion du calcul.
- Module mod_exploit : regroupe les routines concernées par l'exploitation des résultats.

On trouvera § 1.1 (Page 28) un organigramme général du code. Chaque routine `PRIVATE` et/ou `PUBLIC` est introduite sous la forme d'`INCLUDE` dans son module d'appartenance, ce dernier étant indiqué dans les commentaires du début de la routine.

1.6 Test de fonctionnement

Ouvrir une nouvelle fenètre et se placer dans le directory `TESTS`. Editer le fichier mon_modele.don.

- Remplacer "`des_m`" par "`no_des`" si `PGPLOT` n'est pas opérationnel.
- Adapter "`nom_chemin`" au chemin du sous-directory `SUN_STAR_DATA` de votre installation.
- Rétablir éventuellement les instructions suivantes:

```
&NL_CESAM
NOM_CHEMIN='~/SUN_STAR_DATA/',    <==== à adapter
NOM_CTES='ctes_94',
NOM_DES='des_m',   <==== à adapter
NOM_OUTPUT='no_output',
N_MAX=2000,
PRECISION='np'
/
&NL_MASS
MTOT=1.d0,
NOM_PERTM='pertm_ext',
MDOT=0.d0
/
&NL_EVOL
AGEMAX=1.d2,
ARRET='ELSE',
DTLIST=1.d10,
LOG_TEFF=10.d0,
NB_MAX_MODELES=2000,
T_STOP=5.d7,
X_STOP=-0.1d0
/
&NL_CHIM
GRILLE_FIXE=f,
NOM_ABON='solaire_gn',
MODIF_CHIM=.FALSE.,
GARDE_XISH=.FALSE.,
X0=0.70d0,
Y0=0.28d0,
ZSX0=0.d0
/
&NL_CONV
NOM_CONV='conv_jmj',
ALPHA=1.8d0,
OVSHTS=0.d0,
OVSHTI=0.d0,
```

```
 JPZ=F,
 CPTURB=0.d0,
 LEDOUX=F
 /
 &NL_DIFF
 DIFFUSION=F,
 NOM_DIFFM='diffm_mp',
 NOM_DIFFT='difft_nu',
 D_TURB=0.d0,
 RE_NU=1.d0,
 NOM_FRAD='no_frad'
 /
 &NL_ROT
 ROT_SOLID=T,
 W_ROT=0.d0,
 UNIT='jours',
 W_FORM=0.d0,
 NOM_DIFFW='diffw_mpz',
 NOM_PERTW='pertw_0',
 P_PERTW=0.d0,
 PW_EXTEND=0.d0
 /
 &NL_ETAT
 NOM_ETAT='etat_eff',
 F_EOS='eos_opal_250.bin',7*' '
 /
 &NL_OPA
 NOM_OPA='opa_gong',
 F_OPA='opa_yveline.bin',7*' '
 /
 &NL_NUC
 NOM_NUC='ppcno9',
 NOM_NUC_CPL='NACRE',
 MITLER=F
 /
 &NL_ATM
 NOM_ATM='lim_atm',
 NOM_TDETAU='hopf',
 TAU_MAX=10.d0,
 LIM_RO=T
 /
```

Lancer cesam2k.out

```
cesam2k.out <==

-------------------------
CESAM speaks a bit of english if you include in
```

```
the working directory a file named langue with the statements :
&NL_LANGUE
langue='english'
/
cf. aide_mem2k, chapter Personnalisation
---------------------------

Pour arrêter : taper 0 puis RETURN
Pour poursuivre une évolution : taper 1 puis RETURN
Pour initialiser un modèle de ZAMS : taper 2 puis RETURN
Pour initialiser un modèle de PMS : taper 3 puis RETURN
2  <==
 le modèle initial de ZAMS est-il donné en binaire ? o/n
n <==
 entrer le nom du fichier ASCII du modèle initial
 Exemples: m010.zams, m020.zams, m050.zams

m010.zams <==

 CESAM utilise le modèle m010.zams
 entrer l'identificateur du modèle
 Exemple: mon_modele, modele_euler

mon_modele <==

 identificateur des fichiers du modèle : mon_modele

 ***************************************************************

 MODELE DE STRUCTURE INTERNE calculé par CESAM2k version 1.0.0.0

 ***************************************************************

 .....                    .....                    .....
   .....                    .....                    .....
      .....                    .....                    .....

------- Restitution de l'atmosphère (fin) ------

*********
âge= 1.000E+02, LogTeff= 3.777E+00, LogL/Lsol=-7.797E-02, LogR/Rsol=-6.864E-02
Log g= 4.575E+00, Pc= 1.711E+17, Tc= 1.358E+07, Roc= 9.298E+01, Xc= 6.919E-01
en. PP= 98%, en. CNO=  2%, en. 3 alpha=  0%, en. grav=  0%, Yc= 2.875E-01
Var. rel. de masse :  0.000E+00, M*= 1.000E+00Msol, modèle de la série principale
*********

Fin d'évolution avec CESAM2k version : 1.0.0.0
```

```
Sortie car agemax atteint

 .....                         .....                         .....
   .....                         .....                         .....
      .....                         .....                         .....

***********************************

 Type <RETURN> for next page:
```

Le test de fonctionnement est terminé.

Lorsque le dessin "on line" est effectué, suivant la définition de l'écran utilisé, il arrive que les cadres soient tronqués et/ou n'utilisent pas la surface disponible de façon satisfaisante. Il y a lieu ou bien d'adapter les dimensions des cadres dans les routines des_m.f et des_r.f du sous directory `SOURCE`, *cf.* § 1.4 (Page 15), ou bien de personnaliser le dessin suivant les instruction du § 3.9 (Page 56)

1.7 Mise en forme des données physiques

CESAM2k utilise les données physiques tabulées *i.e.* EOS, opacités *etc...* sous forme de fichiers binaires qu'il faut créer à partir des données `ASCII` du sous-directory `SUN_STAR_DATA` dans lequel on se placera.

Les fichiers binaires de données créés pour une version précédente de CESAM sont utilisables avec CESAM2*k*.

1.7.1 Opacités "Yveline"

Décompresser le fichier opa_yveline.data.gz et le programme : ascii2bin_opa.f.gz

```
gunzip opa_yveline.data.gz ascii2bin_opa.f.gz
```

Effectuer la transformation ASCII ==> binaire en exécutant[6] le programme ascii2bin_opa :

```
exe2k ascii2bin_opa
```

A la question :

```
 traduction ASCII ---> binaire (o/n?)
 o <==
```

on obtient :

```
 ASCII ---> binaire
 enter the name of the existing ASCII opacity table: opa_yveline.data
```

répondre :

```
 opa_yveline.data <==
```

on obtient :

[6] Avec F95 il est pratique d'utiliser les procédures du sous-directory `SCRIPTS`.

```
enter the name of the binary table: opa_yveline.bin
```

répondre :

```
opa_yveline.bin <==
```

à la question suivante répondre c, éventuellement recompresser le fichier en ASCII et supprimer l'exécutable. On obtient :

```
PAUSE: OK c'est fait
pour arreter entrer q, poursuivre entrer c
c <==
```

Recompresser les données ASCII : `gzip opa_yveline.data <==` et effacer l'exécutable : `rm ascii2bin_opa.out <==`.
On trouve un complément d'information dans le fichier extract_opa_yveline_explik.

1.7.2 Opacités HOUDEK

Package de calcul de l'opacité OPAL, Iglesias & Rogers (1991), par les routines d'interpolation par splines birationnelles de Houdek & Rogl (1996) il est appelé par la routines externe opa_houdek9. Ce package de nom opint_v9f.tar.gz peut être obtenu "via anonymous ftp from ftp.ast.cam.ac.uk in the sub-dir PUB/HG/OPACITY".

Il est constitué du fichier opint_v9f.tar.gz du sous-directory `SUN_STAR_DATA` qu'il faut décompresser et détarer :

```
gunzip opint_v9f.tar.gz <==
tar -xvf opint_v9f.tar <==
v9/
v9/.make.state
v9/.nse_depinfo
v9/.sbinit
v9/Makefile
v9/OPINTPATH_92
v9/OPINTPATH_95
v9/README
v9/Release_history
v9/a2b95.f

............................

v9/rksuite/Makefile
v9/rksuite/chkfl.f
v9/rksuite/.make.state
v9/alex94bext.f
v9/dopdalex94.f
v9/OPINTPATH_AX
v9/exakop95.f
```

Il est prudent de recompresser le fichier opint_v9f.tar. En suivant les instructions du README, éditer le makefile du directory `v9` et adapter le compilateur et l'optimiseur à la machine UNIX dont on dispose.
Exécuter le MAKEFILE : `make`.
Dans le directory `v9/OPINPATH_AX`, remplacer les "./" par les chemins `./v9/opal95/opal95e.bin` donne, par exemple :

`/home/bilou/SUN_STAR_DATA/v9/opal95/opal95e.bin`

faire de même pour `v9/OPINPATH_91` et `v9/OPINPATH_95`.

PB Dans la routine d'appel opa_houdek9 le chemin d'accès aux données : `tabnam=TRIM(nom_chemin)//TRIM(f_opa(2))` fait appel au **second** fichier `f_opa(2)` d'opacité, ainsi qu'il est codé dans la NAMELIST NL_OPA du test du § 2.3 (Page 32).
Définir les paramètres `iorder` et `imode` en se référant à la notice constituée par le fichier `./doc/notes.ps` du directory `v9`. Dans le script `exe2k` indiquer au LINK le chemin d'accès à la bibliothèque libopint.a.

1.7.3 Equation d'état OPAL

1.7.3.1 OPAL 1991

Package de l'équation d'état OPAL, Iglesias & Rogers (1991), appelée par la routine etat_opal. Le package original et les tables d'équation d'état ont été repris du site WEB `ftp://www-phys.llnl.gov/pub/opal/eos/` où l'on trouve un "README" expliquant la marche à suivre pour construire une table ASCII d'équation d'état pour une valeur de Z fixée. On transforme ensuite cette table ASCII en fichier binaire.

PB Certains des sous-programmes du package original d'OPAL ont été modifiés pour l'exploitation avec CESAM2k; en particulier, le blockdata a été supprimé.

Les fichiers ASCII compressés peos*.gz contiennent les données pour l'équation d'état OPAL. Se placer dans le sous-directory `SUN_STAR_DATA`, décompresser ces fichiers : `gunzip peos*`.

Exécuter le programme ZFSinterppeos après en avoir décompressé la source : `ZFSinterppeos.f.gz`

```
exe2k ZFSinterppeos <==
Encountered 0 errors, 0 warnings in file ZFSinterppeos.f.
type Z:
0.02 <==
```

entrer la valeur de Z, fraction de masse de l'abondance des métaux; on obtient :

```
        0.020
  0.0000   0.0000          3.0000000                4.0036974
  0.0000   0.0200          3.0301077                4.0629535
  0.0000   0.0400          3.0611377                4.1251554
  0.2000   0.0000          2.5018048                2.5114787
```

```
   0.2000   0.0200            2.5158978                    2.5350964
   0.2000   0.0400            2.5302570                    2.5591605
   0.4000   0.0000            2.2741616                    1.8296305
   0.4000   0.0200            2.2828465                    1.8421309
   0.4000   0.0400            2.2916493                    1.8548033
   0.6000   0.0000            2.1437435                    1.4389958
   0.6000   0.0200            2.1498623                    1.4467140
   0.6000   0.0400            2.1560483                    1.4545176
   0.8000   0.0000            2.0592246                    1.1858406
   0.8000   0.0200            2.0638888                    1.1910768
   0.8000   0.0400            2.0685947                    1.1963595
```

Le fichier ASCII EOSdata occupant environ 9Mo a été créé.
Editer la source du programme opal_ascii_bin et donner au fichier binaire le nom désiré, par exemple :

```
......................................

     CLOSE(unit=60)   <===   ligne 117 de opal_ascii_bin.f

c    f_eos='eos_opal_190.bin'
     f_eos='eos_opal_195.bin'
c    f_eos='eos_opal_330.bin'
c    f_eos='eos_opal_180.bin'
c    f_eos='eos_opal_170.bin'
c    f_eos='eos_opal_130.bin'
c    f_eos='eos_opal_050.bin'

     WRITE(*,6)f_eos
6    FORMAT('debut ecriture sur le fichier binaire: ',a50)
     CLOSE(unit=60)

............................................
```

Exécuter le programme opal_ascii_bin ainsi modifié :

```
exe2k opal_ascii_bin <==
Encountered 0 errors, 0 warnings in file opal_ASCII_bin.f.
donnees prises dans le fichier: EOSdata
lecture, et c'est long, de ce fichier
 fin de lecture des tables OPAL
 debut ecriture sur le fichier binaire: eos_opal_195.bin
 fin ecriture sur le fichier binaire
 donnees prises dans le fichier binaire: eos_opal_195.bin
 lecture, des donnees EOS opal, et c'est long
 fin de lecture des donnees EOS opal en binaire
 test relecture effectue
```

Supprimer les modules exécutables :

```
rm *.out <==
rm: remove regular file ‘ZFSinterppeos.out’? y <==
rm: remove regular file ‘opal_ascii_bin.out’? y <==
```

Enfin recompresser les fichiers ASCII, et supprimer les fichiers EOSdata et fort.2 désormais inutiles :

```
gzip peos* <==
rm EOSdata <==
rm: remove regular file ‘EOSdata’? y <==
rm fort.2 <==
rm: remove regular file ‘fort.2’? y <==
```

1.7.3.2 OPAL 2001

Les équation d'état 2001 offrent une alternative pour la métallicité : ou bien, comme dans la version 1991, on crée une table pour une métallicité donnée, et l'abondance d'hydrogène doit être dans l'intervalle $X \in [0, 0.8]$, ou bien $X \in [0, 1]$ mais $Z = 0$, *cf.* § 2.7 (Page 37).

1. Avec $Z \neq 0$ Se placer dans le sous-directory SUN_STAR_DATA, y créer le fichier d'équation d'état correspondant à la métallicité requise en exécutant la programme Z_interp_IEOS :

   ```
   bilou@port-morel% exe2k Z_interp_IEOS
   Encountered 0 errors, 0 warnings in file Z_interp_IEOS.f.
   type Z:  0.025  <===== entrer la valeur de Z
      0.025000000
   bilou@port-morel%
   ```

   Il y a création d'un fichier ACSII compressé de nom EOSdata.gz. Transformer ce fichier en binaire en lui donnant le nom avec lequel il sera identifié par la suite. Pour ce faire, entrer ce nom dans le programme opalZ_ascii_bin (vers la ligne 135) :

   ```
   c f_eos='eos_opal_190.bin'
   c f_eos='eos_opal_195.bin'
   f_eos='eos_opal_250.bin'  <========
   c f_eos='eos_opal_330.bin'
   c f_eos='eos_opal_180.bin'
   ```

   puis exécuter ce programme qui comporte un test de lecture :

   ```
   bilou@port-morel% exe2k opalZ_ascii_bin
   Encountered 0 errors, 0 warnings in file opalZ_ascii_bin.f.
   Fichier de données inconnu : EOSdata
   décompression du fichier ASCII de données : EOSdata.gz
   ```

```
données ASCII prises dans le fichier: EOSdata
lecture, et c'est long, de ce fichier
 fin de lecture des tables OPAL
début des écritures sur le fichier binaire: eos_opal_250.bin
fin des écritures en binaire, test de relecture
 données prises dans le fichier binaire: eos_opal_250.bin
 décompression du fichier
 lecture, des donnees EOS opal, et c'est long
recompression du fichier binaire eos_opal_250.bin
 fin de lecture des données EOS opal en binaire
 test de relecture réussi
bilou@port-morel%
```

Eventuellement supprimer EOSdata.gz :

```
rm EOSdata.gz
```

2. Avec $Z = 0$, l'interpolation utilise le fichier EOSdata_H-He qu'il suffit de transformer en binaire en exécutant le programme opalX_ascii_bin du sous-directory SUN_STAR_DATA.

