User guide for npde 2.0

August 2012

Emmanuelle Comets

Karl Brendel [1]***, Thi Huyen Tram Nguyen, France Mentré***

INSERM, UMR 738, Paris, France

Université Paris Diderot, Sorbonne Paris Cité, Paris, France.

npde website: www.npde.biostat.fr

[1]*Karl Brendel was a student during part of this work, supported by a grant from the Institut de recherches internationales Servier, Courbevoie, France, and has now a permanent position in Servier.*

Contents

This is the User's guide for the add-on package npde (version 2.0, August 2012) for the R language. It is distributed in the `inst` directory of the package.

npde computes normalised prediction distribution errors, a metric to evaluate nonlinear mixed-effect models such as those used in population pharmacokinetic or pharmacodynamic studies. Prediction distribution errors were developed under the name prediction discrepancies by Mentré and Escolano [12]. Brendel et al [1] proposed an improved version taking into account repeated observations within each subject, which were called prediction distribution errors.

The program is an add-on package (or library) running under the R software [17]. Please install and launch R to use npde.

In section 2, we describe briefly the method referenced in [1]. Details concerning the context and the methods can be found in this publication. In section 3, we describe how to use the program to compute, plot and test prediction distribution errors. Finally, in section 4, we illustrate the use of the package through two examples (included in the package).

An additional document is distributed along with the package. Using the same examples as in section 4, it contains a table of graphical options and shows many graphs to illustrate how to use these options and produce specific graphs. This companion document, `demo_npde2.0.pdf`, is also located in the `inst` directory of the package.

1 Installation and Legalese

1.1 Download and installation

npde can be obtained at the following URL: `http://www.npde.biostat.fr/`. The website also contains information concerning the updates to npde, the most recent version of this User Guide, and references to some publications describing and using npde. The current version is 2.0.

The program is distributed as an add-on package (library) for R, and is available on the CRAN, from the menu or using the command install.packages() from within R. Superuser privileges may be required for a system-wide installation.

Please consult the *R Installation and Administration* manual (section 6) provided with R (or available from the CRAN) for further details concerning the installation of packages.

Uninstalling: under Windows, please remove the directory npde from the library directory (path RHOME\library). Under Linux, use the remove option (usually requires superuser privileges), assuming the variable $RHOME contains the path to R:

```
sudo R CMD REMOVE -l $RHOME/library/ npde
```

Note that when using the `install.packages()` function to update the package, it is generally not necessary to uninstall prior versions of npde.

Loading the library is done as usual in R by typing:

```
library("npde")
```

in the R command window. As of version 1.1, downloaded after July 27th, 2007, a message will be printed stating the version and date of the library.

1.2 License

1.3 Citing npde

If you use this program in a scientific publication, we would like you to cite the following reference:

Comets E, Brendel K, Mentré F. Computing normalised prediction distribution errors to evaluate nonlinear mixed-effect models: the npde add-on package for R. *Computer Methods and Programs in Biomedicine* 2008, 90:154-66.

A BibTeX entry for LaTeX users is:

```
@Article{Comets08,
author ={Emmanuelle Comets and Karl Brendel and France Mentr{\'e}},
title ={Computing normalised prediction distribution errors to evaluate
nonlinear mixed-effect models: the npde add-on package for {R}},
volume ={90},
pages ={154--66},
journal ={Computer Methods and Programs in Biomedicine},
year =2008 }
```

Additional references are:

Brendel K, Comets E, Laffont CM, Laveille C, Mentré F. Metrics for external model evaluation with an application to the population pharmacokinetics of gliclazide. *Pharmaceutical Research* 2006, 23: 2036–2049.
Comets E, Brendel K, Mentré F. Computing normalised prediction distribution errors to evaluate nonlinear mixed-effect models: the npde add-on package for R. *Computer Methods and Programs in Biomedicine* 2008, 90: 154-66.
Comets E, Brendel K, Mentré F. Model evaluation in nonlinear mixed effect models, with applications to pharmacokinetics. *Journal de la Societe Francaise de Statistique* 2010, 151: 106-28.
Nguyen TH, Comets E, Mentré F. Extension of NPDE for evaluation of nonlinear mixed effect models in presence of data below the quantification limit with applications to HIV dynamic model. *Journal of Pharmacokinetics and Pharmacodynamics* 2012, 39: 499-518.

with the corresponding BibTeX entries:

```
@Article{Brendel06,
author ={Karl Brendel and Emmanuelle Comets and C{\'e}line Laffont and Christian La
title ={Metrics for external model evaluation with an application to the
population pharmacokinetics of gliclazide},
volume ={23},
pages ={2036--49},
journal ={Pharmaceutical Research},
year =2006 }

@Article{Comets10,
author ={Emmanuelle Comets and Karl Brendel and France Mentr{\'e}},
title ={Model evaluation in nonlinear mixed effect models, with applications
to pharmacokinetics},
volume ={151},
pages ={106--28},
journal ={Journal de la Société Française de Statistique},
year =2010 }

@Article{Nguyen2012,
author = {Thi Huyen Tram Nguyen and Emmanuelle Comets and France Mentr{\'e}},
title   ={{Prediction discrepancies (pd) for evaluation of models with data
under limit of quantification}},
volume ={39},
pages ={499--518},
journal ={Journal of Pharmacokinetics and Biopharmaceutics},
year    =2012 }
```

2 Statistical methods

2.1 Models and notations

Let B denote a learning dataset and V a validation dataset. B is used to build a population pharmacokinetic model called M^B.

Let *i* denote the ith individual (*i* = 1,..., N) and *j* the jth measurement in an individual (*j* = 1,..., n_i, where n_i is the number of observations for subject *i*). Let Y_i=$\{y_{i1},...,yin_i\}$ be the n_i-vector of observations observed in individual *i*. Let the function *f* denote the nonlinear structural model. *f* can represent for instance the pharmacokinetic model. The statistical model for the observation y_{ij} in patient *i* at time t_{ij}, is given by:

$$y_{ij} = f(t_{ij},\theta_i)+\varepsilon_{ij} \tag{1}$$

where θ_i is the p-vector of the individual parameters and ε_{ij} is the residual error, which is assumed to be normal, with zero mean. The variance of ε_{ij} may depend on the predicted concentrations $f(t_{ij},\theta_i)$ through a (known) variance model. Let σ denote the vector of unknown parameters of this variance model.

