

Tutoriel NONMAT6

(version 1)

C. Faivre

INSERM UMR 911

Centre de Recherche en Oncologie Biologique
et Oncopharmacologie (CRO2)

Aix-Marseille Université

Ce tutoriel suppose que le lecteur a une certaine familiarité avec le logiciel NONMEM et aussi quelques notions sur la validation des modèles non linéaires à effets mixtes.

1 Qu'est-ce-que NONMEM ?

NONMEM est un logiciel créé en 1977 par L. Sheiner et S. Beal (California University) pour l'analyse et la validation des modèles PK/PD. Il reste toujours le "gold standard" dans l'industrie pharmaceutique et le monde universitaire. Il est commercialisé maintenant par la société ICON Development Solutions.

Depuis sa création NONMEM s'utilise à partir d'une interface "ligne de commande" : dans une fenêtre terminal, on donne en entrée le fichier control-stream avec lequel on désire travailler et on précise le nom du fichier de sortie (le fichier qui contiendra les résultats de NONMEM). Sa puissance vient de son langage de description de modèles très versatile et de sa bibliothèque de modèles. Cependant NONMEM dispose de possibilités graphiques réduites. Si l'on veut des graphiques élaborés il faut utiliser un programme annexe.

2 Qu'est-ce-que NONMAT6 ?

NONMAT6 est une toolbox MATLAB qui permet de piloter NONMEM 6 ou NONMEM 7 à partir de MATLAB tout en profitant de la convivialité, de la puissance de calcul et des possibilités graphiques de MATLAB. La toolbox NONMAT6 a été conçue à l'origine pour NONMEM 6.

2.1 System requirements

1. NONMEM 6 (ou NONMEM 7) et MATLAB doivent être installés.
2. MATLAB R2007b ou ultérieur.

NONMAT6 ne nécessite pas la présence d'autres toolbox de MATLAB.

2.2 Installation de NONMAT6

1. Copiez le dossier NONMAT6 sur votre ordinateur. Ouvrir MATLAB et indiquer le chemin jusqu'au dossier MACROS inclus dans le dossier

NONMAT6 à l'aide du menu *File* \longrightarrow *SetPath*.

2. Tapez la commande `Settings` dans la command window de MATLAB et entrez dans la fenêtre graphique (figure 1) le chemin jusqu'à l'exécutable NONMEM. Normalement ce chemin est

`C:\nmvi\run\nmfe6`

(sur Windows) pour NONMEM 6 si l'installation de NONMEM s'est faite normalement. Pour NONMEM 7.3 (la dernière version de NONMEM lorsque ce document a été mis à jour) le nom du dossier NONMEM est différent suivant la version de Windows (32 ou bien 64 bits). Pour Windows 32 bits le nom du dossier est `nm73g` et pour Windows 64 bits `nm73g64`. Normalement le chemin est donc

`C:\nm73g\run\nmfe73`

pour Windows 32 bits et

`C:\nm73g64\run\nmfe73`

pour Windows 64 bits. Valider en cliquant sur OK.

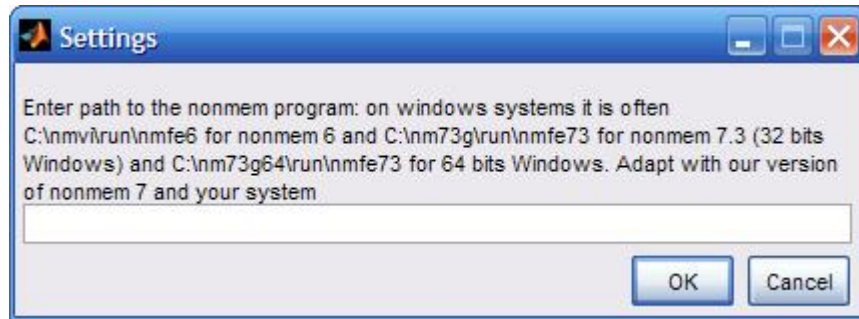


FIG. 1: Commande Settings

3 Utilisation de NONMAT6 (overview)

L'utilisation de NONMAT6 est très aisée :

1. Sélectionner le fichier control-stream que vous voulez analyser avec NONMEM grâce à la commande `SELECT` (à taper dans la command window de MATLAB).

2. Taper RUN

La commande RUN lance (sans sortir de MATLAB) le programme NONMEM pour analyser le modèle décrit dans le control-stream. Les informations essentielles sur le déroulement du run et les valeurs des paramètres de population dans le fichier de sortie de NONMEM s'affichent alors dans la command window de MATLAB. Il existe ensuite d'autres commandes pour tracer des graphiques et valider le modèle. Les diverses commandes de la toolbox sont brièvement décrites ci-dessous. Pour une présentation détaillée voir la section 4.

SELECT	: sélectionne un control-stream
RUN	: analyse le control-stream par NONMEM et affiche les résultats dans command window de MATLAB
DATA	: spaghetti-plot
INITIAL	: choix de valeurs initiales pour les THETA
VAR	: liste des variables de la base de données de NONMAT6
GOF	: graphique goodness of fit
IFIT	: graphique individual fit
ETA	: vérification de la normalité des ETA
ETACOV	: ETA versus une covariable
GETA	: ETA versus ETA
RESIDUALS	: analyse des résidus
IDPLOT	: covariable versus ID
IDCONT	: contribution des individus dans l'estimation des THETA
SE	: précision des estimations pour tous les paramètres
BOOT	: analyse bootstrap
NPDE	: résidus NPDE
VPPC	: graphique VPC
TEST	: test THETA=0
REPORT	: affiche à nouveau le rapport obtenu après RUN
PRINT	: imprime rapport + control-stream dans un fichier
SEARCH	: recherche d'observations dans la base de données
CREATE	: créer les variables de la base de données NONMAT6 dans l'espace de travail MATLAB
GRAPH	: production de graphe
TABLE	: création de table
INIVAR	: re-initialisation de la base de donnée de NONMAT6