1.7.4 Equation d'état MHD

Package de l'équation d'état MHD, Mihalas et al. (1988), appelé par la routine externe etat_mhd. Ce package a été fourni par W. Däppen. Les 8 tables mhd1_oc_tau.tab ... mhd8_oc_tau.tab qui se trouvent sur le serveur anonyme usc.edu, dans le répertoire pub/astro-physics/mhd-oc-tau/wd-evo sont disposées dans le sous-directory SUN_STAR_DATA sous les noms mhd1.tab.gz ... mhd8.tab.gz. Se placer dans le sous-directory SUN_STAR_DATA, décompresser ces tables en ASCII : gzip mhd* <== et la source du programme fmttob.f : gzip fmttob.f.gz. Transformer les tables en binaires en exécutant fmttob.f <==

```
exe2k fmttob
Encountered 0 errors, 0 warnings in file fmttob.f.
 decompression

Press Enter to Continue. <==

 enter main
 ZAMS-type: filename of input (formatted) table #  1
 enter filename of output (unformatted) table #  1
 ZAMS-type: filename of input (formatted) table #  2
 enter filename of output (unformatted) table #  2
 ZAMS-type: filename of input (formatted) table #  3
 enter filename of output (unformatted) table #  3
 center-type: filename of input (formatted) table #  4

 ..................................
```

```
    4.002600       7.7222100E-02   0.2299999

    12.01100       4.2665000E-04   3.8132454E-03

    14.00670       1.0717000E-04   1.1169993E-03

    15.99940       9.1215000E-04   1.0859599E-02

    20.17900       2.8039000E-04   4.2102317E-03

 mean molecular weight =    1.2455340


bilou@port-morel%
```

Supprimer les exécutables :

```
rm *.out <==
rm: remove regular file fmttob.out ?  y <==
```

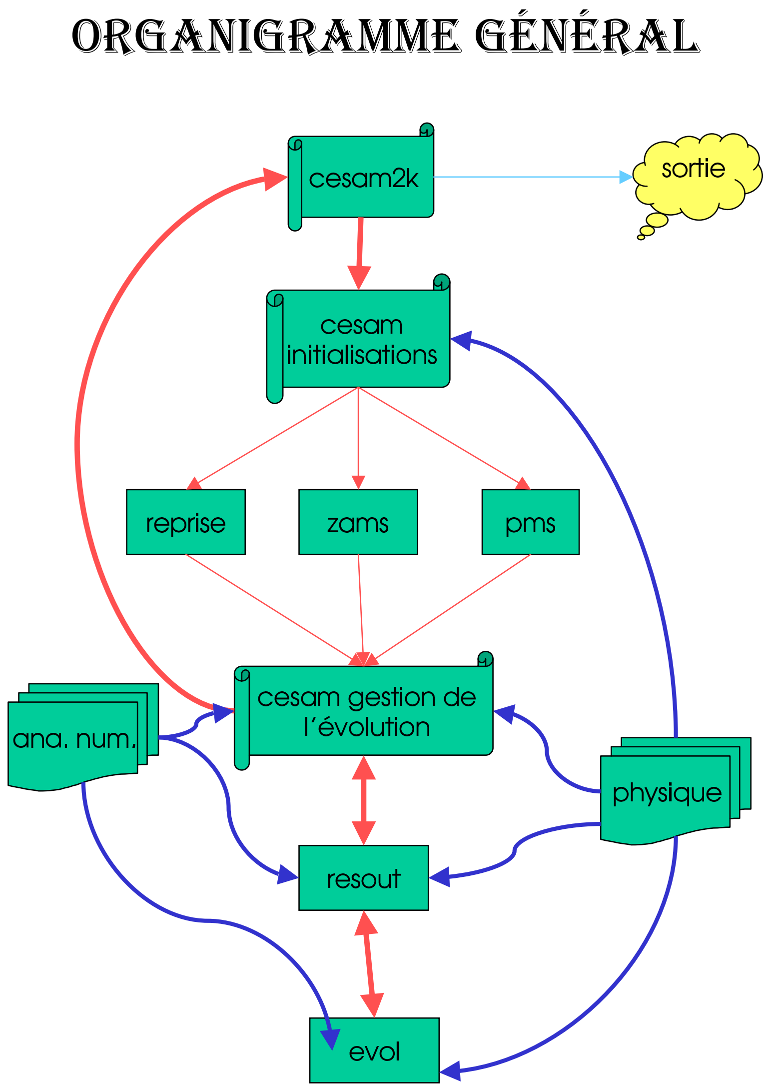

Figure 1.1: Schéma de l'organigramme général de CESAM2k. Le programme cesam2k, compilé une fois pour toutes, fait appel à la routine cesam qui constitue en fait, le programme principal. Dans cesam après des initialisations on sépare le traitement des 3 possibilités : poursuite d'une évolution, initialisation sur la ZAMS homogène, initialisation d'une PMS. Le calcul se poursuit par des opérations de gestion de l'évolution, listing, création de fichiers de sortie, dessin, etc... Est ensuite fait appel à resout où les équations de l'équilibre quasi-statique sont résolues en alternance avec celles de l'évolution du moment cinétique et de la composition chimique. Ces dernières sont résolues séparément dans evol. Une fois la convergence obtenue, il y a retour vers les algorithmes de gestion de l'évolution de cesam et, de là, poursuite de l'évolution ou retour à cesam2k pour sortie. Les routines d'analyse numérique et de physique sont sollicitées à différents niveaux des calculs.

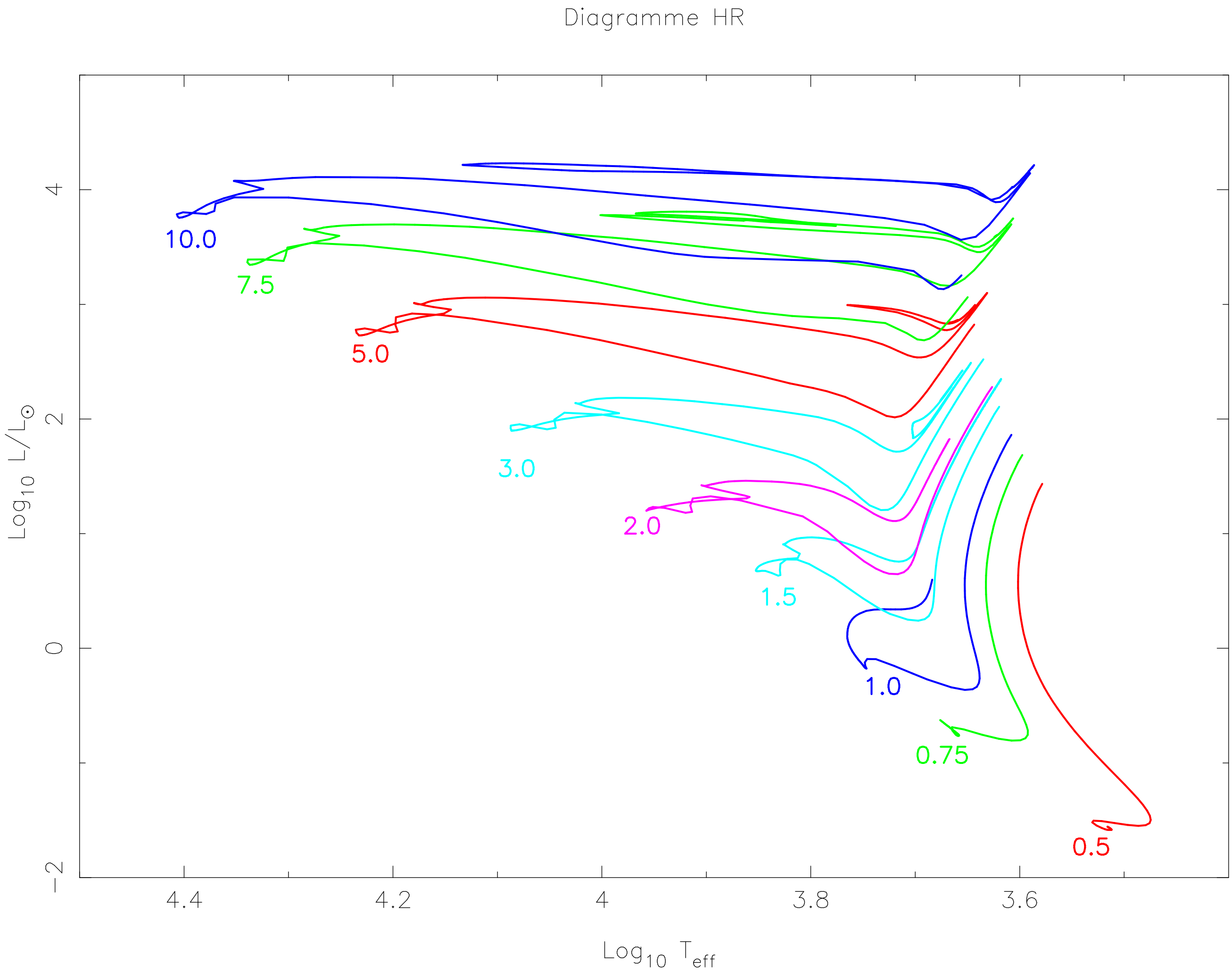

Figure 1.2: Tracés évolutifs incluants la pré-séquence principale pour des étoiles de $0.5M_{\odot}$ à $10M_{\odot}$. Ces tracés ont été obtenus avec le fichier de données du § 2.3 (Page 32).

Chapter 2

Exploitation

Va despacio, estoy de prisa.

Proverbe Mexicain.

2.1 Conseils

Un code numérique aussi bien fait qu'il soit ne peut être clefs en mains. Pour entamer une exploitation on doit toujours être guidé par la remarque triviale :

Un programme qui ne fonctionne pas ou mal pour des cas simples
ne fonctionnera pas ou encore plus mal pour des cas compliqués.

Un code de structure interne n'échappe hélas pas à cette règle. A cet effet, bien que CESAM ait prévu différents types de "précision" standard il est presque toujours nécessaire, à l'aide de "réglages", d'optimiser les paramètres numériques en fonction des conditions rencontrées au cours du calcul. Aborder un problème en introduisant *ad initio* toute la complexité que l'on désire atteindre est une perte de temps car il faudra très rapidement reprendre des cas simples afin d'acquérir une expérience sur la façon dont l'évolution se déroule. Il est conseillé de "dégrossir" le problème en utilisant des options numériques et physiques robustes et efficaces afin d'apprécier rapidement "ce qui se passe", puis d'introduire progressivement la complexité souhaitée.

2.2 Le fichier de données : mon_modele.don

Les options physiques à utiliser et les données sont transmises au programme principal par des `NAMELIST` regroupées dans un fichier en ASCII dont le nom a **nécessairement** l'extension `.don`, par exemple : mon_modele.don. On a disposé un exemple de ce fichier dans le sous-directory `EXPLOIT`.

- `& NL_CESAM` : regroupe les conditions générales du calcul;
- `& NL_MASS` : regroupe les paramètres physiques relatifs à la variable spatiale;

- `& NL_EVOL` : regroupe les paramètres physiques relatifs à la variable temporelle;
- `& NL_CHIM` : regroupe les paramètres physiques relatifs à la composition chimique;
- `& NL_CONV` : regroupe les paramètres physiques relatifs à la convection;
- `& NL_DIFF` : regroupe les paramètres physiques relatifs à la diffusion;
- `& NL_ROT` : regroupe les paramètres physiques relatifs à la rotation;
- `& NL_ETAT` : regroupe les noms des fichiers, *d'accès externe*; éventuellement nécessaires au calcul de l'équation d'état ;
- `& NL_OPA` : regroupe les noms des fichiers, *d'accès externe*, éventuellement nécessaires au calcul de l'opacité;
- `& NL_NUC` : définit des options relatives au calcul des taux de réactions thermonucléaires;
- `& NL_ATM` : regroupe les paramètres physiques relatifs à la restitution de l'atmosphère.

2.3 Exemple de fichier de données

Cet exemple est volontairement différent de celui utilisé au § 1.6 (Page 18).

```
&NL_CESAM
NOM_CHEMIN='~/SUN_STAR_DATA/',
NOM_CTES='ctes_94',
NOM_DES='des_r',
NOM_OUTPUT='no_output',
N_MAX=2000,
PRECISION='pr'
/
&NL_MASS
MTOT=2.d0,
NOM_PERTM='pertm_ext',
MDOT=0.d0
/
&NL_EVOL
AGEMAX=5.d3,
ARRET='ELSE',
DTLIST=1.d10,
LOG_TEFF=10.d0,
NB_MAX_MODELES=2000,
T_STOP=5.d7,
X_STOP=0.1d0
/
```

```
&NL_CHIM
GRILLE_FIXE=.FALSE.,
NOM_ABON='solaire_gn',
MODIF_CHIM=.FALSE.,
GARDE_XISH=.FALSE.,
X0=0.70,
Y0=0.28,
ZSX0=0.0d0
/
&NL_CONV
NOM_CONV='conv_jmj',
ALPHA=1.8d0,
OVSHTS=0.d0,
OVSHTI=0.d0,
JPZ=F,
CPTURB=0.d0,
LEDOUX=F
/
&NL_DIFF
DIFFUSION=.TRUE.,
NOM_DIFFM='diffm_mp',
NOM_DIFFT='difft_nu',
D_TURB=10.d0,
RE_NU=1.d0,
NOM_FRAD='no_frad'
/
&NL_ROT
ROT_SOLID=.TRUE.,
W_ROT=0.d0,
UNIT='kms/s',
W_FORM=0.d0,
NOM_DIFFW='diffw_mpz',
NOM_PERTW='pertw_0',
P_PERTW=0.d0,
PW_EXTEND=0.d0
/
&NL_ETAT
NOM_ETAT='etat_eff',
F_EOS='eos_opal_250.bin',7*' '
/
&NL_OPA
NOM_OPA='opa_yveline',
F_OPA='opa_yveline.bin',7*' '
/
&NL_NUC
NOM_NUC='ppcno9',
NOM_NUC_CPL='NACRE',
```

```
MITLER=F
/
&NL_ATM
NOM_ATM='lim_atm',
NOM_TDETAU='hopf',
TAU_MAX=5.d0,
LIM_RO=.TRUE.
/
```

PB `Les noms de routines sont ''case sensitive'', par exemple il faut coder NACRE et non nacre.`

2.4 Arguments de la NAMELIST `NL_CESAM`

- `NOM_CHEMIN='~/SUN_STAR_DATA/'` : chemin et nom du répertoire à partir duquel où sont disposées les données physiques, les tables d'opacité par exemple.

- `NOM_CTES` : indique le nom de la routine d'initialisation des principales constantes de physique à utiliser :

  - `ctes_85` : constantes de physique proposées par GONG,
  - `ctes_94` : constantes de physique récentes.