In pharmacokinetic (PK)/pharmacodynamic (PD) studies for instance, it is usually assumed that the variance of the error follows a combined error model:

$$\text{var}(\varepsilon_{ij}) = (\sigma_{\text{inter}}+\sigma_{\text{slope}}\ f(t_{ij},\theta_i))^2 \tag{2}$$

where σ_{inter} and σ_{slope} are two parameters characterising the variance. In this case, $\sigma = (\sigma_{\text{inter}},\sigma_{\text{slope}})'$. This combined variance model covers the case of an homoscedastic variance error model, where $\sigma_{\text{slope}} = 0$, and the case of a constant coefficient of variation error model when $\sigma_{\text{inter}} = 0$. Another parameterisation often found is:

$$\text{var}(\varepsilon_{ij}) = \sigma^2_{\text{inter}} + \sigma^2_{\text{slope}}\ f(t_{ij}, \theta_i)^2 \tag{3}$$

Another usual assumption in population PK/PD analyses is that the distribution of the individual parameters θ_i follows a normal distribution, or a log-normal distribution, as in:

$$\theta_i = h(\mu, X_i)\ e^{\eta_i} \tag{4}$$

where μ is the population vector of the parameters, X_i a vector of covariates, h is a function giving the expected value of the parameters depending on the covariates, and η_i represents the vector of random effects in individual i. η_i usually follows a normal distribution $\mathcal{N}(0,\Omega)$, where Ω is the variance-covariance matrix of the random effects, but other parametric or non-parametric assumptions can be used for the distribution of the random effects, as in the first paper using this method, in the context of non-parametric estimation [13]. Although npde were developed in the context of pharmacokinetic and pharmacodynamic analyses, it is a general approach that can be used to evaluate mixed effect models. The key assumption is to be able to simulate from the expected model, and the npde are designed to test whether simulations and observations correspond.

We denote Ψ the vector of population parameters (also called hyperparameters) estimated using the data in the learning dataset B: $\Psi = (\mu, \text{vec}(\Omega), \sigma)'$, where $\text{vec}(\Omega)$ is the vector of unknown values in Ω. Model M^B is defined by its structure and by the hyperparameters $\hat{\Psi}^B$ estimated from the learning dataset B.

2.2 Computing pde and npde in the absence of BQL data

Evaluation methods compare the predictions obtained by M^B, using the design of V, to the observations in V. V can be the learning dataset B (internal validation) or a different dataset (external validation). The null hypothesis (H_0) is that data in the validation dataset V can be described by model M^B. Prediction distribution errors are a metric designed to test this assumption. In this section, we assume that there is no observation below the limit of quantification (LOQ) in the validation dataset; an extension to BQL data is presented in section 2.3.

2.2.1 Prediction discrepancies (pd)

Let $\text{p}_i(y|\Psi)$ be the whole marginal posterior predictive distribution of an observation y for the individual i predicted by the tested model. p_i is defined as:

$$p_i(y|\Psi) = \int p(y|\theta_i, \Psi) p(\theta_i|\Psi) d\theta_i$$

Let F_{ij} denote the cumulative distribution function (cdf) of the predictive distribution $p_i(y|\Psi)$. The prediction discrepancy is defined as the percentile of an observation in the predictive distribution $p_i(y|\Psi)$, as given by:

$$\text{pd}_{ij} = F_{ij}(y_{ij}) = \int^{y_{ij}} p_i(y|\Psi)dy = \int^{y_{ij}} \int p(y|\theta_i,\Psi)p(\theta_i|\Psi)d\theta_i dy$$

In NLMEM, F_{ij} has no analytical expression and can be approximated by MC simulations. Using the design of the validation dataset V, we simulate K datasets $V^{sim(k)}$ (k=1,...,K) under model M^B. Let $Y_i^{sim(k)}$ denote the vector of simulated observations for the ith subject in the kth simulation.

The prediction discrepancy pd_{ij} for observation y_{ij} can then be computed from the cdf F_{ij}, as:

$$\text{pd}_{ij} = F_{ij}(y_{ij}) \approx \frac{1}{K}\sum_{k=1}^{K} 1_{\{y_{ij}^{sim(k)} < y_{ij}\}} \tag{5}$$

To handle extreme values of the observations (defined as values smaller or larger than all the simulated values $y_{ij}^{sim(k)}$), we further set:

$$\text{pd}_{ij} = \begin{cases} \frac{1}{2K} & \text{if } y_{ij} \le y_{ij}^{sim(k)} \; \forall\, k = 1\ldots K \\ 1-\frac{1}{2K} & \text{if } y_{ij} > y_{ij}^{sim(k)} \; \forall\, k = 1\ldots K \end{cases}$$

Under H_0, if K is large enough, prediction discrepancies (pd) follow $\mathcal{U}(0,1)$ by construction.

Smoothing the distribution: the simulation-based computation described above can lead to ties especially at the extremes, that is, different observations may have the same value of pd (this occurs particularly often if the number of simulations is small, or the model is quite different from the data). To avoid this issue, we have implemented an option to smooth the distribution: instead of using directly the quantile of the observation within the simulated distribution, as in equation 5, we draw the pd randomly between this quantile (say k/K) and the one immediately above ($(k+1)/K$). We do this by adding a sample from a uniform distribution over the interval $\left[0,\frac{1}{K}\right]$ to the value defined by the previous equation:

$$\text{pd}_{ij} = u_{ij} + \frac{1}{K}\sum_{k=1}^{K} 1_{y_{ij}^{sim(k)} < y_{ij}} \tag{6}$$

Again, extreme values of the observations are treated separately:

$$\text{pd}_{ij} \sim \begin{cases} U[0,1/K] & \text{if } y_{ij} \le y_{ij}^{sim(k)} \; \forall\, k = 1\ldots K \\ U[1-1/K,1] & \text{if } y_{ij} > y_{ij}^{sim(k)} \; \forall\, k = 1\ldots K \end{cases}$$

This option can be set by using the `ties=FALSE` argument.