3.1 Aide en ligne

Pour toutes les commandes une aide en ligne est disponible. Il suffit de taper (dans la command window de MATLAB) :

```
help nom_de_commande
```

3.2 Base de données de NONMAT6

Certaines commandes de la toolbox utilisent des items du fichier data pour afficher des graphiques. Pour que ceci soit possible NONMAT6 utilise en interne une base de données au même format que le fichier data sauf que les observations sur les doses ont été supprimées. Plus précisément, si l'item MDV est présent dans le champ \$INPUT du control-stream, alors on ne conservera que les observations correspondant à MDV=0. Si MDV n'est pas présent mais que l'item EVID est présent, alors on ne conservera que les observations correspondant à EVID=0. Enfin si ni EVID ni MDV ne sont présents alors les observations correspondant à DV=0 sont supprimées.

Au départ (juste après la commande SELECT) cette base de données est constituée par les variables figurant dans le champ \$INPUT du control-stream qui ne sont pas associées avec DROP ou SKIP. Certaines commandes (comme la commande RUN) modifient cette base de données. A tout moment, la commande VAR permet de voir quelles sont les variables figurant dans la base de données. Ces variables sont des matrices colonnes comme dans le fichier data mais n'existent pas dans l'espace de travail de MATLAB. Il est cependant possible de les créer grâce à la commande CREATE.

3.2.1 Sauvegarde de la base de données

La base de données est sauvegardée lorsque l'on quitte MATLAB. Ainsi si l'on a effectué RUN (par exemple) lors de la précédente session on n'est plus obligé de le faire à nouveau lors de la nouvelle. On peut utiliser directement les autres commandes (GOF, RESIDUALS,...).

3.3 Remarques générales

- Diverses commandes ouvrent des fenêtres graphiques. Dans chaque fenêtre graphique figure dans la barre de titre la chaîne *dossier/nom* où *dossier* est le nom du dossier qui contient le control-stream et *nom* est le nom

du control-stream pour rappeler à l'utilisateur quelle est le modèle qui correspond à cette figure.

- On doit pouvoir écrire dans le fichier control-stream i.e. le fichier control-stream ne doit pas être en lecture seule. Si ce n'est pas le cas modifier ses propriétés dans Windows.

3.4 Visualiser les data avant RUN

Comme on l'a déjà dit, après la commande SELECT la base de donnée de NONMAT6 est constituée des items qui figurent dans le champ \$INPUT du control-stream. On peut alors utiliser certaines commandes avant de faire RUN (pour visualiser les data par exemple), ce sont les commandes :

DATA, VAR, IDPLOT, GRAPH, SEARCH, CREATE, TABLE

Pour pouvoir faire cela il faut cependant un control-stream. Comme c'est uniquement le champ \$INPUT qui détermine la base de donnée au début, il suffit de faire un control-stream le plus simple possible comme par exemple :

```
$PROBLEM
$INPUT item1 item2 ....
$DATA nom
$SUBROUTINE ADVAN1
$PK
K=THETA(1)+ETA(1)
$error
Y=F+EPS(1)
$ESTIMATION METHOD=0
$THETA 1
$OMEGA 1
$SIGMA 1
```

Remarques : les seules commandes qui modifient la base de données de NONMAT6 sont RUN, INIVAR et BOOT (qui ajoute NPDEvalues à la base données).

4 Commandes de la toolbox NONMAT6

1. SELECT

Permet de sélectionner le fichier control-stream avec lequel on désire travailler. Le fichier sélectionné s'ouvre alors dans la fenêtre de l'éditeur de MATLAB.

La commande `SELECT('demo')` ouvre le control-stream sur la theophylline fourni en exemple dans le sous-dossier *Exemple* du dossier NONMAT6.

Remarque importante : la commande `SELECT` n'est à faire qu'une seule fois tant que l'on travaille avec le même control-stream. On peut ensuite modifier ce fichier control-stream si on le désire. Mais pour que les modifications soient prises en compte (au niveau de la base de données) il faut tout d'abord enregistrer et refaire `RUN` qui est la commande (voir plus loin) qui lance l'analyse du fichier control-stream par `NONMEM`.

2. DATA

Spaghetti-plot de DV versus TIME.

La syntaxe `DATA('expr')` affiche un spaghetti-plot de `expr` versus TIME. Ici `expr` peut être une expression MATLAB quelconque faisant intervenir des variables de la base de données de NONMAT6.

Exemple :

```
DATA('log(DV)')
```

3. RUN

Cette commande permet d'analyser le fichier control-stream avec `NONMEM` et affiche (dans la command window de MATLAB) certaines informations sur le modèle (nombre d'individus, AIC, BIC, ...), les valeurs des paramètres de population et reproduit le diagnostic du run tel qu'il figure dans le fichier de sortie de `NONMEM`. Le fichier de sortie de `NONMEM` porte le nom `OUTPUT_name` où `name` est le nom du fichier control-stream et il est situé dans le même dossier que le control-stream. Il est toujours utile de le consulter. Si dans le control-stream figure le champ

```
$NONP ETA
```

(non parametric step) alors les valeurs correspondantes s'afficheront également dans la command window de MATLAB.

Remarques

- (a) Avant de lancer NONMEM, la commande RUN rajoute automatiquement au control-stream initial le champ

`$COVARIANCE PRINT=E`

ainsi que deux autres champs `$TABLE` pour avoir une table de toutes les variables figurant dans le champ `$INPUT` ainsi qu'une table des valeurs des ETA posthoc (si l'option `POSTHOC` a été spécifiée dans `$ESTIMATION`. C'est ce fichier control-stream modifié qui sera en fait analysé par NONMEM. On peut voir tous ces champs en examinant le fichier de sortie de NONMEM car ce dernier reproduit le control-stream qui a été analysé.