- `NOM_DES` : nom de la routine de dessin à utiliser :

  - `des_m` : dessin en fonction de la masse, utilise PGPLOT,
  - `des_r` : dessin en fonction du rayon, utilise PGPLOT,
  - `no_des` : pas de dessin, à utiliser ,en particulier, si on ne dispose pas du logiciel PGPLOT, voir § 1.3 (Page 14),
  - `zoom` : forme personnalisée du dessin décrite au § 3.8 (Page 54).

- `NOM_OUTPUT` : type de fichier ASCII à générer à l'issue du calcul :

  - `osc_adia` : génération du fichier ASCII mon_modele-ad.osc pour le calcul des ocillations adiabatiques et les dessins,
  - `osc_invers` : génération du fichier ASCII mon_modele-inv.osc pour les inversions,
  - `osc_nadia` : génération du fichier ASCII mon_modele-nad.osc pour le calcul des ocillations non adiabatiques,
  - `no_output` : il n'y a pas de génération de fichier ASCII,
  - `ascii` : forme personnalisée du fichier ASCII mon_modele-ascii décrite au § 3.6 (Page 51).

- N_MAX=1000 : le nombre maximum de couches[1] est défini par MAX(N_MAX,n_max); les deux variables internes n_max et le nombre minimal de couches n_min sont définies dans la routine cesam suivant le niveau de précision requis.

- PRECISION='sp' : niveau de précision requis, on trouve dans la Table 2.1 (Page 36) les valeurs des paramètres numériques utilisées suivant les différentes options retenues ainsi qu'ils sont définis dans la routine cesam. Les constantes de répartition sont fixées indépendamment du type de précision. Les valeurs sont celles retenues dans CESAM4 à savoir : ctel=0, ctep=-1, ctem=15, cter=0, ctet=0. Pour utiliser des paramètres différents *cf.* § 3.7 (Page 52).

2.5 Arguments de la NAMELIST NL_MASS

- MTOT=-1.d-14 : masse totale initiale, en $M_\odot$,

- NOM_PERTM : nom de la routine de perte de masse :

  - perte_ext : perte/gain de masse externe, linéaire en fonction du temps;
  - perte_msol : perte/gain de masse externe linéaire en fonction du temps, la masse de l'étoile restant au moins égale à $1\,M_\odot$,
  - perte_tot : perte/gain de masse externe, linéaire en fonction du temps et tenant compte de la perte de masse due aux réactions thermonucléaires ($E = mc^2$),
  - pertm_waldron : perte de masse empirique de Waldron cité dans A.Ap 229 (1990) 469-474.

- MDOT : taux de perte/gain de masse, en $M_\odot$/an, ce taux est positif pour un gain de masse, négatif pour une perte de masse. Dans le cas standard, la composition chimique de l'apport/retrait de masse est celle de la couche la plus externe du modèle. Voir § 3.5 (Page 50) comment personnaliser, si besoin, cette composition chimique.

2.6 Arguments de la NAMELIST NL_EVOL

- AGEMAX=4.5d4 : âge maximal à atteindre en million d'années,

- ARRET='POST' : arrêt sur la ZAMS ('zams'), à la fin de la ZAMS ('post'), au début de la combustion de l'hélium ('cohe') ou du carbone ('coca') ou autre ('else').

- DTLIST=1.d3 : intervalle de temps minimum, en million d'années, séparant deux listes détaillées du modèle (fichier mon_modele.lis),

[1] La nature des paramètres numériques et logiques est identifiable par les valeurs données à titre indicatif.

Table 2.1: Valeurs des paramètres numériques suivant les diverses options de précision prévues. Les valeurs de ces paramètres peuvent être adaptées au type d'évolution en utilisant l'option `'rg'`, *cf.* § 3.7 (Page 52) on utilise alors les paramètres indiqués dans le fichier de nom `reglages` placé dans l'environnement. Les dénominations s'interprètent de la façon suivante : `sp` : super précision, `pr` : précision réaliste, `lm` : pour petites masses (low mass), `sa` : pour modèles avec précision solaire (solar accuracy), `np` : pour modèles de précision normale. La signification des différents paramètres numériques est explicitée au § 3.7 (Page 52) et dans la routine `cesam` du sous-directory `SOURCE`, lignes 741-766, d'autres options sont prévues, certaines utilisent les abscisses eulériennes; elles ne sont pas détaillées ici.

	sp	pr	lm	sa	np
m_qs	2	2	2	2	1
m_ch	2/3*	2/3*	2/3*	2/3*	2
psi0	0.05	0.05	0.08	0.05	0.08
m_rot	3	3	3	3	2
m_rl	4	4	4	4	6
m_tds	2	2	2	2	2
m_ptm	2	2	2	2	2
ordre	2	2	2	2	1
precix	10^{-4}	10^{-3}	10^{-4}	10^{-5}	0.005
precit	0.02	0.15	0.08	0.02	0.3
ro_test	0.1	0.1	0.1	0.1	0.1
d_grav	0.5	0.5	0.5	0.5	0.5
loc_zc	10^{-4}	10^{-3}	10^{-4}	10^{-5}	0.005
dtmax	200	200	300	200	300
ini0	6	5	6	6	4
n_atm	75	75	75	100	50
kipp	.FALSE.	.FALSE.	.TRUE.	.FALSE.	.TRUE.
en_masse	.TRUE.	.FALSE.	.TRUE.	.TRUE.	.TRUE.
mvt_dis	.TRUE.	.TRUE.	.FALSE.	.TRUE.	.FALSE.
n_max	1000	1000	1000	1000	1000
dn_fixe	0.05	0.05	0.1	0.05	0.05

*2 sans diffusion, 3 avec diffusion

- LOG_TEFF=-3.7d0 : arrêt si cette valeur de $\log_{10} T_{\rm eff}$ est traversée, dans le sens croissant si log_teff est positif, dans le sens décroissant si log_teff est négatif,

- NB_MAX_MODELES=500 : arrêt après avoir calculé le nombre de modèles indiqué, éventuellement 0.

- T_STOP=1.d7 : arrêt, si au centre, la température dépasse cette valeur,

- X_STOP=-0.1d0 : arrêt si cette valeur de X, au centre, est traversée.

2.7 Arguments de la NAMELIST NL_CHIM

- GRILLE_FIXE=.TRUE. : on utilisation d'une grille fixe pour la composition chimique et la diffusion du moment cinétique,

- NOM_ABON : nom de la mixture initiale[2] :

  - enhan_al : abondances α-enhanced de Allard,
  - enhan_cha : abondances α-enhanced de Chaboyer,
  - enhan_w : abondances α-enhanced de Weiss,
  - meteorites_ag : abondances météoritiques de Anders & Grevesse 89,
  - meteorites_gs : abondances météoritiques de Grevesse & Sauval 98,
  - mixture : forme personnalisée de la compilation d'abondances décrite au § 3.2 (Page 49),
  - solaire_gn : mixture solaire de Grevesse & Noels 93,
  - solaire_gs : mixture solaire de Grevesse & Sauval 98.

  Ces abondances sont initialisées dans la routine abon_ini.

- MODIF_CHIM=.FALSE. : même s'ils existent dans l'environnement du calcul, CESAM ignorera les fichiers des types modif_mix, rap_iso et vent qui permettent de modifier respectivement la composition chimique initiale *cf.* § 3.4 (Page 50), les rapports isotopiques *cf.* § 3.3 (Page 49) et la composition chimique du vent *cf.* § 3.5 (Page 50). Cette disposition de but **sécuritaire** permet d'éviter de prendre en compte des fichiers qui, par oubli, n'ont pas été soustraits de l'environnement.

- GARDE_XISH=.FALSE. : ce paramètre fait l'objet du paragraphe suivant,

- X0=0.7 : abondance initiale, par unité de masse, de H,

- Y0=0.28 : abondance initiale, par unité de masse, de He,

- ZSX0=0.d0 : valeur initiale de Z/X,

  - la valeur de Y est toujours y0,

[2] *cf.* équation d'état OPAL: WEB site http://www-phys.llnl.gov/V_Div/OPAL/opal.html.

- si `zsx0` $\leq$ `0` l'abondance initiale d'hydrogène est initialisée à `x0`, Z=1-X-Y est déduit de `x0` et `y0`,
- sinon, *i.e.* `zsx0` $>$ `0`, l'abondance initiale d'hydrogène est déduite de `y0` et de `zsx0`, la valeur indiquée pour `x0` est ignorée.

CESAM2*k* offre les possibilités d'utiliser une mixture originale, *cf.* § 3.2 (Page 49), de modifier les rapports isotopiques, *cf.* § 3.3 (Page 49), ou encore les rapports d'abondances, *cf.* § 3.4 (Page 50).

2.7.1 Alternative : conservation métal/H ou métal/Z

Pour une mixture initiale, éventuellement modifiée ou non, *cf.* § 3.4 (Page 50), l'alternative `GARDE_XISH=.TRUE.` (*respt.* `GARDE_XISH=.FALSE.`) permet de fixer la valeur initiale de Z (*respt.* X) de façon à conserver le rapport `métal/H` (*respt.* `métal/Z`). La valeur initiale de Y est **TOUJOURS** celle `y0` lue dans le fichier de données.

Ainsi, en codant `garde_xish=.TRUE.`, les rapports d'abondances métal/H, de la mixture utilisée seront ceux de la mixture initiale (par exemple `solaire_gn`), et non pas les rapports métal/Z de la mixture initiale; la valeur de Z correspondra à celle de la mixture initiale et l'abondance initiale d'hydrogène par gramme sera X=1-Y-Z et non pas `x0`.

A l'inverse, en codant `GARDE_XISH=.FALSE.`, les rapports métal/Z seront conservés[3], les abondances des métaux seront adaptées à la valeur de Z=1-X-Y lue – ou déduite – des valeurs du fichier de données; les rapports métal/H seront, de fait, modifiés. Pour une **calibration solaire** pour laquelle on vise à atteindre une valeur fixée de Z/X, en conservant, en principe, les rapports métal/Z initiaux on codera donc `garde_xish=.FALSE.`.

$\boxed{\textbf{PB}}$ L'utilisation de GARDE_XISH=.TRUE. **EST DÉLICATE**, il est conseillé de bien vérifier, dans le fichier mon_modele.lis, que les valeurs des abondances utiliséees correspondent bien à ce que l'on désire.

$\boxed{\textbf{PB}}$ Il faut être conscient que les modifications de la mixture ne seront pas prises en compte dans le calcul des opacités.

2.8 Arguments de la NAMELIST `NL_CONV`

- `NOM_CONV` : nom de la routine calculant le gradient de température dans les zones convectives.
  - `conv_a0` : convection MLT, avec longueur de mélange $l \to 0$ aux limites ZR/ZC,
  - `conv_cgm_reza` : convection suivant Canuto & Mazitelli avec $l = \alpha H_{\rm p}$, tient compte de la prescription de Bernkopf,
  - `conv_cm` : convection suivant Canuto & Mazitelli avec $l = \alpha H_{\rm p}$,

[3]De façon cohérente avec les tabulations d'opacités.

- conv_cm_reza : convection suivant Canuto & Mazitelli avec $l = \alpha H_\mathrm{p}$, tient compte de δ,
- conv_jmj : convection MLT avec $l = \alpha H_\mathrm{p}$.

- alpha=1.7d0 : longueur de mélange,
- ovshts=0.05d0 : coefficient d'overshoot supérieur,
- ovshti=0.1d0 : coefficient d'overshoot inférieur,
- jpz=.FALSE. : utilisation des prescriptions de JpZ,
- cpturb=0.d0 : coefficient de pression turbulente,
- ledoux=.FALSE. : utilisation du critère de Ledoux[4].

PB L'utilisation d'un paramètre de pression turbulente non nul est délicate.

2.9 Arguments de la NAMELIST NL_DIFF

- DIFFUSION=.TRUE. : on tiendra compte de la diffusion microscopique des éléments chimiques, éventuellement de la diffusion du moment cinétique,
- NOM_DIFFM : nom de la routine calculant les coefficients de diffusion microscopique :
  - diffm_br : calcul des coefficients de diffusion microscopique suivant le formalisme de Burgers en tenant compte, éventuellement, des accélerations radiatives,
  - diffm_mp : calcul des coefficients de diffusion microscopique suivant le formalisme simplifié de Michaud & Proffit,
  - diffm_0 : coefficients de diffusion microscopique nuls.
- NOM_DIFFT : nom de la routine calculant les coefficients de diffusion turbulente :
  - difft_nu : calcul des coefficients de diffusion turbulente incluant la diffusivité radiative.
- D_TURB=1.d1 : coefficient de diffusion turbulente isotrope,
- RE_NU=1.d0 : coefficient de diffusivité radiative,
- NOM_FRAD : nom de la routine de calcul des accélérations radiatives,
  - alecian1 : calcul des accélérations radiatives suivant le premier formalisme de G.Alécian,

[4]A utiliser qu'avec précautions.

- `alecian2` : calcul des accélérations radiatives suivant le second formalisme de G.Alécian,
- `no_frad` : les accélérations radiatives seront ignorées.

PB `Les accélérations radiatives ne sont pas disponibles dans la présente version de CESAM2k.`

2.10 Arguments de la NAMELIST NL_ROT

- `ROT_SOLID = .TRUE.` : la vitesse angulaire locale sera la même en tout point du modèle:
  - `w_rot >= 0` : le moment cinétique est localement conservé, ses variations locales sont transformées en énergie interne,
  - `w_rot < 0` : le moment cinétique est globalement conservé, la vitesse angulaire locale `wrot`, la même en tout point du modèle, variera au cours du temps; elle sera réajustée au cours des itérations à l'issue du calcul du moment cinétique total dont on assurera la conservation en faisant varier `wrot`.
- `ROT_SOLID = .FALSE.` : il y a diffusion du moment cinétique dans les zones radiatives, mélange dans les zones convectives. Les cas incohérents `rot_solid =.FALSE.` avec `w_rot=0` ou `DIFFUSION=.FALSE.` sont éliminés.

PB `La diffusion du moment cinétique n'est pas disponible dans la présente version de CESAM2k.`

- `W_ROT=50.d0` : vitesse angulaire initiale,
- `UNIT` : unité utilisée pour la vitesse angulaire initiale; différents choix sont proposés de façon à s'adapter facilement aux observations :
  - `jours` : période initiale en jours correspond à la vitesse angulaire initiale de rotation de la couche externe,
  - `kms/s` : vitesse de la couche externe en kilomètres/seconde,
  - `rad/s` : la vitesse angulaire initiale est en radians/seconde.
- `W_FORM=0.d0` : coefficient de forme du profil de la vitesse angulaire initiale, `w_form=0` correspond à une vitesse uniforme,
- `NOM_DIFFW` : nom de la routine du calcul des coefficients de diffusion du moment cinétique :
  - `diffw_mpz` : coefficients de Matis, Palacios & Zahn.
- `NOM_PERTW` : nom de la routine de calcul de perte de moment cinétique,
  - `pertw_sch` : variation de vitesse angulaire proportionnelle à Ω^3 (Schumanish),

  - pertw_loc : perte de moment cinétique proportionnelle à l'énergie cinétique de rotation locale,
  - pertw_ptm : perte de moment cinétique conséquence d'une perte de masse,
  - pertw_0 : pas de perte de moment cinétique.