2.2.2 Prediction distribution errors (pde)

When multiple observations are available for one subject, typically in population analyses, the pd are correlated within each subject [12]. To correct for this correlation, we compute the mean $\mathrm{E}(\mathrm{Y}_i)$ and variance $\mathrm{var}(\mathrm{Y}_i)$ of the K simulations. The mean is approximated by:

$$E(\mathrm{Y}_i) \approx \frac{1}{K}\sum_{k=1}^{K} \mathrm{Y}_i^{sim(k)}$$

and the variance-covariance matrix is approximated by:

$$\mathrm{var}(\mathrm{Y}_i) \approx \frac{1}{K}\sum_{k=1}^{K} (\mathrm{Y}_i^{sim(k)} - E(\mathrm{Y}_i))(\mathrm{Y}_i^{sim(k)} - E(\mathrm{Y}_i))'$$

Decorrelation is performed simultaneously for simulated data:

$$\mathrm{Y}_i^{sim(k)*} = \mathrm{var}(\mathrm{Y}_i)^{-1/2}(\mathrm{Y}_i^{sim(k)} - E(\mathrm{Y}_i))$$

and for observed data:

$$\mathrm{Y}_i^{*} = \mathrm{var}(\mathrm{Y}_i)^{-1/2}(\mathrm{Y}_i - E(\mathrm{Y}_i))$$

Since we are looking to obtain 'residuals', different decorrelation options exist to obtain $\mathrm{var}(\mathrm{Y}_i)^{-1/2}$, corresponding to different sets of decorrelated data. In the npde package, we propose 3 options, which will be detailed in the next section (section 2.2.3).

Decorrelated pd are then obtained using the same formula as 5 but with the decorrelated data, and we call the resulting variables prediction distribution errors (pde):

$$\mathrm{pde}_{ij} = F_{ij}^{*}(y_{ij}^{*}) \tag{7}$$

Under H_0, if K is large enough, the distribution of the prediction distribution errors should follow a uniform distribution over the interval [0,1] by construction of the cdf. Normalized prediction distribution errors (npde) can then be obtained using the inverse function of the normal cumulative density function implemented in most software:

$$\mathrm{npde}_{ij} = \Phi^{-1}(\mathrm{pde}_{ij}) \tag{8}$$

By construction, if H_0 is true, npde follow the $\mathcal{N}(0,1)$ distribution and are uncorrelated within an individual. The decorrelation step however does not make the npde truly independent, since this is only valid for Gaussian variables and here the model nonlinearity makes this only approximately true [4].

2.2.3 Decorrelating the data to obtain pde

Decorrelation methods: To calculate the matrix $\mathrm{var}(Y_i)^{-1/2}$ used for decorrelating data, we can use different methods.

The Cholesky decomposition is a standard way to obtain residuals in regression models, and was used in the initial implementation of the npde library. It is computationally simple, numerically stable, and remains the default method. However, as an iterative pivotal algorithm it is sensitive to the ordering of the vector of Y_i. In PK or PD applications, time imposes a natural order on the vector of longitudinal observations which makes this method very relevant, however this may not be as simple for instance when there are multiple responses. The Cholesky decomposition method is also used in the proc MIXED of SAS, to calculate residuals for correlated data. Let C denote the Cholesky root of $\mathrm{var}(Y_i)$ so that

$$C'C = \mathrm{var}(Y_i)$$

Then $\mathrm{var}(Y_i)^{-1/2} = (C')^{-1}$.

Using a Cholesky decomposition is not the only way to define residuals. An alternative which is invariant to re-ordering of the vector of observations is to use the unique square root of the matrix $\mathrm{var}(Y_i)$, obtained using an eigenvalue decomposition. The matrix $\mathrm{var}(Y_i)$ can be factorized as: $\mathrm{var}(Y_i) = Q\Lambda Q^{-1}$, where Q is the square matrix of the same dimension of $\mathrm{var}(Y_i)$, whose i^{th} column is the eigenvector of $\mathrm{var}(Y_i)$ and Λ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues. The square root matrix of $\mathrm{var}(Y_i)$ is then calculated as:

$$\mathrm{var}(Y_i)^{1/2} = Q\Lambda^{1/2}Q^{-1}$$

and this square root matrix is inverted to calculate the matrix $\mathrm{var}(Y_i)^{-1/2}$. This method is currently implemented in MONOLIX 4 and NONMEM 7 to calculate weighted residuals (WRES, IWPRES) and npde. However, when calculating the symmetric square root from the eigen value-eigen vector decomposition, we will essentially be defining principle directions determined by the variance-covariance matrix and thus, the decorrelated observations/residuals are rotated and no longer correspond to the natural ordering.

We have also implemented a third method, combining a Cholesky decomposition with a polar decomposition [10]. Let C denote the Cholesky root of $\mathrm{var}(Y_i)$. Using polar decomposition, the matrix C can be factorized as: $C = UH$, where U is a unitary matrix and H is a positive-semidefinite Hermitian matrix. The square root matrix of $\mathrm{var}(Y_i)$ is then calculated as:

$$\mathrm{var}(Y_i)^{1/2} = U'C$$

This square root matrix is then inversed to calculate the matrix $\mathrm{var}(Y_i)^{-1/2}$.

Choosing a decorrelation method: The Cholesky method was the only option available in the previous version of the library (npde 1.2) and is implemented as the default method in the current version (npde 2.0). Choosing the decorrelation method is done using the `decorr.method=""` option, with the following choices:

decorr.method="cholesky"	Cholesky decomposition (pseudo-inverse obtained by the `chol` function)
decorr.method="inverse"	Inverse (unique inverse, obtained using the `eigen` function)
decorr.method="polar"	polar decomposition (pseudo-inverse obtained by combining the `chol` function with the `svd` function)

Note: The user needs to be aware that sometimes the decorrelation step (regardless of the method chosen to perform it) will induce or mask patterns in the graphs of npde versus time or predictions. When strange patterns are seen, we advise the user to also look at the pd graphs, which do not involve the decorrelation step, to ascertain whether these patterns are really due to model misspecification and not an artefact of the decorrelation step, and/or to test different decorrelation methods.