- (b) Dans certains cas, on peut vouloir lancer NONMEM sans incorporer au control-stream ces champs supplémentaires. En fait la commande RUN peut être déclinée de trois manières différentes : `RUN(0)`, `RUN(1)` et `RUN(2)` (identique à `RUN`). Les commandes `RUN(1)` et `RUN(2)` fonctionnent uniquement en mode "estimation" i.e. si le champ `$ESTIMATION` figure dans le control-stream. Elles sont aussi inactives si dans le control-stream figure le champ `$SIMULATION` ou bien le champ `$MFSI`.

- i. `RUN(0)` : lance NONMEM tel quel. On ne rajoute rien au control-stream. Toutes les commandes de la toolbox NONMAT6 sont alors inactives.

- ii. `RUN(1)` : lance NONMEM en mode 'partiel' i.e. en ne rajoutant rien au control-stream si ce n'est un champ `$TABLE` pour pouvoir disposer des variables figurant dans le champ `$INPUT`.

- iii. `RUN(2)` : identique à `RUN` décrite plus haut.

- (c) *Modification de la base de données après RUN*

La commande RUN modifie la base de données. Après la commande RUN la base de donnée est constituée des variables suivantes :

- i. Toutes les variables figurant dans le champ `$INPUT` du control-stream, sauf celles associées avec `DROP` ou `SKIP`.

- ii. PRED, RES, WRES.
- iii. Toute variable définie dans le control-stream et figurant dans un champ \$TABLE sans l'option FILE.
- iv. Si RUN ou RUN (2) a été invoqué, on rajoute à cette liste les ETA posthoc : ETA1, ETA2,....

Donnons un exemple. Considérons un fichier control-stream avec deux ETA i.e. ETA (1) et ETA (2) et pour champ \$INPUT :

```
$INPUT ID AMT DV DROP=OCC
```

Supposons également qu'il y ait deux champs \$TABLE sans l'option FILE :

```
$TABLE IPRED IWRES
$TABLE EFF
```

Alors les variables de la base de données de NONMAT6 seront ID, AMT, DV, PRED, RES, WRES, IPRED, IWRES, EFF, ETA1, ETA2.

Remarque : La commande CREATE (voir plus loin) permet effectivement de créer tout ou une partie de ces variables dans l'espace de travail de MATLAB sous le même nom. Chaque variable ainsi créée est une matrice colonne comme dans le fichier data.

4. VAR

La commande VAR affiche les noms de toutes les variables figurant dans la base de données de NONMAT6 (voir section 3.2). Si dans le champ \$INPUT du control-stream certaines variables ont été renommées alors le nom qui apparaîtra dans VAR sera le nouveau nom. Par exemple si l'on a

```
$INPUT ID TIME CONC=DV MDV
```

alors ce sera CONC qui apparaîtra dans VAR et non DV. Pour les variables dans un champ \$TABLE les noms des variables suivent les conventions de NONMEM dans les tables de sortie. NONMEM 6 ne retient que les 4 premières lettres (ce n'est plus le cas avec NONMEM 7). Par exemple si l'on a dans le control-stream

```
$TABLE IPRED
```

alors le nom affiché dans VAR sera IPRE pour NONMEM 6 et IPRED pour NONMEM 7.

5. GOF

Goodness of Fit. Scatter-plot de DV versus PRED et DV versus IPRED si la variable IPRED est dans la base de données (figure 2).

Deux autres syntaxes sont possibles pour cette commande :

- (a) `GOF ('cond')`
- (b) `GOF ('cond1', 'color1', 'cond2', 'color2', ...)`
- (a) La syntaxe `GOF ('cond')` effectue la même chose que `GOF` mais affiche seulement les observations qui vérifient le test défini par la chaîne de caractères `'cond'`. Cette commande peut être utile par exemple si la variable DV combine deux types d'observations (par exemple médicament+métabolite) pour vérifier le fit pour les observations du médicament ou bien du métabolite. On peut mettre dans `'cond'` une expression logique quelconque faisant intervenir des opérateurs logiques et des variables de la base de données de NONMAT6. Les noms des variables doit être celui qui apparaît dans la commande VAR.
- (b) La syntaxe

```
GOF ('cond1', 'color1', 'cond2', 'color2', ...)
```

affiche les observations vérifiant la condition `cond1` avec la couleur `color1`, les observations vérifiant la condition `cond2` avec la couleur `color2` etc... Les couleurs doivent être celles de MATLAB : red, green, blue, cyan, magenta, yellow, black, white.

Exemples :

- (a) `GOF ('ID==1')`
- (b) `GOF ('WRES>5 & TIME>1')`
- (c) `GOF ('TIME<5', 'red', 'TIME>=5', 'blue')`

Remarque :

Utiliser les opérateurs logiques `&`, `|`, `...` au lieu des opérateurs logiques `&&`, `||`, `....`

6. INITIAL

Ouvre une fenêtre graphique (voir figure 3) pour faciliter le choix des valeurs initiales (uniquement pour les THETA) avant de faire RUN. La