- P_PERTW=-1.d-9 : paramètre de perte de moment cinétique attaché respectivement à chacune des routines précédentes,
- PW_EXTEND=1.d-2 : définit, en proportion de la masse totale, la partie externe de l'extension de la zone de perte de moment cinétique.

2.11 Arguments de la NAMELIST NL_ETAT

- NOM_ETAT : nom de la routine d'équation d'état :
  - etat_ceff : équation d'état de Eggleton et al., avec corrections Coulombiennes (*appel à* etat_eff *en cas de difficulté*);
  - etat_eff : équation d'état de Eggleton et al. (*appel à* etat_gong2 *en cas de difficulté*);
  - etat_gong1 : équation d'état de gong1, seuls H et He sont pris en compte et supposés totalement ionisés;
  - etat_gong2 : équation d'état de gong2, seuls H et He4 sont seuls pris en compte;
  - etat_mhd : équation d'état de Mihalas, Hummer & Dappen *appel à* etat_eff *en cas de difficulté* utilise les tables en binaire mhd1.bin ... mhd7.bin construites dans le sous-directory SUN_STAR_DATA, voir § 1.7.4 (Page 26); en binaire mhd1.bin ... mhd7.bin construites dans le
  - etat_opal : équation d'état de opal 1991, (*appel à* etat_eff *en cas de difficultés*) utilise les tables en binaire eos_opal*.bin construites dans le sous-directory SUN_STAR_DATA, voir § 1.7.3 (Page 23).
  - etat_opalZ : équation d'état de opal 2001, (*appel à* etat_eff *en cas de difficultés*) utilise les tables en binaire eos_opalZ*.bin construites pour **Z fixé** dans le sous-directory SUN_STAR_DATA, voir § 1.7.3 (Page 23). Par rapport à la version 1991, les données de la version 2001 ont été étendues et améliorées, des bugs ont été corrigés. Les données ne permettent pas de dépasser $X = 0.8$. Les fichiers de données 2001 et 1991 sont incompatibles.
  - etat_opalX : équation d'état de opal 2001, se différenciant de etat_opalZ par l'extension du domaine de l'abondance d'hydrogène, $X \in [0, 1]$, dans le but de calculer des modèles sans zone convective externe étendue *avec diffusion*. Le prix à payer est $Z \equiv 0$, ce qui n'est pas très pénalisant, l'équation d'état étant peu sensible à Z. etat_opalX utilise **la** table en binaire eos_opalX construites pour **Z=0 fixé** dans le sous-directory SUN_STAR_DATA, voir § 1.7.3 (Page 23).

- `F_EOS='eos_opal_250.bin',7*' '` : noms des fichiers d'équation d'état (au plus 8).

2.12 Arguments de la NAMELIST `NL_OPA`

- `NOM_OPA` : nom de la routine de calcul des opacités moyenne de Rosseland :
  - `opa_gong` : opacités simplifiées (Kramers amélioré);
  - `opa_houdek9` : opacités de Houdek version 9, (OPAL+Alexander), interpolation par rational B-spline; utilise les tables en binaire construites dans le sous-directory `SUN_STAR_DATA/V9`, voir § 1.7.2 (Page 22)
  - `opa_yveline` : opacités OPAL+Alexander interpolation et raccord d'Yveline; utilise tables en binaire `opa_yveline*.bin` construites dans le sous directory `SUN_STAR_DATA`, voir § 1.7.1 (Page 21);
  - `opa_yveline_lisse` : opacités OPAL+Alexander raccord d'Yveline, interpolation linéaire ou lissage; utilise les tables en binaire `opa_yveline*.bin` construites dans le sous-directory `SUN_STAR_DATA`, voir § 1.7.1 (Page 21).
- `F_OPA='opa_yveline.bin',7*' '` : noms des fichiers d'opacité (au plus 8).

2.13 Arguments de la NAMELIST `NL_NUC`

- `NOM_NUC` : nom de la routine de calcul des taux des réactions thermonucléaires et d'initialiation de la composition chimique :
  - `pp1` : calcul simplifié du cycle PP (ne permet pas la diffusion microscopique);
  - `ppcno9` : réactions PP+CNO, 9 éléments, H2, Li7, Be7 à l'équilibre;
  - `ppcno9Fe` : réactions PP+CNO, 9 éléments+Fe, H2, Li7, Be7 à l'équilibre;
  - `ppcno10` : réactions PP+CNO, 10 éléments, H2, Be7 à l'equilibre;
  - `ppcno10Fe` : réactions PP+CNO, 10 éléments+Fe, H2, Be7 à l'équilibre;
  - `ppcno10BeBFe` : réactions PP+CNO, 10 éléments + Li6, Be9, B11, Fe avec H2 et Be7 à l'équilibre;
  - `ppcno11` : réactions PP+CNO, 11 éléments, Be7 à l'équilibre;
  - `ppcno12` : réactions PP+CNO, 12 éléments;
  - `ppcno12Be` : réactions PP+CNO, 12 éléments + Be9;
  - `ppcno12Li` : réactions PP+CNO, 12 éléments + Li6;
  - `ppcno12BeBFe` : réactions PP+CNO, 12 éléments + Li6, Be9, B11, Fe;
  - `ppcno3a9` : réactions PP+CNO+3α, 9 éléments, H2, Be7 à l'équilibre;

- ppcno3ac10 : réactions PP+CNO+3α+C, 10 éléments H2, Li7, Be7 à l'équilibre.

- nom_nuc_cpl : nom de la compilation des réactions thermonucléaires :
  - Adelb : Compilation d'Adelberger et al. [1998];
  - Cau-Fow : Compilation de Caughlan & Fowler [1988];
  - NACRE : Compilation de NACRE (Angulo et al. [1999]).
- MITLER=.FALSE. : effet d'écran selon Mitler.

2.14 Arguments de la NAMELIST NL_ATM

- NOM_ATM : nom de la routine de restitution de l'atmosphère :
  - lim_atm : restitution de l'atmosphère à partir d'une loi $T(\tau)$,
  - lim_gong1 : restitution simplifiée de l'atmosphère, cas de GONG1,
  - lim_tau1 : restitution simplifiée de l'atmosphère, atmosphère monocouche.
- NOM_TDETAU : nom de la loi $T(\tau)$ utilisée pour la restitution de l'atmosphère :
  - edding : loi $T(\tau)$ totalement radiative d'Eddington;
  - hopf : loi $T(\tau)$ totalement radiative de Hopf;
  - K5750 : loi $T(\tau)$ non totalement radiative, dérivée du modèle d'atmosphère solaire de Kurucz, $T_{\rm eff} = 5750\,{\rm K}$;
  - K5777 : loi $T(\tau)$ non totalement radiative, dérivée du modèle d'atmosphère solaire de Kurucz, $T_{\rm eff} = 5777\,{\rm K}$;
  - roger00 : loi $T(\tau)$ non totalement radiative, dérivée de modèles d'atmosphère de Kurucz, de métallicité $[\frac{\rm Fe}{\rm H}] = 0.0$;
  - roger02 : loi $T(\tau)$ non totalement radiative, dérivée de modèles d'atmosphère de Kurucz, de métallicité $[\frac{\rm Fe}{\rm H}] = +0.2$;
  - roger05 : loi $T(\tau)$ non totalement radiative, dérivée de modèles d'atmosphère de Kurucz, de métallicité $[\frac{\rm Fe}{\rm H}] = -0.5$;
  - roger10a : loi $T(\tau)$ non totalement radiative, dérivée de modèles d'atmosphère de Kurucz, de métallicité $[\frac{\rm Fe}{\rm H}] = -1.0$ avec éléments α "enhanced".
- TAU_MAX=10.d0 : profondeur optique du raccord avec l'enveloppe,
- LIM_RO=.TRUE. : condition limite externe en densité, sinon en pression.

2.15 Fichiers de données des versions antérieures

CESAM2k accepte d'exploiter les fichiers de données de CESAM4 et CESAM5; dans ces versions antérieures, il fallait créer un module exécutables adapté à la physique utilisée. Pour exploiter un fichier de données de CESAM4 ou de CESAM5 avec CESAM2k, il faut indiquer la physique à utiliser et introduire les données manquantes. Pour ce faire, CESAM2k utilise les données et les options physiques explicitées dans un fichier de nom physique45 placé dans l'environement du calcul. On trouvera un exemple du fichier `physique45` au § 3.10 (Page 57). En l'absence d'un tel fichier, des données et des options physiques sont définies par défaut dans la routine lit_nl_45.f subordonnée de la routine lit_nl.f. Ces options physiques sont les suivantes :

```
NOM_CHEMIN=~/SUN_STAR_DATA/'
NOM_CTES='ctes_94'
NOM_DES='des_m'
NOM_OUTPUT='no_output'
NOM_PERTM='pertm_ext'
NOM_CONV='conv_jmj'
NOM_DIFFM='diffm_mp'
NOM_DIFFT='difft_nu'
NOM_ETAT='etat_eff'
NOM_OPA='opa_yveline'
NOM_NUC='ppcno9'
NOM_NUC_CPL='NACRE'
NOM_ATM='lim_atm'
NOM_TDETAU='hopf'
```

CESAM2k accepte aussi les fichiers de données de ses premières versions d'évaluation. Le but poursuivi dans l'exploitation des anciens fichiers de données est d'offrir la possibilité de poursuivre des calculs précédents, de ce fait, seules les fonctionnalités des anciennes versions peuvent être prises en compte.

2.16 Exemple d'exploitation sous UNIX/LINUX,

Sous UNIX/LINUX il est conseillé de créer et de laisser le module exécutable cesam2k.out dans le sous-directory `SOURCE`. Il pratique de créer un sous-directory propre à chaque application dans lequel on disposera des fichiers de données de type mon_modele.don propres à l'application et les fichiers d'initialisation tels que (m020.zams), de réglage et de personnalisation.
Par exemple, pour calculer des modèles de ξ Hya ($3M_\odot$) de la ZAMS jusqu'à 400 My :

1. Créer un sous-directory `KSI_HYA`
2. Copier les fichiers m020.zams[5] et mon_modele.don du sous-directory `EXPLOIT`.
3. Renommer ksi_hya.don le fichier de données.

[5]Utiliser des fichiers d'initialisation ASCII ou binaire de $3M_\odot$ si on en dispose.

4. Adapter la physique et les paramètres de ce fichier au problème de ξ Hya.

Pour utilisation ultérieure, lancer le calcul d'un modèle préliminaire de ZAMS (`AGEMAX = 0.d0`) adapté aux paramètres et à la physique adoptés; d'abord avec une précision réduite (`PRECISION='np'`) en utilisant le modèle d'initialisation en ASCII, puis avec (`PRECISION='pr'`) en utilisant le modèle binaire ksi_hya_B.hom obtenu à l'étape précédente. Copier le fichier ksi_hya_B.hom obtenu à l'issue de cette dernière étape dans le fichier ksi_hya_ini_B.hom ce dernier servant, par la suite, à initialiser le calcul de modèles utilisant des paramètres numériques ou physiques (longueur de mélange, théorie de la convection, précision, *etc...*) quelque peu différents de ceux qui ont été utilisés pour construire le fichier d'initialisation.
Dans le fichier ksi_hya.don indiquer l'âge à atteindre : (AGEMAX = 400.d0) et lancer l'exécution : cesam2k.out en l'initialisant avec ksi_hya_ini_B.hom.
Pour faire divers modèles en changeant un ou plusieurs paramètres (âge, overshoot, *etc...*), créer pour chaque cas particulier un fichier de données en adaptant le fichier ksi_hya.don.

2.17 Routines d'exploitation

Des programmes du sous directory `EXPLOIT` permettent une exploitation des fichiers ASCII créés par CESAM2*k* :

- `calib2k` : formation du fichier de données mon_modele.don pour calibration d'un modèle solaire.
- `des2k_abon` : dessin des abondances en fonction de la masse ou du rayon.
- `des2k_hr` : dessin du diagramme HR.
- `des2k_osc` : dessin de 1, 2 ou 3 modèles en fonction du rayon, de la masse avec des possibilités de zoom.
- `des2k_ZC` : dessin de l'évolution des zones convectives en fonction du temps.
- `f037_2k` : extension de modèles.

PB `Ce programme n'est pas disponible dans la présente version de` CESAM2*k*.

2.18 Scripts d'exploitation

Les scripts UNIX (en shell `csh`) suivants du sous directory `SCRIPTS` facilitent l'installation et l'exploitation[6] de CESAM2*k* :

- `calib2k_pms` : procédure de calibration d'un modèle solaire avec PMS.
- `calib2k_zams` : procédure de calibration d'un modèle solaire initialisé sur la ZAMS homogène.

[6] Les options de compilation sont celles du compilateur LINUX LF95, il convient de les adapter à celles du compilateur utilisé.

- `compile2k` : script de compilation avec optimisation.
- `compile2k-dbg` : script de compilation pour debug.
- `compile2k-dbg_list` : script de compilation pour debug pour un ensemble de routines d'une liste.
- `compile2k_list` : script de compilation avec optimisation pour un ensemble de routines d'une liste avec création ou mise à jour d'une bibliothèque.
- `genere_cesam2k-dbg` : script créant la bibliothèque et l'exécutable `cesam2k-dbg.out` de debug,
- `genere_cesam2k` : script créant la bibliothèque et l'exécutable `cesam2k.out` d'exploitation,
- `evol2k_pms` : procédure pour l'évolution à partir de la PMS homogène.
- `evol2k_zams` : procédure pour l'évolution à partir de la ZAMS homogène.
- `exe2k` : procédure d'exécution d'un programme avec optimisation
- `exe2k-dbg` : procédure d'exécution d'un programme avec debug.
- `lib_del` : script permettant de supprimer une routine d'une bibliothèque.
- `lib_del_repl` : script permettant de supprimer puis de remplacer une routine dans une bibliothèque.
- `lib_repl` : script permettant de remplacer une routine dans une bibliothèque.
- `on_voit` : script permettant de créer un fichier postscript à partir d'un fichier dvi.
- `rempl2k_mod` : script permettant de compiler, remplacer dans la bibliothèque, un ou plusieurs modules et de créer le module exécutable `cesam2k.out`.
- `rempl2k_mod-dbg` : script permettant de compiler, remplacer dans la bibliothèque, un ou plusieurs modules et de créer le module de debug exécutable `cesam2k-dbg.out`.
- `def_standard.tex` : ensemble de définitions LaTeX personnalisées.

Chapter 3

Personnalisation

Bien que l'exécutable soit construit un fois pour toutes, il est possible d'adapter de façon externe un certain nombre de paramètres. Ces adaptations s'effectuent, à la demande, par l'intermédiaire de fichier à disposer dans le sous-directory où s'effectue le calcul. Ne seront concernés que les modèles calculés dans cet environnement. De façon générale, dans le directory concerné, une personnalisation peut affecter soit un modèle particulier, soit tous les modèles calculés dans le directory. CESAM recherche d'abord si une personnalisation existe pour le modèle à calculer puis, à défaut, si elle existe pour tous les modèles du directory. Des exemples de tels fichiers de personnalisation se trouvent dans le sous-directory `EXPLOIT` de la distribution. On peut ainsi :

- Avoir des commentaires d'interactivité en une langue différente du français[1].
- Utiliser une mixture initiale différente de celles implémentées.
- Modifier les rapports d'abondances d'une mixture.
- Redéfinir les rapports isotopiques.
- Créer un fichier de sortie en ASCII contenant des quantités en nombre et ordre différents de ce qui est implémenté.
- Utiliser des réglages personnalisés.
- Dessiner des boites correspondants à des objectifs à atteindre et adapter les échelles du diagramme HR tracé "on line".
- Adapter les dimensions des cadres et le nom du device à utiliser.
- Utiliser les fichiers de données de l'une des versions précédentes de CESAM.