2.3 Handling BQL data

Choosing the method to handle BQL data: BQL data means that we do not observe y_{ij} directly, but that we know the observation to be below a censoring value. We restrict ourselves to data below the LOQ, although extension to interval-censored data is straightforward. BQL data in the validation dataset can be treated in different ways:

- removed from the dataset: option cens.method = "omit"
- imputed to model predictions: population predictions (option cens.method = "ppred") or individual predictions (option cens.method = "ipred")
  - with the `"ppred"` method, population predictions are computed using the simulated datasets (for observation y_{ij}, the population prediction is $E_k(y_{ij}^{sim(k)})$))
  - with the `"ipred"` method, individual predictions for each observation obtained during the estimation process need to be included in the data file as an additional column
  - pd and npde are computed after replacing observed and simulated data by the imputed values

- imputed to a fixed value: to LOQ (option cens.method = "loq") or to a value chosen by the user (option cens.method = "fixed",loq=LOQ where LOQ is a number)
  - as in the previous method, pd and npde are computed after replacing observed and simulated data by the imputed values

Sometimes the data includes different LOQs (for instance when the dataset pools several studies with different analytical methods). In this case, the program computes the smallest LOQ in the dataset, and this value is used to censor the simulated data (eg, any simulated value in the dataset lower than the LOQ is omitted or replaced using the imputation method chosen).

Note that all methods involve some loss of power, all the more important when the fraction of BQL data is large, and thus conclusions must be made with prudence when using these imputation methods. However, simulations show that the cens.method = "cdf" is the most suitable [16], and that methods imputing directly with a fixed value or with population model predictions have a poor performance.

Imputation of BQL data using the cumulative distribution function (option cens.method = "cdf") [16]
For an observation above LOQ, the pd is computed as described above, as the quantile of the observation within the predicted distribution. For a BQL observation (left-censored observation) y_{ij}^{cens} of the i^th individual at time t_{ij}, we first evaluate its probability of being under LOQ $\Pr(y_{ij}^{cens} \leq \mathrm{LOQ})$ using the predictive distribution predicted from the model:

$$\Pr(y_{ij}^{cens} \leq \mathrm{LOQ}) = F_{ij}(\mathrm{LOQ}) = \frac{1}{K}\sum_{k=1}^{K} 1_{y_{ij}^{sim(k)} \leq \mathrm{LOQ}} \tag{9}$$

(these predicted probabilities are stored and returned in the results, see section 3.5). Since we only know that the actual observation is below LOQ, we propose to compute the pd for a left-censored observation y_{ij}^{cens}, pd_{ij}^{cens}, as a random sample from a uniform distribution over the interval $[0,\Pr(y_{ij}^{cens} \leq \mathrm{LOQ})]$.

To obtain the npde however, we first need to also impute observations which are below LOQ. We transform the imputed pd back to an imputed observation using the inverse function of the predictive distribution function F_{ij}:

$$y_{ij}^{cens(new)} = F_{ij}^{-1}(pd_{ij}^{cens}) \tag{10}$$

Since pd are quantiles, this corresponds to finding the quantile immediately after the imputed pd_{ij} and setting $y_{ij}^{cens(new)}$ to the value in the simulated distribution F_{ij} corresponding to that quantile (see figure 1 for an illustration). The new vector of observations Y_i^{new} now contains both observed values, for non censored data, and imputed values for censored data.

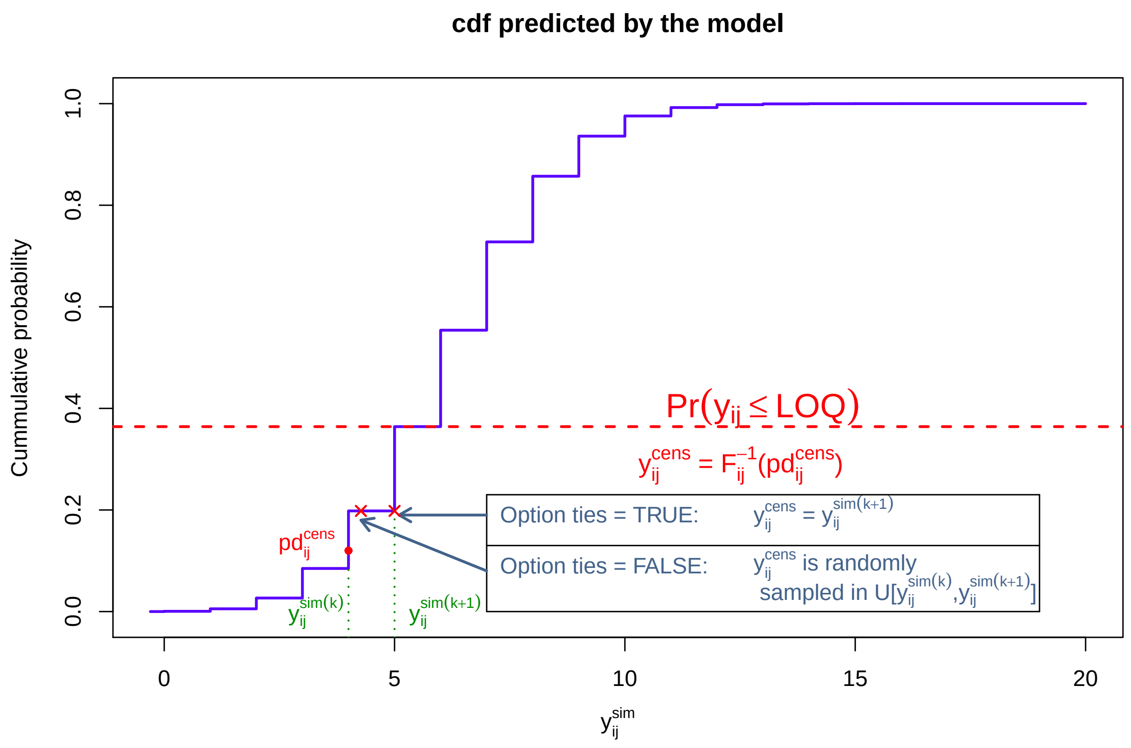

Figure 1 : *Imputation y_{ij}^{cens} from pd_{ij}^{cens} and F_{ij} with the two options* "ties = TRUE" *and* "ties = FALSE"

We cannot simply decorrelate the vector of observed data y_i using the simulations from the model, because the simulated dataset also contains values that would have been censored and treating them as simulated. We therefore propose to impute these censored data to a value between 0 and LOQ using the same method. We impute a $pd_{ij}^{sim(new),k}$ for each $y_{ij}^{sim,k}$ below LOQ in the simulated data and these $y_{ij}^{sim,k}$ are replaced using the same imputation method applied to the observed data.