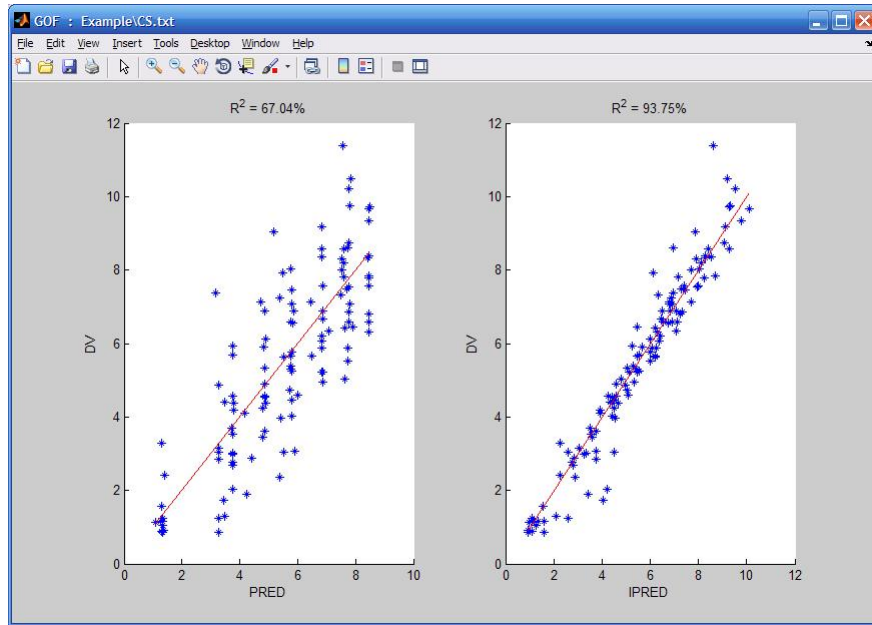


FIG. 2: Commande GOF

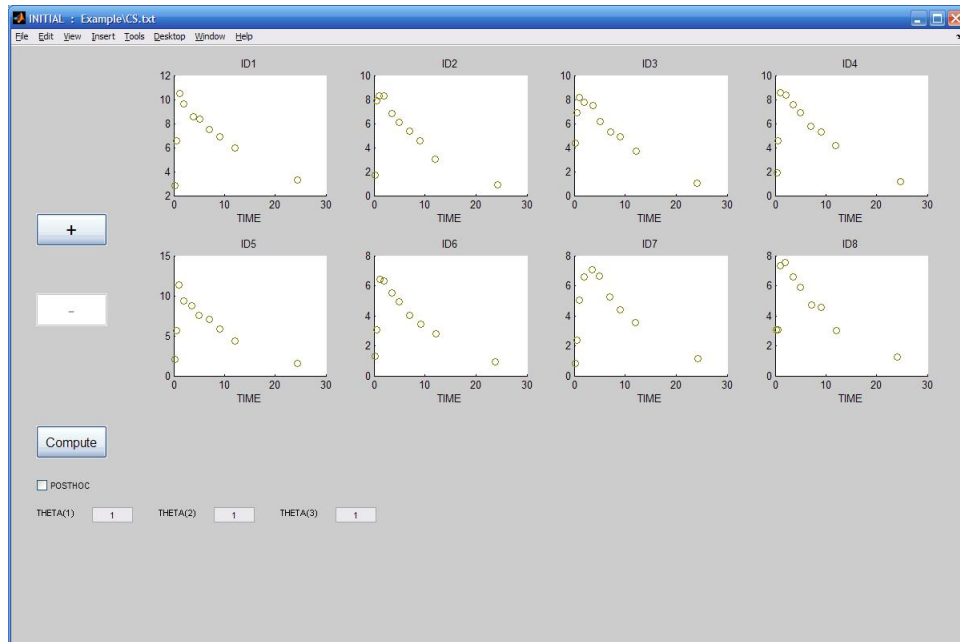


FIG. 3: Commande INITIAL

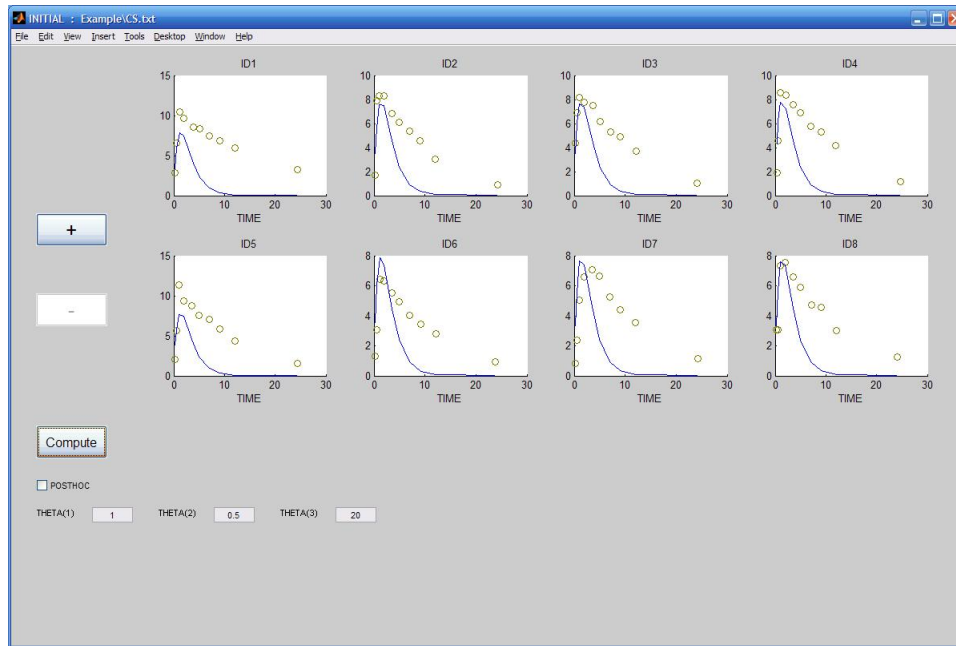


FIG. 4: Commande INITIAL (après Compute)

fenêtre affiche en haut DV versus TIME individu par individu et les boutons + et – permettent de progresser parmi les individus. Le bouton 'compute' calcule alors les valeurs prédites par NONMEM (en prenant pour valeurs des THETA les valeurs de la fenêtre) et les affiche sur les graphes des individus (figure 4). Il s'agit ici des valeurs prédites avec tous les ETA égaux à 0.

Remarque : Si avant de cliquer sur Compute on coche la case posthoc alors seront affichées les valeurs prédites avec les ETA posthoc. Pour calculer les valeurs posthoc NONMEM utilise (en plus des THETA de la fenêtre) les valeurs des champs \$OMEGA et \$SIGMA du control-stream.