Par sécurité, ainsi qu'il est explicité pour chaque cas particulier, certains des fichiers de personnalisation concernant la composition chimiques ne seront pris en compte que si le paramètre `modif_chim` de la NAMELIST `nl_chim` du fichier de données est `.TRUE.` *cf.* § 2.7 (Page 37).

[1] Seulement en anglais dans la présente version.

3.1 Langue

Une alternative pour avoir des commentaires d'interactivité en une langue différente du français :

1. Si on désire **toujours** avoir des commentaires en une langue différente du français, **avant** la compilation, dans la routine cesam du sous-directory SOURCE, commenter ou supprimer les instructions (vers les lignes 270-278) :

```
      INQUIRE(file='langue',exist=ok)
      IF(ok)THEN
       OPEN(unit=30,form='formatted',status='old',file='langue')
       READ(30,nl_langue) ; CLOSE(UNIT=30)
      ELSE
       WRITE(*,30)
30     FORMAT(/,'Sometimes CESAM can speak english,',/,
     1 'see aide_mem2k, chapter personnalisation ')
       langue='francais'
      ENDIF
```

   et modifier l'instruction de la ligne suivante (279) :

```
c langue='francais' ===>    langue='english'
```

   Les commentaires, ici en anglais, seront utilisés pour toute exploitation.

2. Si on désire n'avoir qu'occasionnellement les commentaires en anglais, introduire dans l'environnement d'exploitation un fichier de nom langue dans lequel coder :

```
&NL_LANGUE
langue='english'
/
```

   Dans toutes les exploitations effectuées dans cet environnement les commentaires apparaitront en anglais.

Par défaut, *i.e.* en l'absence de l'une des dispositions précédentes, les commentaires seront en français.

$\boxed{\textbf{PB}}$ Dans la version actuelle de CESAM2*k*, seulement une partie des principaux commentaires a été traduite en anglais.

$\boxed{\textbf{PB}}$ **Call for man power** : il est facile de transposer dans une autre langue les algorithmes créés pour l'anglais ou encore de compléter et améliorer la syntaxe des ces derniers, les volontaires sont attendus.

3.2 Mixture

Pour utiliser une mixture initiale différente[2] de celles implémentées, coder `nom_abon='mixture'` dans le fichier de données et, dans l'environnement, placer un fichier de nom mixture contenant la mixture désirée en Dex. CESAM **refuse toute mixture dont la normalisation est différente de H=12**. L'exemple suivant se trouve dans le sous-directory `EXPLOIT` :

```
&NL_MIXTURE
ab(1)=12.00d0 , ab(2)=10.d0 , ab(3)=2.69d0  , ab(4)=2.15d0  , ab(5)=2.60d0
ab(6)=4.55d0  , ab(7)=3.97d0  , ab(8)=4.87d0  , ab(9)=1.56d0  , ab(10)=4.08d0
ab(11)=2.33d0 , ab(12)=3.58d0 , ab(13)=2.47d0 , ab(14)=3.55d0 , ab(15)=1.45d0
ab(16)=3.21d0 , ab(17)=1.5d0  , ab(18)=2.52d0 , ab(19)=1.12d0 , ab(20)=2.36d0
ab(21)=-1.17d0 , ab(22)=1.02d0 , ab(23)=0.d0   , ab(24)=1.67d0 , ab(25)=1.39d0
ab(26)=3.5d0  , ab(27)=0.92d0 , ab(28)=2.25d0
/
```

On trouve dans la routine abon_ini.f l'ordre et l'identification des éléments par leurs indices en fait, celle des opacités OPAL. Dès lors **TOUS** les modèles calculés dans ce directory, s'ils utilisent l'option mixture, auront des abondances initiales conformes à celles du fichier mixture. Une autre possibilité permet de personnaliser davantage; elle consiste à donner au fichier mixture le nom générique du modèle avec l'extension `.mix`, Ex : `mon_modele.mix`. Dès lors **SEUL** le modèle `mon_modele` aura la composition chimique initiale désirée. Avec l'option mixture, CESAM cherche **en priorité** à utiliser le fichier `mon_modele.mix`, en son absence c'est le fichier mixture qui est utilisé; le calcul n'est effectué que si l'un de ces deux fichiers existe dans l'environnement.

$\boxed{\textbf{PB}}$ `Avec l'option nom_abon='mixture', les abondances relatives des métaux devraient, en principe, correspondre à celles des opacités, au moins, faute de mieux, pouvoir être prises en charge par la routine d'opacité utilisée.`

3.3 Rapports isotopiques

Pour utiliser des rapports isotopiques différents de ceux que CESAM prend par défaut, placer dans l'environnement du calcul un fichier de nom mon_modele.rap_iso dans lequel seront précisés les rapports isotopiques à utiliser. Si **tous** les modèles de l'environnement doivent être calculés avec des rapports modifiés, donner au fichier précédent le nom de rap_iso. Exemple :

```
&NL_RAP_ISO
be7sbe9=1.d-25,
be7sz=1.d-29,
c13sc12=1.10d-2,
h2sh1=3.01d-5,
he3she4=1.1d-4,
he3she4z=4.185d-4,
li6sli7=7.5d-2,
mg25smg24=0.0125,
mg26smg25=0.013,
ne22sne20=6.79d-2,
```

[2]Voir aussi § 4.2 (Page 60).

```
n15sn14=0.366d-2,
o17so16=0.038d-2
/
```

Les noms utilisés pour les variables suggèrent les notations des rapports isotopiques (be7sbe9 : rapport isotopique ^{7}Be/^{9}Be). `he3she4z` est le rapport isotopique ^{3}He/^{4}He sur la ZAMS lorsque le deutérium initial a été transformé en hélium3, `be7sz` est l'abondance, quasi nulle, du ^{7}Be initial dans Z.

Par sécurité ce fichier ne sera pris en compte que si le paramètre `modif_chim` de la NAMELIST `nl_chim` du fichier de données est `.TRUE.` *cf.* §2.3 (Page 32).

3.4 Personnalisation des rapports d'abondances

La mixture une fois définie, par exemple `solaire_gn` (Grevesse & Noëls 93), il est possible d'y modifier les abondances. Pour ce faire, créer dans l'environnement du calcul un fichier de nom mon_modele.modif_mix dans lequel seront indiquées les modifications en DeX à apporter. Si tous les modèles de l'environnement doivent être calculés avec ces modification, donner au fichier précédent le nom de modif_mix. Exemple se trouvant dans le directory EXPLOIT :

```
&nl_modif_mix
add_Li=2.d0 ; add_Be=0.d0 ; add_B=0.d0
add_C=-1.0d0 ; add_N=0.d0 ; add_O=-0.3d0 ; add_F=0.d0 ; add_Ne=0.d0
add_Na=0.d0 ; add_Mg=0.d0 ; add_Al=0.d0 ; add_Si=0.d0 ; add_P=0.d0
add_S=0.d0 ; add_Cl=0.d0 ; add_Ar=0.d0 ; add_K=0.d0 ; add_Ca=0.d0
add_Sc=0.d0 ; add_Ti=0.d0 ; add_V=0.d0 ; add_Cr=0.d0 ; add_Mn=0.d0
add_Fe=0.d0 ; add_Co=0.d0 ; add_Ni=0.d0 ; add_Z=-1.d0
/
```

Les notations suggèrent les quantités à modifier ainsi `add_Li=2.d0` est le nombre de DeX à ajouter à l'abondance du lithium. `add_Z=-1.d0` signifie que les abondances **de tous les métaux** seront diminuées d'un DeX. **Par sécurité** ce fichier ne sera pris en compte que si le paramètre `modif_chim` de la NAMELIST `nl_chim` du fichier de données est `.TRUE.` *cf.* §2.3 (Page 32).

Si on a codé `GARDE_XISH=.FALSE.` dans le fichier de données *cf.* §2.7.1 (Page 38), le paramètre `add_Z`, qui agit sur les métaux dans leur ensemble, n'a pas d'incidence dans la détermination des rapports métal/Z. A l'opposé, une valeur non nulle de `add_C`, par exemple, aura une incidence sur les rapports métal/Z.

3.5 Personnalisation de la composition chimique du vent.

Dans le cas standard, avec `Mdot` non nul, la composition chimique de la masse perdue ou gagnée par le 'vent' est celle de la couche la plus externe du modèle. Il est possible de personnaliser cette composition chimique afin qu'elle diffère de celle de la couche externe. Pour ce faire, placer dans l'environnement du calcul un fichier de nom mon_modele.vent dans lequel seront indiquées les fractions de masse de chaque élément chimique dans l'apport ou la perte de masse. Si tous les modèles de l'environnement doivent être calculés avec ces modification, donner au fichier précédent le nom de vent. Un exemple se trouve dans le sous-directory EXPLOIT :

```
&nl_vent
vt_H=0.7347,vt_He=0.2483,vt_Li=1.033e-08,vt_Be=1.72816e-10,
vt_B=4.859e-09,vt_C=2.899e-03,vt_N=8.493e-04,vt_O=7.885e-03,
vt_F=4.182e-07,vt_Ne=1.768e-03,vt_Na=3.501e-05,vt_Mg=6.736e-04,
vt_Al=6.078e-05,vt_Si=7.434e-04,vt_P=8.198e-06,vt_S=3.704e-04,
vt_Cl=4.924e-06,vt_Ar=7.315e-05,vt_K=3.845e-06,vt_Ca=6.541e-05,
vt_Sc=4.126e-08,vt_Ti=3.041e-06,vt_V=3.888e-07,vt_Cr=1.856e-05,
vt_Mn=1.357e-05,vt_Fe=1.287e-03,vt_Co=3.492e-06,vt_Ni=7.6092e-05,
p_vent=1.000e-03
/
```

Les notations sont suggestives, ainsi `vt_Li=1.033e-08` est la fraction masse du lithium dans le vent. Une renormalisation étant effectuée lors de l'utilisation, il n'est pas nécessaire de normaliser les fractions de masse figurant dans ce fichier, certaines peuvent être nulles et même négatives. Suivant le signe de `MDOT`, la quantité de matière, apportée ou soustraite, est supposée provenir d'une couche de matière de masse $p_{\rm vent} \times M_\star$ située sous la surface. Le paramètre $p_{\rm vent}$ étant, hélas, arbitraire, le fichier vent peut être construit[3] par le programme fichier_vent du sous-directory EXPLOIT.

Par sécurité ce fichier ne sera pris en compte que si le paramètre `modif_chim` de la NAMELIST `nl_fdon` du fichier de données est `.TRUE.` *cf.* § 2.7 (Page 37).

3.6 Ascii

Pour créer un fichier de sortie en ASCII personnalisé, coder `nom_output='ascii'` dans le fichier de données et, dans l'environnement, placer un fichier de nom sortie_ascii[4] dans lequel est définie la structure souhaitée. Le codage, un peu complexe, est décrit dans la routine output.f, les indices des variables sont identifiés dans la routine ascii.f du sous-directory `SOURCE`. Dans l'exemple :

```
12  3
1 2 3 4 5 6 9 10 11 12 13 15
1 2 3
.FALSE.
.TRUE.
.TRUE.
Fichier pour sortie ascii:
```

Ces quantités ont les significations suivantes :

- 12 3 ⟶ on écrira 12 variables globales et 3 variables locales,
- 1 2 3 4 5 6 9 10 11 12 13 15 ⟶ indices des 12 variables globales (mstar*msol, rtot*rsol, ltot*lsol, z0, x0, alpha, X dans ZC, Y dans ZC, d2p, d2ro, age, vsal, w_rot initial).
- 1 2 3 ⟶ indices des 3 variables locales : Rayon, log(M/Mtot), Température)'.

[3]La composition chimique de l'exemple est celle de la mixture météoritique de Grevesse & Sauval 98. Par ailleurs il n'est pas nécessaire de respecter la 'case'.

[4]Dont un exemple se trouve dans le sous-directory `EXPLOIT`.

- .FALSE. ⟶ la masse sera écrite en fraction de la masse totale et non en DeX.
- .TRUE. ⟶ tabulation du centre à la surface.
- .TRUE. ⟶ on ajoute la composition chimique locale à la suite des variables locales.
- Fichier pour sortie ascii : texte d'entête du fichier de sortie.

Le nom générique du fichier de données est repris pour former celui du fichier de sortie en ASCII : avec mon_modele.don le nom du fichier de sortie est mon_modele-ascii. Avec le fichier cité comme exemple on a obtenu :

```
Fichier pour sortie ascii: mon_modele-ascii
CESAM2k version 0.0.0.0 lagr colloc  1  2 np no diffus, 06 Juillet 2003 17h46
Physique utilisée: etat_eff, opa_yveline_lisse, conv_jmj, ppcno9, NACRE
solaire_gn, lim_atm, hopf, perte_ext, diffm_mp, difft_nu, ctes_94
 10  H1  He3  He4  C12  C13  N14  N15  O16  O17  Si28
       541        12        13        10       -1
 1.989190000000E+33 6.010741880620E+10 2.478601588572E+33 2.000000000000E-02 7.000000000000E-01
 1.800000000000E+00 7.000000000000E-01 2.800000000000E-01-4.241618755712E+01-2.519592625187E+01
 0.000000000000E+00 0.000000000000E+00
 0.000000000000E+00 0.000000000000E+00 1.359013041915E+07 7.000000000000E-01 8.826927693868E-05
 2.799117307231E-01 3.425410508692E-03 4.128413464627E-05 1.059170894932E-03 4.168107978225E-06
 9.641655715570E-03 3.903146720978E-06 5.824407491460E-03
 1.302042603905E+09 1.175206179724E-08 1.353639070045E+07 7.000000000000E-01 8.826927693868E-05
 2.799117307231E-01 3.425410508692E-03 4.128413464627E-05 1.059170894932E-03 4.168107978225E-06
 9.641655715570E-03 3.903146720978E-06 5.824407491460E-03
           ......................................................

 6.014328252162E+10 1.000000000164E+00 4.543950048106E+03 7.000000000000E-01 8.826927693868E-05
 2.799117307231E-01 3.425410508692E-03 4.128413464627E-05 1.059170894932E-03 4.168107978225E-06
 9.641655715570E-03 3.903146720978E-06 5.824407491460E-03
 6.014636429604E+10 0.000000000000E+00 4.543482782113E+03 7.000000000000E-01 8.826927693868E-05
 2.799117307231E-01 3.425410508692E-03 4.128413464627E-05 1.059170894932E-03 4.168107978225E-06
 9.641655715570E-03 3.903146720978E-06 5.824407491460E-03
```

Comme pour les abondances initiales *cf.* § 3.2 (Page 49), il est possible de ne créer ce fichier de sortie que pour un modèle particulier en donnant au fichier de sortie sortie_ascii le nom du modèle avec l'extension `.ascii`; exemple : mon_modele.ascii. S'il existe dans l'environnement, ce seront les paramètres de ce fichier qui seront pris en compte pour **LE** modèle de nom mon_modele. En son absence ce seront les quantités du fichier sortie_ascii, à défaut, il n'y aura pas de sortie ASCII.