$$y_{ij}^{sim(new),k} = F_{ij}^{-1}(pd_{ij}^{sim(new),k}) \text{ if } y_{ij}^{sim} \le LOQ \tag{11}$$

As previously, to avoid ties in the pd and npde, we can jitter the imputed pd and y. Figure 1 shows an illustration for both cases:

- method "ties = TRUE": if $F_{ij}(y_{ij}^{sim(k)}) < pd_{ij}^{cens} \le F_{ij}(y_{ij}^{sim(k+1)})$, then $y_{ij}^{cens} = y_{ij}^{sim(k+1)}$
- method "ties = FALSE": if $F_{ij}(y_{ij}^{sim(k)}) < pd_{ij}^{cens} \le F_{ij}(y_{ij}^{sim(k+1)})$, then y_{ij}^{cens} is randomly sampled in a uniform distribution over the interval $[y_{ij}^{sim(k)}, y_{ij}^{sim(k+1)})]$

After the imputation step, a new vector of observations Y_i^{new} and new simulated data $Y_i^{sim_{new}}$ are obtained. The complete data are then decorrelated using the same technique as described above. Note that the matrix $\text{var}(Y_i)$ used to decorrelate is computed using the imputed data, while the predictive distribution functions F_{ij} are computed using the original simulated data before the imputation step.

2.4 Tests

2.4.1 Tests on the distribution of npde

Under the null hypothesis that model M^B describes adequately the data in the validation dataset, the npde follow the $\mathcal{N}(0,1)$ distribution. We report the first three central moments of the distribution of the npde: mean, variance, skewness, as well as the kurtosis, where we define kurtosis as the fourth moment minus 3 so that the kurtosis for $\mathcal{N}(0,1)$ is 0 (sometimes called excess kurtosis). The expected value of these four variables for the expected $\mathcal{N}(0,1)$ are respectively 0, 1, 0 and 0. We also give the standard errors for the mean (SE=$\sigma/\sqrt{N}$) and variance (SE=$\sigma\ \sqrt{2/(N-1)}$).

We use 3 tests to test the assumption that the npde follow the $\mathcal{N}(0,1)$ distribution: (i) a Wilcoxon signed rank test, to test whether the mean is significantly different from 0; (ii) a Fisher test for variance, to test whether the variance is significantly different from 1; (iii) a Shapiro-Wilks test, to test whether the distribution is significantly different from a normal distribution. The package also reports a global test, which consists in considering the 3 tests above with a Bonferroni correction [2]. The p-value for this global test is then reported as the minimum of the 3 p-values multiplied by 3, the number of simultaneous tests (or 1 if this value is larger than 1) [19]. A graphical code is used in the library to highlight significant results, similar to the code used by other statistical functions in R such as lm (see example). The normality test is very powerful, especially with large amount of observations. When the test remains significant even after model refinement, QQ-plots should be used to assess model adequacy in addition to the 3 statistical tests. This is especially useful in large datasets where the sheer amount of data will lead to reject even reasonable models.

2.4.2 Tests for covariate models

In [2], we proposed two approaches to evaluate a model with or without covariates with a validation dataset.

In the first approach, for continuous covariates we can test for correlations between the covariate and npde, using the Spearman correlation test; for categorical covariates we can use Wilcoxon or Kruskal-Wallis tests. If the model and validation data correspond, there should be no relationship between npde and covariates.

In the second approach, we proposed to split the npde according to the values of the covariate, and test within each category that npde follows a $\mathcal{N}(0,1)$ distribution. For categorical covariates, npde are split by categories of the covariate. We proposed to discretise continuous covariates in 3 classes, below first quartile (<Q_1), between first and third quartiles (Q_1–Q_3) and above third quartile (>Q_3). If the model and validation data correspond, there should be no significant departure from $\mathcal{N}(0,1)$ within each category: a test is performed for each category, and the resulting p-values are corrected with a Bonferroni correction.

Both approaches gave similar results in terms of type I error in a simulation study, but the second approach has a slightly larger type I error and a correspondingly slight increase in power [2]. Tests for covariate models will be added shortly to the library.

2.5 Graphs

2.5.1 Diagnostic graphs

Graphs can be used to visualise the shape of the distribution of the npde. Classical plots include quantile-quantile plots (QQ-plots) of the distribution of the npde against the theoretical distribution, as well as histograms and empirical cumulative distributions of the npde and pd. We also find that scatterplots of the npde versus the independent variable, the predicted dependent variables, or covariates, can help pinpoint model deficiencies. Some of these graphs are plotted by default (see section 3.3.2). The package computes for each observation the predicted value as the empirical mean over the k simulations of the simulated predicted distribution (denoted $E_k(y_{ij}^{sim(k)})$), which is reported under the name ypred along with the npde and/or pd.

In the field of population PK/PD, graphs of residuals versus predictions use the values predicted by the model even when the residuals have been decorrelated, as is the case for both spe and npde here. Comparing metrics to their theoretical distributions can be done through QQ-plots or histograms [10]. Examples of these different graphs will be shown in the next section.

Visual Predictive Check (VPC) are now standard diagnostic graphs. In contrast to npde, they do not handle heterogeneity in the design (eg dose regimen) or covariates, so that they are most useful in balanced designs to evaluate models without covariates. However since they are directly obtained from model observations and predictions they illustrate very nicely the shape of the evolution of independent versus dependent variable. VPC are obtained by simulating repeatedly under the model and plotting selected percentiles of the simulated data, comparing them to the same percentiles in the observed data. By default, the VPC produced in the npde package correspond to the median and limits of the 95% prediction interval (eg, the 2.5th, 50th and 97.5th percentile of the data). The observed data can also be plotted over the interval, or omitted for very large datasets.

Finally, when the dataset includes data below the LOQ, a plot of the probability of being BQL can be useful to assess whether the model is able to adequately predict low values, and is available in the npde package.