Remarque :

Comme pour GOF, la syntaxe INITIAL('cond') effectue la même chose mais n'affiche que les observations qui vérifient la condition 'cond'

Exemple :

INITIAL('TYPE==1')

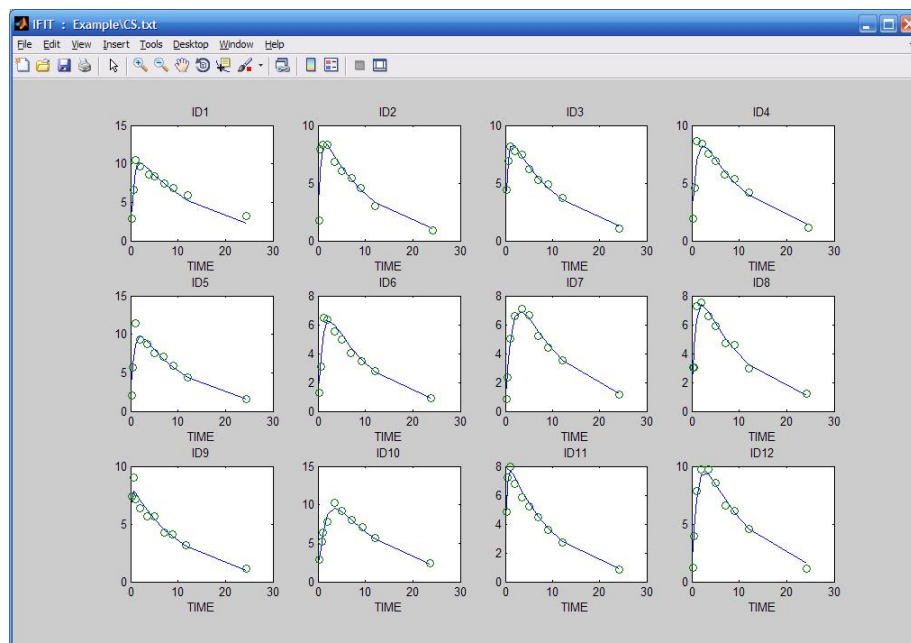


FIG. 5: Commande IFIT

7. IFIT

Individual fit. Affiche une fenêtre graphique (figure 5) pour juger du fit individu par individu. Les boutons + et - permettent de progresser parmi les individus. La courbe est obtenue avec les valeurs de IPRED (si la variable IPRED est dans la base de données). Si certains individus ont été dosés en plusieurs occasions (repérés impérativement par la variable de nom OCC dans le champ \$INPUT du control-stream, alors dans IFIT on aura un graphe suivant chaque occasion. Si OCC est présente dans \$INPUT mais que l'on ne souhaite pas la prendre en compte on peut par exemple modifier OCC en DROP=OCC dans \$INPUT (en effet les variables de \$INPUT précédées de DROP ou SKIP ne sont pas prises en compte) ou bien lui donner un autre nom. Refaire RUN pour prendre en compte les changements.

Remarque :

La syntaxe IFIT ('cond') n'affichera que les observations vérifiant la condition 'cond'.

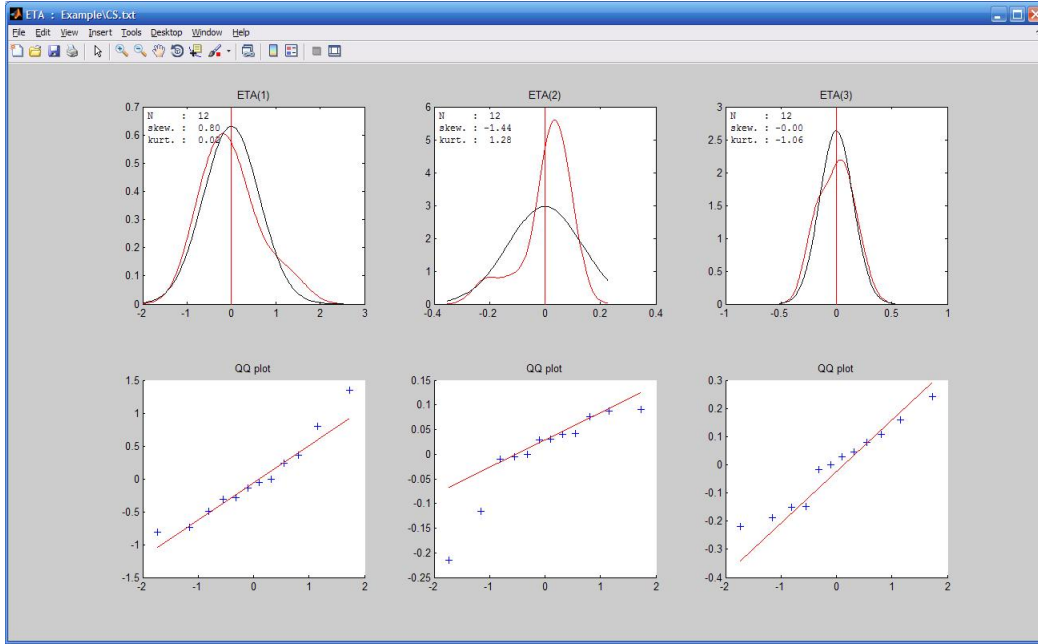


FIG. 6: Commande ETA

8. ETA

Cette commande affiche pour chaque ETA deux graphiques (figure 6) :

- Dans le graphe du haut, la courbe rouge est une estimation de la densité de probabilité des valeurs posthoc de ETA obtenue par la méthode des noyaux. La courbe noire représente la densité de probabilité de la loi normale $N(0, \sigma^2)$ avec σ^2 égal à $\text{OMEGA}(i)$. Donc normalement ces deux courbes doivent être proches.
- Le graphe du bas est un qq-plot des valeurs posthoc de ETA.