3.7 Réglages

Les réglages, définis dans la routine cesam.f en fonction du paramètre `PRECISION` du fichier de données mon_modele.don, peuvent être personnalisés en codant : `PRECISION='rg'` et en disposant dans l'environnement un fichier de nom : reglages[5] comportant les réglages à utiliser. Exemple d'un tel fichier reglages :

```
&NL_RLG
m_qs=2,
m_ch=3,
m_rot=3,
m_rl=8,
```

[5]Dont un exemple se trouve dans le sous-directory `EXPLOIT`.

```
m_tds=2,
m_ptm=2,
ordre=1,
precix=1.d-4,
precit=0.1d0,
ro_test=0.1d0,
psi0=0.05d0,
d_grav=0.5d0,
loc_zc=5.d-4,
dtmax=200.d0,
dt0=1.d0,
ini0=5,
n_atm=75,
kipp=.TRUE.,
en_masse=.TRUE.,
ctel=0.0d0,
ctep=-1.d0,
ctem=15.d0,
cter=0.d0,
ctet=0.d0,
mvt_dis=.TRUE.,
n_max=2000,
dn_fixe=0.05d0,
dpsi=0.05d0,
d_conv=1.d13
/
```

Les significations des paramètres sont les suivantes :

```
m_qs=2: ordre des splines pour les variables quasi-stat.
m_ch=3: ordre des splines pour l'interpolation de la comp.chim.
m_rot=3: ordre des splines pour l'interpolation du moment cinétique
m_rl=8: ordre des splines pour le lissage des coefficients de la rotation
m_tds=2: ordre des splines pour interpolation du TdS
m_ptm=2: ordre des spl. pour inter. de la masse (perte de masse)
ordre=1: ordre du schéma d'int. des réac.nuc. avec rk_imps
precix=1.d-4: précision sur les itérations NR
precit=0.1d0: var. rela. max. pour int. temp. de la comp. chim.
ro_test=0.1d0: test de variation du TdS si ro > ro_test
psi0=0.05d0: constante de répartition à assurer
d_grav=0.5d0: variation maximale du TdS
loc_zc=5.d-4: précision de la localisation des limites ZR/ZC
dtmax=200.d0: pas temporel maximum
ini0=5: nb. iter. N-R avec réestim. de comp.chim. et des lim. ZR/ZC
n_atm=75: nb. couches pour l'atmosphère restituée
kipp=.FALSE.: approximation de Kippenhahn ou TdS=dE+PdV
en_masse=.TRUE.: variables quasi_stat. en lagrangien
ctel, ctep, ctem, cter, ctet: facteurs de répartition
```

```
mvt_dis=.TRUE.: ajustement de comp.chim. suivant mvt. des discon.
n_max=max(n_max,1000): nb. max de couches
dn_fixe=0.05d0: taux pour modif de grille fixe en comp.chim
dpsi=0.02: variation max pour modification de n_qs
d_conv=1.d13
```

L'ajustement des constantes de répartition **est délicat**. Les valeurs par défaut sont `ctel=0, ctep=-1, ctem=15, cter=0, ctet=0`. Prendre garde que `ctep` et `ctet` doivent être négatives car elles affectent respectivement la pression et la température, quantités décroissantes du centre à la surface.

Comme pour les abondances initiales *cf.* § 3.2 (Page 49), il est possible de plus personnaliser ces réglages en donnant au fichier des réglages celui du modèle avec l'extension `.rg` Ex : mon_modele.rg. S'il existe dans l'environnement ce seront les paramètres de ce fichier qui seront pris en compte pour **LE** modèle de nom mon_modele. En son absence ce seront les quantités du fichier reglages.

3.8 Zoom

Il est possible de personnaliser le dessin "on line" du diagramme HR en y dessinant un rectangle cible à atteindre par le trajet évolutif ainsi que de modifier l'échelle du diagramme en centrant celui-ci autour de la valeur du point cible à des distances précisées en DeX de $\log T_{\rm eff}$ et $\log L$. On peut aussi tracer une cible de [Fe/H] à atteindre dans le cadre du dessin des abondances en fonction de la masse. Pour ce faire, coder `nom_des='zoom'` dans le fichier de données et, dans l'environnement placer un fichier de nom zoom, dans lequel sont introduites les données nécessaires. Exemple d'un tel fichier zoom[6] :

```
&NL_DES
teff_des=6530.,
dteff_des=50., -30.,
zoom_t=0.2, -0.1,
l_des=6.844,
dl_des=0.6, -0.4
zoom_l=0.1, -0.2,
fesh_des=0.05,
dfesh_des=0.03, -0.04,
logteff_max=4.6,
logteff_min=3.5,
logl_max=5.,
logl_min=2.5
/
```

- Il n'y a pas de personnalisation si `teff_des` et/ou `l_des` est négatif ou nul.
- Le zoom n'est effectif qu'avec `zoom_t` et `zoom_l` non nuls.
- Pour ne pas dessiner de cible en métallicité coder `fesh_des=100`.

[6]Exemple se trouvant dans le sous-directory `EXPLOIT`.

- Les valeurs des abscisses et ordonnées maximales *i.e.* `logteff_max (min)`, `logl_max (min)` **prévalent** sur celles des cibles; dans l'exemple précédent les échelles logarithmiques en température effective et en luminosité variront respectivement dans les intervalles $[4.6, 3.5]$ et $[2.5, 5.]$.

- Pour **seulement** adapter les échelles du diagramme HR coder, par exemple :

```
&NL_DES
teff_des=0.,
dteff_des=0., 0.,
zoom_t=0., 0.,
l_des=0.,
dl_des=0., 0.
zoom_l=0., 0.,
fesh_des=0.,
dfesh_des=0., 0.,
logteff_max=3.6,
logteff_min=2.5,
logl_max=2.,
logl_min=-2.5
/
```

- Pour **seulement** cadrer le diagramme HR autour de la cible coder :

```
&NL_DES
teff_des=6530.,
dteff_des=50., -30.,
zoom_t=0.2, -0.1,
l_des=6.844,
dl_des=0.6, -0.4
zoom_l=0.1, -0.2,
fesh_des=100.,
dfesh_des=0., 0.,
logteff_max=0.,
logteff_min=0.,
logl_max=0.,
logl_min=0.
/
```

La signification des variables est la suivante :

- teff_des=6530. : température effective du centre de la cible,

- dteff_des=50., -30. : écarts en température effective,

- zoom_t=0.2, -0.1 : extension en température effective autour de la cible en DeX

- l_des=6.844 : luminosité du centre de la cible,
- dl_des=0.6, -0.4 : écarts en luminosité,
- zoom_l=0.1, -0.2 : extension en luminosité autour de la cible en DeX,
- fesh_des=0.05 : cible à atteindre en métallicité,
- dfesh_des=0.03, -0.04 : écarts de la cible en métallicité.
- logteff_max=4.6 : $\log_{10}$ abscisse maximale du diagramme HR,
- logteff_min=3.5 : $\log_{10}$ abscisse minimale du diagramme HR,
- logl_max=5. : $\log_{10}$, ordonnée maximale du diagramme HR,
- logl_min=2.5 : $\log_{10}$, ordonnée minimale du diagramme HR.

Il est possible de personnaliser davantage en donnant au fichier `zoom` le nom attaché au modèle avec l'extension `.zoom`, Ex : `mon_modele.zoom`. Dès lors les dispositions précédentes ne seront effectives **QUE POUR** le modèle `mon_modele`. Avec l'option `zoom`, CESAM cherche à utiliser le fichier `mon_modele.zoom`, en son absence, le fichier `zoom` est utilisé.

3.9 Device

Pour adapter les dimensions des cadres et le nom du device à utiliser, placer dans l'environnement du calcul un fichier de nom `device`, dans lequel sont introduites les données nécessaires. Exemple d'un tel fichier `device`[7] :

```
&NL_DEVICE
h=6.5,
dh=1.5,
ld=8.5,
dl=2.,
xleft=1.8,
ybot=1.4,
device='/xw'
/
```

La signification des variables est la suivante (unités en cm) :

- h=6.5 : hauteur des cadres,
- dh=1.5 : espace vertical entre les cadres,
- ld=8.5 : largeur des cadres,
- dl=2. : espace horizontal entre les cadres

[7]Exemple se trouvant dans le sous-directory `EXPLOIT`.

- xleft=1.8 : distance du dessin à la limite gauche de l'écran,
- ybot=1.4 : distance du dessin à la limite inférieure de l'écran,
- device='/xw' : nom du device.

On peut aussi modifier directement les valeurs de ces paramètres dans le module `mod_donnees` du sous-directory `SOURCE`.

3.10 Physique45

Un fichier, de nom physique45, placé dans l'environnement du calcul, *cf.* §2.15 (Page 44), permet d'utiliser un fichier de données de CESAM4 ou de CESAM5 avec une physique différente de celle prévue par défaut. Par exemple :

```
&NL_PHYSIQUE45
 NOM_CHEMIN=~/SUN_DATA/'
 NOM_CTES='ctes_85'
 NOM_DES='des_r'
 NOM_OUTPUT='osc_adia'
 NOM_PERTM='pertm_ext'
 NOM_CONV='conv_cm'
 NOM_DIFFM='diffm_0'
 NOM_DIFFT='difft_nu'
 NOM_ETAT='etat_ceff'
 NOM_OPA='opa_yveline'
 NOM_NUC='ppcno3ac10'
 NOM_NUC_CPL='Cau-Fow'
 NOM_ATM='lim_atm'
 NOM_TDETAU='roger_00'
/
```

On trouvera ce fichier dans le sous-directory `EXPLOIT`.

Chapter 4

Extensions

4.1 Ajout d'une chaîne de réactions thermonucléaires

Pour ajouter une chaîne de réactions thermonucléaires, il est souvent nécessaire d'ajouter un ou plusieurs isotopes et aussi une ou plusieurs réactions thermonucléaires.

- Pour ajouter un isotope[1] :

  1. Dans le module `mod_nuc` augmenter le `PARAMETER niso_tot` d'une unité, Exemple : `niso_tot=28`.
  2. Dans la routine `taux_nuc` :
     (a) Dans les déclarations, ajouter l'excès de masse de l'isotope et son symbole, Exemple : `Fe56=-60.6054d0`.
     (b) Ajouter la masse de l'isotope, (il est conseillé, d'ajouter la masse de l'isotope plutôt dans les routines de constantes physiques `ctes_85` et `ctes_95`, et par suite dans les déclarations du module `mod_donnees` et dans l'instruction de restriction `ONLY` de la routine `taux_nuc`, Exemple : `nucleot(27)=afe56, afe56=55.847d0, afe56, af18...`).
     (c) Ajouter la charge de l'isotope, Exemple : `zit(28)=11`.
     (d) Ajouter le nom de l'isotope, Exemple : `'B11 '`. (ces chaînes de 4 caractères sont à cadrer à gauche sauf pour les symboles de 2 caractères qui sont centrés.).

- Pour ajouter une réaction :

  1. Dans la routine `taux_nuc` :
     (a) Compléter la liste de commentaires en y indiquant le symbole de la réaction et son indice, Exemple : `réaction 46 :  B11(p,g)12C`.

[1] CESAM2k prévoit la possibilité d'utiliser des éléments chimiques de l'hydrogène ($Z = 1$) jusqu'au nickel ($Z = 28$). Pour introduire des éléments de masse supérieure, les disposer dans la routine `abon_ini` en appliquant une procédure voisine de celle décrite pour l'ajout d'un isotope.

(b) Initialiser `nom_reac, izz(i,1)..`, Exemple : `nom_react(2)='H2(p,g)He3' ; nuc=H2+p-He3 qt(2)=nuc ; izzt(2,1)=1 ; izzt(2,2)=1.`

(c) Entrer le taux de la réaction[2] `rt(i)` en ln penser à y inclure les 1! ou 2! ou 3! du dénominateur, Exemple : `réaction 8 : C12(p,g)N13(e+ nu)C13 z0=6, z1=1........` en recalculant, au besoin, les coefficients à partir des $S(0)$, $S'(0)$, $S''(0)$ suivant l'équation 4-49 de Lang Astrophysical formulae [1968]

2. Dans la routine tabul_nuc

   (a) Entrer les paramètres du calcul des taux, le nombre de réactions et les correspondances des indices, Exemple : `CASE('ppcno12') ; WRITE(2,6) ; WRITE(*,6)....`

3. Dans le module mod_nuc :

   (a) Introduire par un `INCLUDE` le nom de la routine de la chaîne de réactions, Exemple : `INCLUDE 'ppcno9.f'`

4.2 Ajout d'une mixture initiale

Ainsi qu'il est décrit au § 3.2 (Page 49) il est possible de personnaliser les abondances initiales à l'aide d'un fichier mixture. Quand la mixture initiale est d'intérêt général[3] il est préférable d'ajouter la table des abondances dans la routine abon_ini de la source en lui donnant un nom d'au plus 20 caractères. Il y lieu ensuite de mettre à jour la bibliothèque et de refaire le module exécutable en recompilant le module mod_nuc :

`rempl2k_mod mod_nuc`

[2] C'est pour de raisons historiques que ces calculs sont effectués en ln. Les raisons étaient d'éviter d'une part, les erreurs de troncature et d'autre part, les dépassements de capacité avec la limitation à 10^{38}.

[3] Faire preuve de civisme en communiquant la routine ainsi transformée aux responsables de la mise à jour de CESAM2k.

Chapter 5

Debug

5.1 Debug

La structure de CESAM2k en modules permet une vérification efficace de la syntaxe par le compilateur; le revers de la médaille est la mise au point ou le debug qui nécessitent de recompiler le module auquel appartient la routine dans lequel le bug est recherché. Lors de l'installation il est utile de créer, conjointement à la bibliothèque d'exploitation `libcesam2k.a` une bibliothèque de debug `libcesam2k-dbg.a`. Cette bibliothèque est créée automatiquement lors de l'exécution du script `genere_cesam2k-dbg` du sous-directory `SCRIPTS`. Pour ce faire, utiliser les options de debug du compilateur la compilation; la procédure `compile2k-dbg` du sous-directory `SCRIPTS` en est un exemple. Le temps nécessaire à la construction de la bibliothèque de debug est significativement plus court que celui de la construction de la bibliothèque d'exploitation, facteur 5 à 10 par exemple, facteur qui se retrouve mais dans l'autre sens lors d'une exécution.