2.5.2 Prediction intervals

In the current version of the library, prediction bands around selected percentiles, which can be obtained through repeated simulations under the model being tested, can be added to most graphs to highlight departures from predicted values [4]. Prediction intervals build on the idea of simulating from the model to evaluate whether it can reproduce the observed data. For the VPC, a 95% prediction interval on a given percentile (eg the median) can be obtained by computing the median for the K simulated datasets and taking the region where 95% of these K medians lie. This can also be applied to scatterplots of npde or pd, where for each percentile plotted in the graph we can compute a prediction interval of a given size. By default, 95% is used in the npde package, and each prediction interval is plotted as a coloured area (blue for the 2.5 and 97.5th percentile and pink for the median); the corresponding 2.5th, 50th and 97.5th percentiles of the observed data are plotted as lines or points, and should remain within the coloured areas.

A binning algorithm is used for the prediction intervals (the number of bins can be adjusted by the user). Different options are: (1) equal bin sizes on the X-axis; (2) equal bin widths on the X-axis; (3) optimal binning using a clustering algorithm to select the optimal breaks; (4) user-selected breaks. The binning algorithm uses the `Mclust` library [6, 7] for `R`, which implements model-based clustering with an EM-algorithm to select the optimal number of clusters for the variable on the X-axis of the graph.

2.5.3 Graphs for covariate models

Scatterplots of npde versus continuous covariates and box-plots of the npde split by categorical covariates can show the relationship between npde and covariates and help diagnose model misspecifications. Prediction bands are also added to those graphs.

3 The npde package

3.1 Preparation of the input

The library needs two files:

- the file containing the dataset to be evaluated (hereafter named ’observed data’)
- the file containing the simulations (hereafter named ’simulated data’)

The library does not perform the simulations. R, NONMEM [18], MONOLIX [11] or any program of your choice can be used for that purpose.

3.1.1 Observed data

The observed data file must contain at least the following 3 columns:

- id: patient identification
- xobs: independent variable (time, X, ...)
- yobs: dependent variable (DV, concentrations, effects...)

Additional (optional) columns may be given. The program will recognise the following input:

- cens: censoring information (0 for observed data, 1 for censored data)
- mdv: missing data (0 for observed data, 1 for missing data); this information supersedes censoring information, so that an observation with mdv equal to 1 will be treated as missing, not as censored; observations with values "." or NA will also be considered as missing
- ipred: individual model predictions
- covariates

The computation of pd and npde will remove missing observations from the observed dataset reported in the output (see section 3.3).

Other columns may be present but will not be used by the library. The actual order of the columns is unimportant, since the user may specify which column contain the requested information, but the default order is 1. id, 2. xobs, 3. yobs and no MDV column. A file header may be present, and column separators should be one of: blank space(s), tabulation mark, comma (,) or semi-colon (;). Finally, the digit mark should be a dot as in English (eg a number would read 4.5) and not a comma as in French (4,5).

3.1.2 Simulated data

The user must provide a file containing the K simulated datasets stacked one after the other. Within each simulated dataset, the order of the observations must be the same as within the observed dataset. The dimensions of the two datasets must be compatible: if n_{obs} is the number of lines in the observed dataset, the file containing the simulated datasets must have $K \times n_{obs}$ lines.

The simulated data file must contain at least 3 columns, in the following order:

1. id : patient identification

2. xsim: independent variable (time, X, ...)

3. ysim: dependent variable (DV, concentrations, effects...)

Additional columns may be present but will not be used by the library.

The length of the id (resp xobs) column must be equal to the length of the id (resp xobs) column of the observed dataset repeated K times.

An example of how to set up simulations for an example dataset can be found in section 4.1, and examples of a simulated and observed dataset are available in the subdirectory doc/inst of the library.

3.1.3 Number of simulations

Based on the results of simulation studies [2, 4], we recommend to use at least K=1000 but the actual number may depend on the dataset involved, and should be increased when the dataset includes a large number of subjects. This will be investigated in more details in future work on npde. A warning will be issued when K is smaller than 1000 (see examples in section 4).

3.2 Execution

3.2.1 Interactive execution

The interactive mode is called by the function npde():

```
myres<-npde()
```

The user will be prompted to enter successively:

- the name of the file or dataframe containing the observed data

- the columns in which id, xobs, dependent variable yobs, and possibly column with missing data MDV can be found (the default order is 1, 2, 3 and no MDV column)

- the name of the file or dataframe containing the simulated data
- whether results should be saved to disk; if so, the user must also enter
  - the format of the graph (one of: Postscript, JPEG, PNG or PDF)
  - the name of the files: an extension .npde will be added to this name for the file in which numerical results are to be saved (see section 3.3), and an extension depending on the format of the graph will be added to this name for the file in which graphs are to be saved (respectively .eps, .jpeg, .png, .pdf for the formats above). For instance, if the user enters myoutput and requests that the graphs be saved in PDF format, the results file will be named myoutput.npde and the graph files will be myoutput.pdf.
- whether npde should be computed
- whether pd should be computed
- whether a message should be printed as the computation of npde begins in a new subject
- whether the function should return values (see section 3.3.1)

Alternatively, one or both filenames for the observed and simulated data can be replaced by a dataframe if the data has already been loaded in R (see example in the online documentation provided with the package).

3.2.2 Non-interactive execution

In the non-interactive mode, the required information is fed to the function autonpde() instead of answering a series of questions. The minimum input should include the name of the observed data file (for example, theopp.tab) and the name of the simulated data file (for example, simtheopp.tab), as in:

```
autonpde("theopp.tab","simtheopp.tab")
```

A number of options can also be set as arguments, and are given in table I. Note that option `output` has been deprecated (the `invisible()` option is used to suppress the output when it is not affected to an object).

Here is an example of the call to autonpde() with a number of arguments (see example in section 4.1.4 for an illustration):

```
x<-autonpde(namobs="theopp.tab",namsim="simtheopp.tab",iid=1,ix=2,iy=3,
    imdv=0,namsav="output.eps",boolsave=T,type.graph="eps",output=F,verbose=T)
```

Table I: *Options available for the autonpde function.*

Option	Effect	Default value
iid	number of the column with ID information in the observed data file	1
ix	number of the column with the independent variable (X)	2
iy	number of the column with the observed data (Y)	3
imdv	number of the column indicating missing data (MDV)	0
icens	number of the column indicating censoring (CENS)	0
iipred	number of the column with individual predictions (IPRED)	0
detect	if TRUE, the datafile containing the original data is assumed to have a header, and the program will attempt to guess which columns contain ID, X, Y, MDV, CENS and IPRED	FALSE
units	units for the X and Y axis (will be used in plots)	none
boolsave	whether results should be saved to disk	TRUE
namsav	name of the files where results will be saved (without extension)	output
type.graph	graph format (one of: Postscript, JPEG, PNG or PDF, corresponding to extensions .eps, .jpeg, .png and .pdf respectively for the graph file)	eps
calc.npde	whether normalised prediction distribution errors should be computed	TRUE
calc.pd	whether prediction discrepancies should be computed	TRUE
decorr.method	method used to decorrelate	cholesky
cens.method	method used to handle censored data	impute
ties	if FALSE, the distributions of pdand npdeare smoothed to avoid ties	TRUE
verbose	whether a message should be printed as the computation of npde begins in a new subject	FALSE

3.3 Results

Both execution modes will produce the same results. Three types of results are produced by default, but options can be used so that only some of them are created:

1. an R object of class NpdeObject, containing several elements, including the npde and/or pd (see section 3.3.1). With the option output=F the object is not returned.