Remarques :

- (a) Un test de normalité pour chaque ETA et le shrinkage (ETA-shrinkage) s'affichent dans la command window de MATLAB.
- (b) Les ETA pris en compte dans cette commande sont ceux pour lesquels le OMEGA correspondant n'a pas été fixé à 0 dans le champ $\$ \text{OMEGA}$ du control-stream.

9. ETACOV('expr')

Scatter-plot des estimations posthoc de chaque ETA en fonction de `expr` qui peut être une expression MATLAB quelconque faisant intervenir des variables de la base de données de NONMAT6.

Exemples :

Supposons que les variables WT et HT soient dans la base de données. Alors les syntaxes suivantes sont correctes :

(a) ETACOV('WT')

(b) ETACOV('log(WT)')

(c) ETACOV('WT./HT.^2')

Dans le troisième exemple la présence du point est due au fait que les variables de la base de données sont des matrices colonnes.

10. GETA

Scatter-plot des ETA (posthoc) entre eux. Ces graphes sont utiles pour déceler d'éventuelles corrélations entre deux ETA.

11. RESIDUALS

Scatter-plot de WRES vs TIME et WRES vs PRED. Une autre fenêtre affiche un qq-plot de WRES. Si IWRES et IPRED sont dans la base de données (comme on l'a déjà signalé, les noms qui apparaissent dans VAR sont IWRES et IPRED pour NONMEM 6), la commande affiche également un scatter-plot de IWRES vs TIME et IWRES vs IPRED (figure 7) et un qq-plot de IWRES (figure 8). Rappelons que IWRES doit être défini par la formule :

$$IWRES = \frac{DV - IPRED}{SD}$$

avec SD désignant l'écart-type.

Remarques :

On a également les deux syntaxes suivantes (identiques à celles de la commande GOF) :

(a) RESIDUALS('cond')

(b) RESIDUALS('cond1','color1','cond2','color2',...)

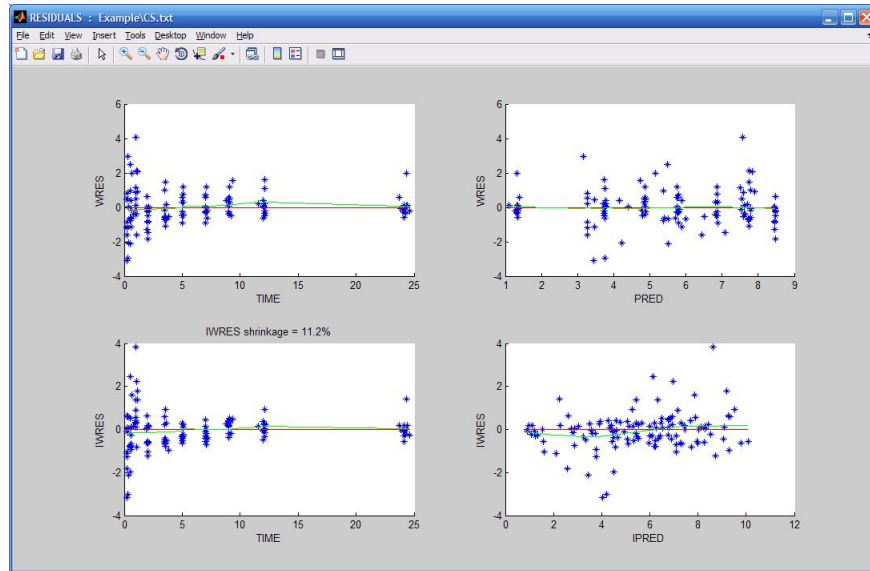


FIG. 7: Commande RESIDUALS

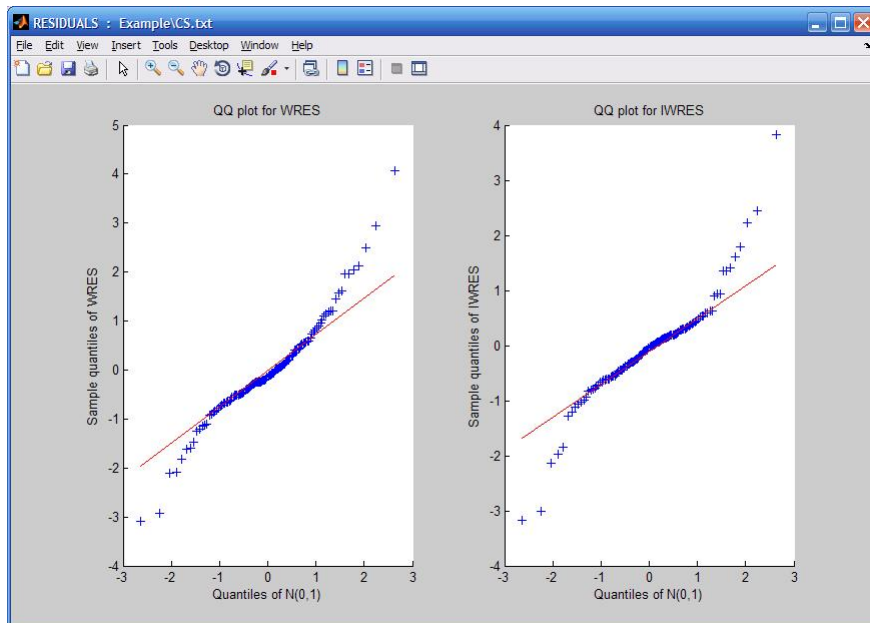


FIG. 8: Commande RESIDUALS (qq-plots)

12. SE

Affiche les standard errors pour tous les paramètres du modèle (THETA, OMEGA, SIGMA). La commande RUN n'affiche que les standard errors pour les THETA.

13. IDPLOT

Index-plot. Scatter-plot de IWRES versus ID si IWRES est dans la base de données, sinon scatter-plot de WRES versus ID.

La syntaxe IDPLOT ('expr') fait la même chose avec les observations de expr qui peut être une expression MATLAB quelconque faisant intervenir des variables de la base de données de NONMAT6.