La plupart des compilateurs indiquent le nom de la routine où le bug se produit. L'exemple suivant se place dans le cas où un bug a été détecté dans la routine des_m.f du module mod_cesam.f. On introduit des instructions de debug dans la routine incriminée puis on exécute le programme `test_cesam` dans lequel on inclus le module mod_cesam.f [1]. Lors de l'exécution les instructions de débug, prises en compte, permettent de remédier au défaut. Une fois celui-ci corrigé, l'exécution du script `mod_repl` du sous-directory `SCRIPTS` avec l'argument `mod_cesam` :

`mod_repl mod_cesam`

remplacera le module mod_cesam.f dans la bibliothèque et recréra l'exécutable `cesam2k.out`.

```
INCLUDE '../SOURCE/mod_cesam.f'

c******************************************************************

PROGRAM test_cesam
```

[1] Au link, ce module se trouvant dans l'environnement, sera utilisé et non pas celui du sous-directory `SOURCE`.

```
USE mod_cesam

IMPLICIT NONE

CALL cesam

STOP

END PROGRAM test_cesam
```

Appendix A

Liste des fichiers ASCII

A.1 Fichiers pour oscillations

Viennent en tête des fichiers de sortie ASCII pour calcul des oscillations, 4 lignes permettant leur identification et la physique utilisée. Ensuite une ligne indique le nombre et les noms des éléments chimiques utilisés, enfin une ligne regroupe le nombre de couches, le nombre de "global" (13), le nombre de variables, le nombre d'éléments chimiques (redondant), enfin l'indice de la vitesse angulaire (s'il n'en est pas tenu compte l'indice est -1). Exemple :

```
Fichier pour inversion: test-inv.osc
CESAM2k version 0.0.0.0 lagr colloc  1  2 np no diffus, 31 Aout 2003 17h45
Physique utilisée: etat_eff, opa_int_zsx, conv_jmj, ppcno9, NACRE
solaire_gn, lim_atm, hopf, perte_ext, diffm_mp, difft_nu, ctes_94
 10  H1  He3  He4  C12  C13  N14  N15  O16  O17  Si28
      472        13        25        10        -1
 1.989190000000E+33 6.959888774683E+10 3.846010241481E+33 1.723296157963E-02 7.033715921640E-01
 1.601359931639E+00 7.033715881909E-01 2.793954502293E-01-1.234804262016E+02-1.250212463069E+02
 4.650000000000E+03 0.000000000000E+00 0.000000000000E+00
 6.965125316795E+10 1.125188830934E-10 4.712514552821E+03 1.082443642936E+03 3.550000000000E-09
 2.317104979700E-04 3.846010241481E+33 5.992732100399E-03 0.000000000000E+00 1.664711685203E+00
 3.986098681160E-01 1.004603449555E+00 1.630687926937E+08 1.300571860485E+00 2.500897748491E+03
 0.000000000000E+00 1.729836584721E+00 8.429278614488E-01 0.000000000000E+00 0.000000000000E+00

.........................................

 3.983397251906E-01 9.536638658353E-01 2.457927862174E+08 8.289918269170E-01 0.000000000000E+00
 0.000000000000E+00-1.948967682005E+00 3.878822249046E-01 2.023289474023E+02 1.828866051074E+01
 1.000000000000E+00 3.288637435509E-01-5.586998681649E-03 6.642708700530E-03-8.894901969547E-03
 3.750487629847E-01 1.039826643345E-05 6.071493899575E-01 1.637161572818E-05 4.475726236722E-06
 4.374234929779E-03 1.935496963607E-07 7.968745830613E-03 3.970500883875E-04 5.030377050940E-03
```

A.1.1 Fichier pour oscillations adiabatiques

```
glob: variables globales du fichier mon_modele-ad.osc

glob(1)=mstar*msol
glob(2)=rtot*rsol
glob(3)=ltot*lsol
glob(4)=z0
glob(5)=x0
```

```
glob(6)=alpha
glob(7)=X dans ZC
glob(8)=Y dans ZC
glob(9)=d2p
glob(10)=d2ro
glob(11)=age
glob(12)=w_rot initial

var: variables locales utilisées ; nvar=22 pour oscillations adiabatiques

var(1,i)=r*rsol
var(2,i)=log(m/mstar) -1.d38 au centre
var(3,i)=t
var(4,i)=Ptot
var(5,i)=ro
var(6,i)=gradient réel d ln T / d ln P
var(7,i)=l
var(8,i)=kap
var(9,i)=énergie thermo+gravifique
var(10,i)=grand Gamma1
var(11,i)=gradient adiabatique
var(12,i)=delta
var(13,i)=cp
var(14,i)=1 / (mu elec.)
var(15,i)=vaissala, 0 au centre
var(16,i)=vitesse angulaire, radian/sec
var(17,i)=d ln kappa / d ln T
var(18,i)=d ln kappa / d ln ro
var(19,i)=d epsilon(nuc) / d ln T
var(20,i)=d epsilon(nuc) / d ln ro
var(21,i)=Ptot / Pgas
var(22,i)=gradient radiatif

composition chimique

var(22+j,i)=xchim(j)*nucleo(j), j=1,nbelem
```

A.1.2 Fichier pour oscillations non adiabatiques

```
glob: variables globales du fichier mon_modele-nad.osc

var: variables
glob(1)=mstar*msol
glob(2)=rtot*rsol
glob(3)=ltot*lsol
glob(4)=z0
```

```
glob(5)=x0
glob(6)=alpha
glob(7)=X dans ZC
glob(8)=Y dans ZC
glob(9)=d2p
glob(10)=d2ro
glob(11)=age
glob(12)=wrot vitesse de rotation globale
glob(13)=w_rot initial

var: variables locales utilisées ; nvar=44 pour oscillations non adiabatiques

var(1,i)=r*rsol
var(2,i)=log(m/mstar) -1.d38 au centre
var(3,i)=t
var(4,i)=Ptot
var(5,i)=ro
var(6,i)=gradient reel d ln T / d ln P
var(7,i)=l
var(8,i)=kap
var(9,i)=énergie thermo+gravifique
var(10,i)=grand Gamma1
var(11,i)=gradient adiabatique
var(12,i)=delta
var(13,i)=cp
var(14,i)=1 / (mu elec.)
var(15,i)=vaissala, 0 au centre
var(16,i)=vitesse angulaire, radian/sec
var(17,i)=d ln kappa / d ln T
var(18,i)=d ln kappa / d ln ro
var(19,i)=d epsilon(nuc) / d ln T
var(20,i)=d epsilon(nuc) / d ln ro
var(21,i)=Ptot / Pgas
var(22,i)=gradient radiatif
var(23,i)=d Gamma1 / d log P
var(24,i)=d Gamma1 / d log T
var(25,i)=d Gamma1 / dY = d Gamma1 / dZ
var(26,i)=dP / dro (TX)
var(27,i)=dP / dT (roX)
var(28,i)=dP / dX (Tro)
var(29,i)=du / dro (TX)
var(30,i)=du / dT (roX)
var(31,i)=du / dx(Tro)
var(32,i)=énergie interne
var(33,i)=d^2P / dro^2 (TX)
var(34,i)=d^2P / dro dT (X)
var(35,i)=d^2P / dT^2(roX)
```

```
var(36,i)=d^2U / dro^2 (TX)
var(37,i)=d^2U / dro dT (X)
var(38,i)=d^2U / dT^2 (X)
var(39,i)=dK / dX
var(40,i)=d^2K / dT^2
var(41,i)=d epsi / dX
var(42,i)=dX / dR
var(43,i)=J-B
var(44,i)=Edding. facteur

composition chimique

var(44+j,i)=xchim(j)*nucleo(j), j=1,nbelem
```

A.1.3 Fichier pour inversions

```
glob: variables globales du fichier mon_modele-inv.osc

glob(1)=mstar*msol
glob(2)=rtot*rsol
glob(3)=ltot*lsol
glob(4)=z0
glob(5)=x0
glob(6)=alpha
glob(7)=X dans ZC
glob(8)=Y dans ZC
glob(9)=d2p
glob(10)=d2ro
glob(11)=age
glob(12)=wrot vitesse de rotation globale
glob(13)=w_rot initial

var: variables locales utilisées ; nvar=25 pour inversion

var(1,i)=r*rsol
var(2,i)=log(m/mstar) -1.d38 au centre
var(3,i)=t
var(4,i)=Ptot
var(5,i)=ro
var(6,i)=gradient reel d ln T / d ln P
var(7,i)=l
var(8,i)=kap
var(9,i)=énergie thermo+gravifique
var(10,i)=grand Gamma1
var(11,i)=gradient adiabatique
var(12,i)=delta
```

```
var(13,i)=cp
var(14,i)=1 / (mu elec.)
var(15,i)=vaissala, 0 au centre
var(16,i)=vitesse angulaire, radian/sec
var(17,i)=d ln kappa / d ln T
var(18,i)=d ln kappa / d ln ro
var(19,i)=d epsilon(nuc) / d ln T
var(20,i)=d epsilon(nuc) / d ln ro
var(21,i)=Ptot / Pgas
var(22,i)=gradient radiatif
var(23,i)=d Gamma1 / d log P
var(24,i)=d Gamma1 / d log T
var(25,i)=d Gamma1 / dY = d Gamma1 / dZ

composition chimique

var(25+j,i)=xchim(j)*nucleo(j), j=1,nbelem
```

A.2 Fichier pour diagramme HR

Le fichier pour tracé/exploitation du diagramme HR est systématiquement implémenté au cours de chaque exécution de CESAM2k. Il est créé lors de l'initialisation d'un modèle sur la ZAMS ou la PMS. Il est complété en cas de reprise d'un modèle existant.

Pour chaque modèle, sur une première ligne sont reportés : l'âge en 10^6 ans, le nombre d'éléments du vecteur de composition chimique généralisé, le nombre de limites ZR/ZC et, pour chacune d'entre elles, la mention "T" ou "F" s'il s'agit, ou non, du début d'une ZC; suivant le nombre de limites une ou plusieurs lignes sont nécessaires pour indiquer $\log T_{\rm eff}$, $\log L/L_\odot$, $\log R/R_\odot$ et, pour chaque limite, $M_\star/M_\odot$, $(M_\star - m)/M_\odot$, $r/R_\odot$, $r_{\rm ov}/R_\odot$, $r_{\rm ov}$ étant le rayon de la limite étendue par overshooting. Dans les cas particuliers on utilise les conventions suivantes :

- modèle totalement convectif : 1 (seule) limite "F" placée au centre ($r/R_\odot = 0$),

- modèle totalement radiatif : 0 (aucune) limite "F" placée en -100 ($r/R_\odot = -100$).

Viennent ensuite les noms des éléments chimiques et leurs abondances, par unité de masse, au centre et à la surface. S'il n'est pas imposé d'overshoot les valeurs $r_{\rm ov}$ des rayons des limites étendues sont fixées à -100.

Exemple :

```
 0.000000000000000E+00 10  2 F T
 3.749146E+00-1.723498E-01-6.098130E-02 1.000000E+00 9.053737E-01 1.262897E-01
-1.000000E+02 2.969686E-02 6.310311E-01-1.000000E+02
 H1  6.97718E-01 6.97718E-01
He3  8.91487E-05 8.91487E-05
He4  2.82700E-01 2.82700E-01
```

```
C12  3.33842E-03 3.33842E-03
C13  4.02357E-05 4.02357E-05
N14  1.03227E-03 1.03227E-03
N15  4.06226E-06 4.06226E-06
O16  9.39680E-03 9.39680E-03
O17  3.80402E-06 3.80402E-06
Si28 5.67649E-03 5.67649E-03
 1.000000000000000E+01 10  2 F T
 3.750235E+00-1.660591E-01-5.995672E-02 1.000000E+00 9.485921E-01 1.002733E-01
-1.000000E+02 2.900062E-02 6.330032E-01-1.000000E+02
 H1  6.97097E-01 6.97795E-01
He3  6.62006E-05 8.91324E-05
He4  2.83225E-01 2.82627E-01
C12  2.45937E-03 3.33777E-03
C13  4.11117E-04 4.02273E-05
N14  1.66274E-03 1.03208E-03
N15  2.18466E-07 4.06144E-06
O16  9.39741E-03 9.39506E-03
O17  3.87613E-06 3.80329E-06
Si28 5.67687E-03 5.67553E-03

..................................

 4.685000000000000E+03 10  1 T
 3.761741E+00-3.270666E-06 2.093004E-06 1.000000E+00 2.259322E-02 7.173108E-01
-1.000000E+02
 H1  3.22007E-01 7.30094E-01
He3  6.46882E-06 8.19412E-05
He4  6.57151E-01 2.51936E-01
C12  2.05912E-05 3.04480E-03
C13  5.63346E-06 3.64768E-05
N14  5.13761E-03 9.44078E-04
N15  2.25834E-07 3.69734E-06
O16  8.99719E-03 8.61328E-03
O17  7.99800E-04 3.47279E-06
Si28 5.87454E-03 5.24189E-03
```

Appendix B

Installation avec `ifc`

B.1 Création de la bibliothèque

Avec le compilateur[1] `ifc` la Norme F95 est implémentée de façon stricte telle qu'elle a été définie en 1996. Depuis, on s'est aperçu d'une limitation "idiote" de la Norme, interdisant de mettre un tableau `ALLOCATABLE` alors que :

`REAL(kind=dp), INTENT(out), POINTER, DIMENSION(:)`

était autorisé. Alors que, dans certains contextes, `POINTER` et `ALLOCATABLE` sont synonymes. Est paru quelques temps plus tard, un Technical Report qui "corrigeait" en l'autorisant. Les compilateurs de Compaq/Digital (stations Unix, PC Windows) Lahey-Fujitsu (Linux, Windows) ont corrigé mais pas INTEL dans son compilateur gratuit.

Remède : Remplacer `ALLOCATABLE` par `POINTER`, chaque fois que le compilateur manifeste. Par la suite, quand le compilateur INTEL sera à jour tout rentrera dans l'ordre. Actuellement la Norme Fortran 2003 différencie `ALLOCATABLE` et `POINTER`.

Les corrections à apporter sont les suivantes :

1. Routine tabul_nuc.f transformer :

   ```
   REAL (kind=dp), INTENT(out), ALLOCATABLE, DIMENSION(:,:) :: taux_reac
   REAL (kind=dp), INTENT(out), ALLOCATABLE, DIMENSION(:) :: ar,q0, temp, ttemp
   INTEGER, INTENT(out) :: i3al, knot_temp, m_temp, n_temp
   CHARACTER (len=20), INTENT(out), ALLOCATABLE, DIMENSION(:) :: nom_reac
   ```

   en :

   ```
   REAL (kind=dp), POINTER, DIMENSION(:,:) :: taux_reac
   REAL (kind=dp), POINTER, DIMENSION(:) :: ar,q0, temp, ttemp
   INTEGER, INTENT(out) :: i3al, knot_temp, m_temp, n_temp
   CHARACTER (len=20), POINTER, DIMENSION(:) :: nom_reac
   ```

2. Routine tabul_nuc transformer :

[1] Diagnostique, commentaires et remèdes selon B.Pichon.

```
REAL (kind=dp), ALLOCATABLE, SAVE, DIMENSION(:,:) :: decrx,dzetax,taux_reac
REAL (kind=dp), ALLOCATABLE, SAVE, DIMENSION(:) :: ar, datx, dbidx,
1 dc1x, decro, decrt, dfx, df12x, dkx, dnex, dr, dzstarx, ecran,
2 lambda, mz1z2, q0, temp, ttemp, tx, z1z2
REAL (kind=dp), SAVE, DIMENSION(2) :: zeta, dzetat, dzetaro
REAL (kind=dp), SAVE :: cte1, cte2, cte3, cte4, cte5, cte6, cte7,
1 cte8, cte9, cte10
REAL (kind=dp) :: zstar, ne, c1, f12, at, k, bid, bid1, df,
1 e0, eps, beta, dnero, dc1ro, df12t, df12ro, datt, datro, dkt,
2 dkro, dbidt, dbidro, dft, dfro
INTEGER, SAVE :: i3al, knot_temp, l=1, m_temp, n_temp
INTEGER :: i, j
CHARACTER (len=20), ALLOCATABLE, DIMENSION(:) :: nom_reac
```

en :

```
REAL (kind=dp), POINTER, SAVE, DIMENSION(:,:) :: decrx,dzetax,taux_reac
REAL (kind=dp), POINTER, SAVE, DIMENSION(:) :: ar, datx, dbidx,
1 dc1x, decro, decrt, dfx, df12x, dkx, dnex, dr, dzstarx, ecran,
2 lambda, mz1z2, q0, temp, ttemp, tx, z1z2
REAL (kind=dp), SAVE, DIMENSION(2) :: zeta, dzetat, dzetaro
REAL (kind=dp), SAVE :: cte1, cte2, cte3, cte4, cte5, cte6, cte7,
1 cte8, cte9, cte10
REAL (kind=dp) :: zstar, ne, c1, f12, at, k, bid, bid1, df,
1 e0, eps, beta, dnero, dc1ro, df12t, df12ro, datt, datro, dkt,
2 dkro, dbidt, dbidro, dft, dfro
INTEGER, SAVE :: i3al, knot_temp, l=1, m_temp, n_temp
INTEGER :: i, j
CHARACTER (len=20), POINTER, DIMENSION(:) :: nom_reac
```

3. Routine mod_donnees transformer :

```
CHARACTER (len=4), SAVE, PUBLIC, ALLOCATABLE, DIMENSION(:) :: nom_elem
```

en :

```
CHARACTER (len=4), SAVE, PUBLIC, POINTER, DIMENSION(:) :: nom_elem
```

B.2 Installation de PGPLOT

Thierry Corbard, Observatoire de la Côte d'Azur a construit le fichier de configuration `ifc_gcc.conf` de PGPLOT avec LINUX pour le compilateur F90 (gratuit) `ifc` de INTEL.
Pour créer la source de PGPLOT en utilisant ce compilateur :

1. Se placer dans le sous-directory `/usr/local/src/pgplot/sys_linux` de la source de PGPLOT.

(a) Y inclure le fichier de configuration : `ifc_gcc.conf` que l'on trouvera ci-dessous, ou encore dans le sous-directory `SCRIPTS`.