2. a graph file containing diagnostic plots of the npde (output.eps with the default values; see section 3.3.2). The graph also appears in the graphic window of the current R session. With the option boolsave=F the graph is shown but not saved to a file.

3. a text file with the same name as the graph file and extension .npde containing the following data (output.npde with the default values), organised in columns: id, xobs, ypred, npde, pd With the option boolsave=F, the results are not saved.

3.3.1 Value

By default, the function returns an object of class NpdeObject:

```
myres<-npde()
```

Version 2.0 of the npde package uses the S4 class system, and many functions have been defined to handle the object and produce descriptive summaries and plots (please refer to section 3.5 for more details about S4 classes). As a result, the output is no longer a list that can be manipulated, but a slightly more complicated object. However, for compatibility with the previous version of npde, a `summary` function can be used to produce a list containing the main results:

```
x1<-summary(myres)
```

The object returned by the function contains 5 elements:

1. **data**: a NpdeData object containing the information relative to the observed data

2. **datsim**: a NpdeSimData object containing the information relative to the simulated data

3. **results**: a NpdeRes object containing the results

4. **options**: a list of options used in the computations

5. **prefs**: a list of graphical preferences

The first three elements are S4 objects also, with their own class and own methods, while the last two elements are R lists. More information on how to handle these objects and which methods have been defined for them can be found in 3.5.

3.3.2 Graphs

Four graphs are produced by default:

1. a quantile-quantile plot: plot of the npde versus the corresponding quantiles of a normal distribution
   - the line $y = x$ is also drawn
2. a histogram of the npde
   - the shape of the normal distribution $\mathcal{N}(0,1)$ is also shown
3. a plot of the npde versus the independent variable X
4. a plot of the npde versus ypred
   - for these last two graphs, we plot the lines corresponding to $y = 0$ and to the critical values 5% and 95% (delimiting the 90% confidence interval in which we expect to find the bulk of the npde).

The default graphs now include (approximated) prediction intervals (obtained by simulating from the theoretical $\mathcal{N}(0,1)$ distribution, see section 2.5); for compatibility with the behaviour of npde version 1.2, the option bands=FALSE can be used to suppress plotting the prediction intervals.

These graphs are designed as diagnostics for the npde; a function providing similar graphs for pd is plotpd.

3.4 Errors during execution

Sometimes the function is unable to compute the decorrelated prediction distribution errors for one or more subjects. The following error messages can appear:

```
The computation of the npde has failed for subject xx because the Cholesky
decomposition of the covariance matrix of the simulated data could not be
obtained.
```

or

```
The computation of the npde has failed for subject xx because the covariance
matrix of the simulated data could not be inverted.
```

followed by:

```
This usually means that the covariance matrix is not positive-definite. This can
be caused by simulations widely different from observations (in other words, a
poor model). We suggest to plot a prediction interval from the simulated data
to check whether the simulations are reasonable, and to consider prediction
discrepancies.
Prediction discrepancies will now be computed.
```

In our experience, this usually happens when the model is so ill-conditioned that the matrices involved in the computation of the prediction distribution errors are singular, and mostly happens when the model predicts the data very poorly. A prediction interval (or Visual Predictive Check) can be plotted to check this.

When npde cannot be computed, the program computes automatically pd even if the `calc.pd=F` option was used. The following graphs are plotted using pd instead of npde

1. a quantile-quantile plot: plot of the pd versus the corresponding quantiles of a uniform distribution
    - the line $y = x$ is also drawn
2. a histogram of the pd with the uniform density $\mathcal{U}(0,1)$ overlain
3. a plot of the pd versus the independent variable X
4. a plot of the pd versus ypred
    - for these last two graphs, we plot the lines corresponding to $y = 0$ and to the 5% and 95% critical values (delimiting the 90% confidence interval in which we expect to find the bulk of the pd).

In this case, approximated prediction intervals are not plotted by default, since the approximation (sampling from the standard gaussian distribution) neglects the correlation between the different pd in an individual, and this leads to substantially narrower prediction intervals when the number of data per subject is large. Prediction bands may be added by combining the option `bands=TRUE` with option `approx.pi` set to either `TRUE` for an approximated prediction interval (fast but rough) or `FALSE` for an approximated prediction interval (obtained using the simulated datasets), as in:

```
x<-dist.pred.sim(x)
plot(x,bands=TRUE,approx.pi=TRUE)
```

As seen here, requesting an exact (simulated) prediction interval requires first to compute the distribution of pd in the simulated dataset, using the function `dist.pred.sim` (by default and in the interest of time, this function computes only the distribution of the pd, but if called with the additional argument calc.npde=TRUE the npde in the simulated dataset will also be included in the output, allowing the user to use `approx.pi=TRUE` for graphs including npde).

3.5 Functions in the npde package

npde has been programmed using the S4 classes in R. S4 classes implement Object oriented programming (OOP) in R, allowing to construct modular pieces of code which can be used as black boxes for large systems. Most packages in the base library and many contributed packages use the former class system, called S3. However, S4 classes are a more traditional and complete object oriented system including type checking and multiple dispatching. S4 is implemented in the methods package in base R. More information on S4 classes and Rpackages can be found in tutorials on the Web. I used extensively the following manual [8] (in French).