Exemples :

```
IDPLOT ('log (WRES) ')
```

14. IDCONT

IDCONT est l'acronyme de individual contribution. Cette commande affiche une fenêtre graphique afin de voir l'influence globale de chaque individu dans l'estimation des THETA et ainsi de repérer des individus hors-normes. Le fonctionnement de cette commande est le suivant : pour chaque individu on regarde les THETA estimés par NONMEM quand on supprime l'individu en question du fichier data. On fait cela pour chaque individu. On a donc pour chaque individu une estimation des THETA quand l'individu n'est pas compté dans le fichier data. On fait ensuite une ACP (Analyse en Composantes Principales) de tous les paramètres THETA ainsi obtenus (figure 9). Les résultats de l'ACP sont enregistrés dans un fichier texte de nom PCA_name (où name est le nom du fichier control-stream) dans le même dossier que le control-stream.

Remarques :

Signalons également deux autres syntaxes de cette commande :

(a) IDCONT ('file')

(b) IDCONT (x)

(a) La commande IDCONT ('file') permet d'afficher à nouveau la fenêtre graphique à partir du fichier texte PCA_name. Cette commande est utile quand la commande IDCONT prend beaucoup

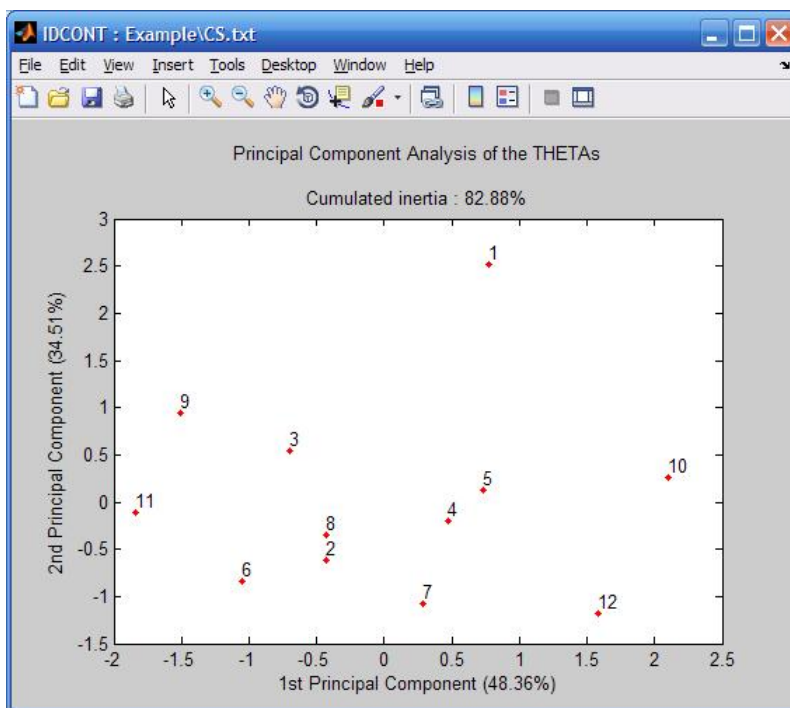


FIG. 9: Commande IDCONT

de temps. Par exemple, on peut l'utiliser pour afficher à nouveau le graphique de l'ACP (après avoir quitté MATLAB par exemple) sans avoir à refaire les calculs.

- (b) Ici x est une sous-liste de $1:N$ où N le nombre d'individus dans le fichier data (1 désigne le premier individu du fichier data, 2 le second etc.). La commande IDCONT est identique à IDCONT(1:N). Quand x est distinct de $1:N$ cette commande ne produit pas de sortie elle enregistre simplement les THETA correspondant à x dans le fichier texte PCA_name. La commande IDCONT(x) permet ainsi de scinder l'exécution de IDCONT en plusieurs fois (si l'exécution de IDCONT est trop longue). Par exemple, la commande IDCONT(1:4) limite l'exécution de la commande IDCONT quand on supprime successivement les quatre premiers individus de la base de données. On renomme alors le fichier PCA_name en PCA14 par exemple (sinon la prochaine exécution de la commande va écraser le contenu antérieur). Puis on invoque IDCONT(x) en sélectionnant une autre partie des individus, par exemple IDCONT(5:9). On renomme le fichier PCA_name en PCA59 par exemple et on lance encore IDCONT(x) etc. On mixe ensuite tous les fichiers ainsi obtenus dans un fichier unique que l'on nomme PCA_name. On peut ensuite faire l'ACP à l'aide de la commande IDCONT('file').

Remarque :

Pour le bon fonctionnement de la commande IDCONT le champ \$INPUT doit précéder le champ \$DATA dans le control-stream. Si ce n'est pas le cas permuter les deux champs dans le control-stream.

15. BOOT

Analyse bootstrap de l'estimation des THETA. Par défaut 200 répliques (fichiers bootstrap) sont effectuées. La commande BOOT(N) effectue N répliques. Les résultats sont affichées dans la command window de MATLAB.

16. NPDE

Test du modèle par les résidus NPDE (Normalised Prediction Distribution of Errors). Par défaut 200 répliques sont effectuées. Si l'on veut N répliques taper NPDE(N). La commande affiche un qq-plot des résidus NPDE ainsi que plusieurs statistiques dans la command window

de MATLAB. La commande NPDE rajoute à la base de données de NONMAT6 la variable NPDEvalues contenant les valeurs des résidus NPDE. Pour la définition des résidus NPDE voir [1],[2].