(b) Y créer le sous-directory `ifc_src` : `mkdir ifc_src`.

(c) Inclure, dans ce sous-directory `ifc_src`, le fichier `ifc_src/grgenv.f` que l'on trouvera ci-dessous, ou encore dans le sous-directory `SCRIPTS`.

2. Exécuter :

   `/usr/local/src/pgplot/makemake /usr/local/src/pgplot linux ifc_gcc`

B.3 Fichier ifc_gcc.conf

```
# The intel ifc FORTRAN compiler and Gnu gcc C compiler on an elf-system.
# Thierry Corbard, Observatoire de la Cote d'Azur, June 2003
#-----------------------------------------------------------------------

# Optional: Needed by XWDRIV (/xwindow and /xserve) and
#           X2DRIV (/xdisp and /figdisp).
# The arguments needed by the C compiler to locate X-window include files.

   XINCL="-I/usr/X11R6/include"

# Optional: Needed by XMDRIV (/xmotif).
# The arguments needed by the C compiler to locate Motif, Xt and
# X-window include files.

   MOTIF_INCL="$XINCL"

# Optional: Needed by XADRIV (/xathena).
# The arguments needed by the C compiler to locate Xaw, Xt and
# X-window include files.

   ATHENA_INCL="$XINCL"

# Optional: Needed by TKDRIV (/xtk).
# The arguments needed by the C compiler to locate Tcl, Tk and
# X-window include files.

   TK_INCL="-I/usr/include $XINCL"

# Optional: Needed by RVDRIV (/xrv).
# The arguments needed by the C compiler to locate Rivet, Tcl, Tk and
# X-window include files.

   RV_INCL=""

# Mandatory.
```

```
# The FORTRAN compiler to use.

   FCOMPL="ifc"

# Mandatory.
# The FORTRAN compiler flags to use when compiling the pgplot library.
# (NB. makemake prepends -c to $FFLAGC where needed)

   FFLAGC="-Vaxlib  -static -KPIC -O -cm -w95"

# Mandatory.
# The FORTRAN compiler flags to use when compiling fortran demo programs.
# This may need to include a flag to tell the compiler not to treat
# backslash characters as C-style escape sequences

   FFLAGD="-Vaxlib  -static -O -nbs -cm -w95"

# Mandatory.
# The C compiler to use.

   CCOMPL="gcc"

# Mandatory.
# The C compiler flags to use when compiling the pgplot library.

   CFLAGC="-Wall -fPIC -DPG_PPU -O"

# Mandatory.
# The C compiler flags to use when compiling C demo programs.

   CFLAGD="-Wall -O"

# Optional: Only needed if the cpgplot library is to be compiled.
# The flags to use when running pgbind to create the C pgplot wrapper
# library. (See pgplot/cpg/pgbind.usage)

   PGBIND_FLAGS="bsd"

# Mandatory.
# The library-specification flags to use when linking normal pgplot
# demo programs.

   LIBS="-L/usr/X11R6/lib -lX11 -ldl"

# Optional: Needed by XMDRIV (/xmotif).
# The library-specification flags to use when linking motif
# demo programs.
```

```
   MOTIF_LIBS="-lXm -lXt $LIBS"

# Optional: Needed by XADRIV (/xathena).
# The library-specification flags to use when linking athena
# demo programs.

   ATHENA_LIBS="-lXaw -lXt -lXmu -lXext $LIBS"

# Optional: Needed by TKDRIV (/xtk).
# The library-specification flags to use when linking Tk demo programs.
# Note that you may need to append version numbers to -ltk and -ltcl.

   TK_LIBS="-L/usr/lib -ltk -ltcl $LIBS -ldl"

# Mandatory.
# On systems that have a ranlib utility, put "ranlib" here. On other
# systems put ":" here (Colon is the Bourne-shell do-nothing command).

   RANLIB="ranlib"

# Optional: Needed on systems that support shared libraries.
# The name to give the shared pgplot library.

   SHARED_LIB="libpgplot.so"

# Optional: Needed if SHARED_LIB is set.
# How to create a shared library from a trailing list of object files.

   SHARED_LD="gcc -shared  -o $SHARED_LIB"

# Optional:
# On systems such as Solaris 2.x, that allow specification of the
# libraries that a shared library needs to be linked with when a
# program that uses it is run, this variable should contain the
# library-specification flags used to specify these libraries to
# $SHARED_LD

   SHARED_LIB_LIBS=""

# Optional:
# Compiler name used on Next systems to compile objective-C files.

   MCOMPL=""

# Optional:
# Compiler flags used with MCOMPL when compiling objective-C files.
```

```
   MFLAGC=""

# Optional: (Actually mandatory, but already defined by makemake).
# Where to look for any system-specific versions of the files in
# pgplot/sys. Before evaluating this script, makemake sets SYSDIR to
# /wherever/pgplot/sys_$OS, where $OS is the operating-system name
# given by the second command-line argument of makemake. If the
# present configuration is one of many for this OS, and it needs
# different modifications to files in pgplot/sys than the other
# configurations, then you should create a subdirectory of SYSDIR,
# place the modified files in it and change the following line to
# $SYSDIR="$SYSDIR/subdirectory_name".

   SYSDIR="$SYSDIR/ifc_src"
```

B.4 Fichier grgenv.f

```
C*GRGENV -- get value of PGPLOT environment parameter (NAGWare f95)
C+
      SUBROUTINE GRGENV(NAME, VALUE, L)
C      USE F90_UNIX_ENV, ONLY : GETENV
      CHARACTER*(*) NAME, VALUE
      INTEGER L
C
C Return the value of a PGPLOT environment parameter. In Sun/Convex-UNIX,
C environment parameters are UNIX environment variables; e.g. parameter
C ENVOPT is environment variable PGPLOT_ENVOPT. Translation is not
C recursive and is case-sensitive.
C
C Arguments:
C  NAME   : (input) the name of the parameter to evaluate.
C  VALUE  : receives the value of the parameter, truncated or extended
C           with blanks as necessary. If the parameter is undefined,
C           a blank string is returned.
C  L      : receives the number of characters in VALUE, excluding
C           trailing blanks. If the parameter is undefined, zero is
C           returned.
C--
C 25-Mar-1988
C-----------------------------------------------------------------------
      INTEGER I, LIN, ERRNO
      CHARACTER*32 TEST
C
      TEST = 'PGPLOT_'//NAME
      LIN = INDEX(TEST, ' ')-1
```

```
      ERRNO = 0
C      CALL GETENV(TEST(:LIN), VALUE, ERRNO=ERRNO)
      CALL GETENV(TEST(:LIN), VALUE)
C      IF (ERRNO.NE.0 .OR. VALUE.EQ.' ') THEN
      IF (VALUE.EQ.' ') THEN
          L = 0
      ELSE
          DO 10 I=LEN(VALUE),1,-1
              L = I
              IF (VALUE(I:I).NE.' ') GOTO 20
   10     CONTINUE
          L = 0
   20     CONTINUE
      END IF
      END
```

Bibliography

[1998] Adelberger E., et al., 1998, Rev. Mod. Phys. 70, 4, 1265

[1989] Anders E., Grevesse N., 1989, Geochimica et Cosmochimica Acta 53, 197

[1999] Angulo C., Arnould M., Rayet M., and the NACRE collaboration 1999, Nuclear Physics A 656, 1 and WEb site http://pntpm.ulb.ac.be/Nacre/nacre.htm

[1985] Alécian G., 1985, A&A 145, 275

[1990] Alécian G., Artru M.C., 1990, A&A 234, 323

[1990] Alécian G., Michaud G., Tully J., 1990, ApJ 411, 882

[1969] Arnett W.D., Truran J.W., 1969, ApJ 157, 339

[1969] Burgers J.M., 1969, Flow equations for composite gases, Academic Press, New york and London

[1988] Caughlan G.R., Fowler W.A., 1988, Atomic Data and Nuclear Data Tables 40, 284

[1991] Canuto V.M., Mazitelli I., 1991, ApJ 370, 295

[1996] Canuto V.M., Goldman ???, Mazitelli I., 1996, ApJ 473, 550 **INCOMPLET**

[1988] Christensen-Dalsgaard J., 1988, Computational procedures for GONG solar model project, Astronomisk Institut, Aarhus Universitet

[1968] Clayton D.D. ,1968, Principles of Stellar Evolution and Nucleosynthesis, Mc Graw–Hill

[1987] Conte S.D., de Boor C., 1987, Elementary Numerical Analysis, McGraw-Hill Book Company, third edition

[1968] Cox J.P., Guili R.T., 1968, Principles of Stellar Structure, vol I & II, Gordon & Bretch, New-York

[1989] Cox A.N., Guzik J.A., Kidman R.B., 1989, ApJ 342, 1187

[1978] De Boor C., 1978, A Practical Guide to Splines Springer; third ed. 1985

[1971] Eggleton P., 1971, Mon. Not. Roy. astr. Soc. 151, 351

[1972] Eggleton P., 1972, Mon. Not. Roy. astr. Soc. 156, 361

[1973] Eggleton P.P., Faulkner J., Flannery B.P., 1973, A&A 23, 325

[1975] Fowler W.A., Caughlan G.R., Zimmerman B.A., 1975, Ann. Rev. Astron. Astrophys. 13, 69

[1997] Gabriel M., 1997, A&A 327, 771

[1993] Grevesse N., Noels A., 1993, Cosmic Abundances of the Elements. In: Prantzos N., Vangioni-Flam & Cassé M. (Eds.), Origin and Evolution of the Elements. Cambridge University Press, 14

[1991] Hairer E., Wanner G., 1991, Solving Ordinary Differential Equations II, Springer-Verlag

[2002] Heiter??? et al????, 2002, A&A **INCOMPLET**

[1962] Henrici P., 1962, Discrete Variable Methods in Ordinary Differential Equations, John Wiley & Sons.

[1959] Henyey L.G., Le Levier R., Levee R.D., 1959, ApJ 129, 2

[1965] Henyey L.G., Vardya M.S., Bodenheimer P., 1965, ApJ 142, 841

[1996] Houdek G., Rogl J., 1996, Bull. Astr. Soc. India 24, 317

[1964] Iben I., 1964, ApJ 141, 993

[1975] Iben I., 1975, ApJ 196, 525

[1985] Iben I., MacDonald J., 1985, ApJ 296, 540

[1991] Iglesias C.A., Rogers F.J., 1991, ApJ 371 ,408

[1979] Kaps P., Rentrop P., 1979, Numer. Math. 33, 55

[1968] Kippenhahn R., Weigert A., Hofmeister E., 1968, Methods of Computational Physics 7, 129

[1968] Lang Astrophysical formulae A COMPLETER aussi dans la bibli de CESAM2k

[1991] Kippenhahn R., Weigert A., 1991, Stellar Structure and Evolution, Springer-Verlag

[1983] Maeder A., 1983, A&A 120, 113

[1987] Maeder A., Meynet G., 1987, A&A 182, 243

[1985] Marchouk G., Agochkov, V., 1985, Méthode des Eléments Finis, Mir

[1978] Mihalas D., 1978, Stellar Atmosphere, 2d Ed. Freeman and Cie

[1988] Mihalas D., Däppen W., Hummer D.G., 1988, ApJ 331, 815

[1993] Michaud G., Proffitt C.R., 1993, *Inside the Stars*, ed. A. Baglin & W.W. Weiss (San Francisco: ASP), 246

[1977] Mitler H.E., 1977, ApJ 212, 513

[1994] Morel P., van't Veer C., Provost J., Berthomieu G., Castelli F., Cayrel R., Lebreton Y., 1994, A&A 286, 91

[1997] Morel P., 1997, A&AS 124, 597 (une version incluant les appendices, est disponible sur le WEB : http://www.obs-nice.fr/morel/articles.html)

[1999] Morel P., Pichon B., Provost J., Berthomieu G., 1999, A&A 350, 275

[2002] Morel P., Thévenin F., 2002, A&A 390, 611

[1989] Muchmore D., 1984, ApJ 278, 769

[1986] Paquette C., Pelletier C., Fontaine G., Michaud G., 1986, ApJS 61, 177

[1986] Press W.H., Flannery B.P., Teukolsky S.A., Vetterling W.T., 1986, Numerical Recipes, Cambridge University Press, Cambridge

[1991] Proffit C.R., Michaud G., 1991, ApJ 371, 584

[1994] Quarteroni A., Valli A., 1994, Numerical approximation of partial differential equations, Springer-Verlag

[1967] Richtmyer R.D., Morton K.W., 1967, Finite Differences Methods for Initial Values Problems, 2d. Ed., Intersciences Publishers, New-York

[1981] Schatzman E., Maeder A., Angrand F., Glowinski R., 1981, A&A 96, 1

[1990] Schatzman E., Praderie F., 1990, Les Etoiles, Ed. InterEditions/Editions du CNRS

[1981] Schumaker L., 1981, Splines Functions: Basic Theory, John Whiley

[1979] Stoer J., Bulirsch R., 1979, Introduction to Numerical Analysis, Springer-Verlag

[1970] Strittmatter P.A., Faulkner J., Robertson J.W., Faulkner D.J., 1970, ApJ 161, 369

[1994] Thoul A.A., Bahcall J.N., Loeb A., 1994, ApJ 421, 828

[1980] Trenoguine V., 1980, Analyse Fonctionnelle, Mir

[1969] Wagoner R.V., 1969, ApJS 18, 247

[1980] Winkler K.H., Newman M.J., 1980, ApJ 236, 201

[1991] Zahn J.P., 1991, A&A 252, 179