17. VPPC

VPPC est l'acronyme de Visual Post Predictive Check. Par défaut 100 réplifications sont effectuées. Les différentes syntaxes de cette commande sont les suivantes :

- (a) VPPC
- (b) VPPC(level)
- (c) VPPC(level,rep)
- (d) VPPC(level,rep,Nbin)
- (e) VPPC(level,rep,Nbin,CIlevel)

level : vecteur des niveaux (par défaut [10 50 90])
rep : nombre de datasets simulés (par défaut 100)
Nbin : nombre de bins (par défaut 10)
CIlevel : niveau de confiance (par défaut 95%)

Exemples :

- (a) VPPC
- (b) VPPC(50,200)
- (c) VPPC([10,50],100,10,90)

18. REPORT

Permet d'afficher à nouveau le rapport sur le modèle en cours. Le rapport en question est celui qui apparait dans la command window de MATLAB après la commande RUN. La commande REPORT est surtout utile quand par exemple, après avoir quitté MATLAB on veut visualiser à nouveau le résultat de la commande RUN sans re-exécuter celle-ci surtout si son exécution prend un certain temps.

19. PRINT

Produit un fichier texte (dans le même dossier que le control-stream) de nom REPORT_nom, où nom est le nom du fichier control-stream. Ce fichier est constitué de 2 parties. Dans la première, la sortie de la

commande RUN est reproduite et la deuxième partie est constituée du control-stream. Ce fichier permet de garder une trace de l'exécution d'un control-stream particulier.

20. TEST

Test THETA=0 et intervalle de confiance pour chaque THETA. Par défaut le niveau de chaque intervalle de confiance est 95%. On peut changer le niveau. Par exemple, pour un niveau de confiance de 90% taper TEST(0.90).

21. GRAPH('expr1','expr2')

Scatter-plot de expr1 versus expr2. Ici expr1 et expr2 peuvent être des expressions MATLAB quelconques faisant intervenir des variables de la base de données de NONMAT6.

La syntaxe GRAPH('expr1','expr2','cond') permet de limiter l'affichage aux observations qui vérifient la condition cond

Exemples :

- (a) GRAPH('log(DV)', 'WRES')
- (b) GRAPH('log(DV)', 'log(WRES)')
- (c) GRAPH('log(DV)', 'log(WRES)', 'ID==1')

22. SEARCH('cond')

Localise dans le fichier data associé au control-stream la ou bien les observations vérifiant le test défini par la chaîne de caractères 'cond'. Par exemple, la commande

```
SEARCH('WRES>5 & TIME>1')
```

affiche dans command window de MATLAB les observations (s'il en existe) telles que WRES est supérieur à 5 et TIME supérieur à 1. Pour chaque observation ainsi retenue les valeurs correspondantes de ID et TIME sont affichées. On peut faire en sorte que dans l'affichage apparaissent la valeur d'une ou bien de plusieurs autres variables de la base de données de NONMAT6. Il suffit de préciser leur nom après 'cond'. Par exemple :

```
SEARCH('WRES>5 & TIME>1', 'WRES', 'PRED')
```

affichera pour chacune des observations trouvées la valeur de WRES et PRED en plus de ID et TIME.

23. CREATE

Créer dans l'espace de travail de MATLAB des variables de la base de données de NONMAT6.

Deux syntaxes sont possibles :

(a) CREATE

(b) CREATE ('var1', 'var2', ...)

(a) La syntaxe CREATE permet de créer dans l'espace de travail de MATLAB *toutes* les variables de la base de données de NONMAT6. Les variables ainsi créés dans MATLAB porteront le même nom.

(b) La syntaxe CREATE ('var1', 'var2', ...) permet de ne créer que les variables var1, var2,

Pour voir quelles sont les variables dans la base de données de NONMAT6 utiliser la commande VAR.

Exemple :

```
CREATE ('ETA1', 'ETA2')
```

va créer dans l'espace de travail de MATLAB deux variables appelées ETA1 et ETA2. Chacune de ces variables est à l'origine une matrice colonne comme dans le fichier data.

Remarque :

Comme les valeurs de ETA1 et ETA2 sont les mêmes pour toutes les observations d'un même individu, il peut être utile de ne conserver pour ETA1 et ETA2 qu'une seule valeur pour chaque individu. On peut arriver à cela en précisant avant la commande CREATE une variable de filtrage (en l'occurrence ID ici) grâce à la commande FILTER. Avant CREATE taper FILTER ('ID')

24. TABLE

Enregistre des variables de la base de données de NONMAT6 dans un fichier texte. Le nom de ce fichier texte est TableExport.txt et il est situé dans le même dossier que le control-stream. Comme pour CREATE deux syntaxes sont possibles :

- (a) TABLE
- (b) TABLE('var1','var2',...)
- (a) La syntaxe TABLE enregistre toutes les variables de la base de données de NONMAT6.
- (b) La syntaxe TABLE('var1','var2',...) n'enregistre que les variables var1,var2,....

Remarque :

Comme pour la commande CREATE, on peut utiliser la commande FILTER.

Exemple :

```
TABLE('ID','TIME','DV','WRES')
```

25. FILTER('var')

Permet de changer de variable de filtrage. Ne sert que pour les commandes CREATE et TABLE.

Dans FILTER('var'), 'var' doit être une chaîne de caractère associée au nom d'une variable de la base de données de NONMAT6 ou bien être égale à 'no'. Dans ce cas il n'y a pas de filtrage et les variables ont alors le même format que dans le fichier data.

Exemples :

```
FILTER('ID')
```

```
FILTER('no')
```

26. INIVAR

Permet de ré-initialiser la base de données de NONMAT6. Après la commande INIVAR la base de données est constituée par les items figurant dans le champ \$INPUT du control-stream (sauf ceux associé avec la commande DROP ou SKIP).

Références

- [1] K. Brendel, E. Comets, C. Laffont, and C. Lavielle. Metrics for external model evaluation with an application to the population pharmacokinetics of gliclazide. *Pharmaceutical Research*, 23 :2036–49, 2006.
- [2] E. Comets, K. Brendel, and F. Mentré. Model evaluation in nonlinear mixed effects models, with applications to pharmacokinetics. *Journal de la Société Française de Statistique*, 151 :106–28, 2010.