The object returned by the `npde()` and `autonpde()` functions has the `NpdeObject` class, and both generic and specific methods have been defined for this class:

- print: the print function produces a summary of the object in a nice format
- show: this function is used invisibly by Rwhen the name of the object is typed, and produces a short summary of the object (more details can be obtained by using the alternative `showall()` function
- summary: this function produces a summary of the object, and invisibly returns a list with a number of elements, which provides an alternative way to access elements of the class; the list contains the following elements:

obsdat	the data (a matrix with 3 columns, id=subject id, xobs=observed X, yobs=observed Y, plus if present in the data additional columns containing censored information, mdv, covariates, ...)
id	subject id (first column in obsdat)
x	observed x (second column in obsdat)
y	observed y (third column in obsdat)

npde	the computed npde
pd	the computed prediction discrepancies
ploq	the probability of being BQL for each observation (if computed)
N	number of subjects
nrep	number of simulations used in the computations
ntot.obs	total number of non-missing observations
ypred	predicted Y (the empirical mean of the simulated predicted distribution for each observation ($E_k(y_{ij}^{sim(k)})$))
ycomp	completed vector of observed y (includes the values that were imputed during the computation when BQL data are in the dataset)
ydobs	the decorrelated observed data y_{ij}^{*}
ydsim	the decorrelated simulated data $y_{ij}^{sim(k)*}$
xerr	an integer valued 0 if no error occurred during the computation or a positive number (1 or 2) depending on the error encountered, if an error occurred
options	options (can also be seen by using the `print` function)
prefs	graphical preferences (can also be seen by using the `print` function)

- plot: this produces plots of the different objects
  - when called without any argument, the default four plots are produced
  - an argument plot.type can be used to produce different plots
  - all plots can be tweaked to add titles, change colours,...
- [function: the get function, used to access the value of the slots in an object
- [<-: function: the set function, used to replace the value of the slots in an object
- gof.test: goodness-of-fit tests on the distribution of `npde` (or `pd`)

Examples of calls to these functions are given in the corresponding man pages and in the documentation (section 4.1).

3.6 Options for graphs

The document `test_npde.pdf` has been created to test the npde library and showcase the different graphs. It contains several fits for the datasets shown in section 4, and provides many examples of graphs setting graphical options. There is also a list of the graphical options with their significance. This document is included in the `inst` directory, where the present guide is also located.

An object `x` resulting from a call to npde() or autonpde() contains a slot `prefs` where the graphical preferences are stored as a list. Options can be set on the fly for a given plot, by simply adding them to the call to plot() as an argument (see examples in section 4), and they will then supersede the preferences attached to the object:

```
plot(x,plot.type="data",col="red",main="Raw data")
```

The options can also be modified directly in the object, and they will then apply to the next plots, for instance changing the new default color to red for all plots is done by setting the attribute col in the list:

```
x["prefs"]$col<-"red"
```

Options given on the fly will always supersede the options stored in the `prefs` slot.

Binning: most graphs now have the option of added prediction intervals. These prediction intervals are computed using simulations under the model, and they require a binning algorithm. The influence of the number and position of bins is quite important for the visual assessment of the fit.

Several options are available for binning, and can be set using the `vpc.method` option. Possible options are:

- `equal`: uses quantiles of the data to have about the same number of points in each bin
- `width`: divides the interval into bins of equal size
- `user`: user-defined breakpoints, set in `vpc.breaks` (will automatically be expanded to include the lower and upper value of the data if not provided) (if `vpc.breaks` is not supplied, the `equal` method will be used instead)
- `optimal`: uses the `Mclust` function from the `mclust` library (when available) to provide the optimal clustering; the Mclust method

With all methods, if the number of bins requested is larger than the number of unique values of X, the number of bins will be limited to the number of unique values.

Warning: when using the 'optimal' method, warnings may appear. The optimal number of bins is selected from a range equal to `vpc.bin` ±5, but a message such as:'`In map(out$z) : no assignmen` usually indicates that the number of bins is too large, it is then advised to change the value of `vpc.bin` and start again.

Specifying c(0.01,0.99) with the '`equal`' or '`width`' binning method and vpc.bin=10 will create 2 extreme bands containing 1% of the data on the X-interval, then divide the region within the two bands into the remaining 8 intervals each containing the same number of data; in this case the intervals will all be equal except for the two extreme intervals, the size of which is fixed by the user; complete fine-tuning can be obtained by setting the breaks with the vpc.method="user"

Further options: More details on available graphs and options can be found in the online documentation. Please type:

```
?plot.npde
```

to get started.

If further tweaking is required, any graph can also be recreated with a bit of work using the output from the package. Using the function `summary` will extract the necessary elements from the object, and the user can then use those to produce her or his own graphs.

```
x1<-summary(x)
names(x1)
head(x1$x)
head(x1$npde)
```

4 Examples

4.1 Model evaluation for theophylline PK

The subdirectory doc contains a full working example using npde. We used the theopp.tab dataset provided with NONMEM as an example which most users are already familiar with. This dataset is also available under the name Theoph in the dataset package in R, under a slightly different format. This dataset was provided by a study by Dr. Robert Upton of the kinetics of the anti-asthmatic drug theophylline [18].

The subdirectory doc contains the following files:

theopp.tab	the observed data
simtheopp.tab	the simulated data (with K=100)[1]
fittheop.ctr	the NONMEM control file used for the estimation
simultheop.ctr	the NONMEM control file used for the simulations
runtheo.res	the result file from the NONMEM estimation
theophylline.eps	the graphs
theophylline.npde	the file containing the results
npde_userguide.pdf	the present user guide
vtrue.dat[2]	a file with data simulated under H_0
vfalse.dat[2]	a file with data simulated assuming a bioavailability divided by 2

[1] *We used K=100 to provide a very quick computation of the* npde *and to avoid including a large file in the package, however we recommend using at least K=1000 for the simulations.*

[2] *These datasets were simulated as examples of external validation datasets in [3]*

4.1.1 Data

Theophylline concentrations were measured in 12 patients over a period of 24 hr after a single oral dose of the drug. Each patient received a different dose. The data file has the following structure:

Column number	1	2	3	4	5
Column name	ID	Dose	Time	Conc	Wt
Item meaning	Patient id	Dose	Time	Concentrations	Weight

Doses are given in mg, times in hours and concentrations are reported in mg.L^{-1}.

Figure 2 displays the dataset. The data for the first two patients is given in the appendix (see page 58), to show the format of the data file.

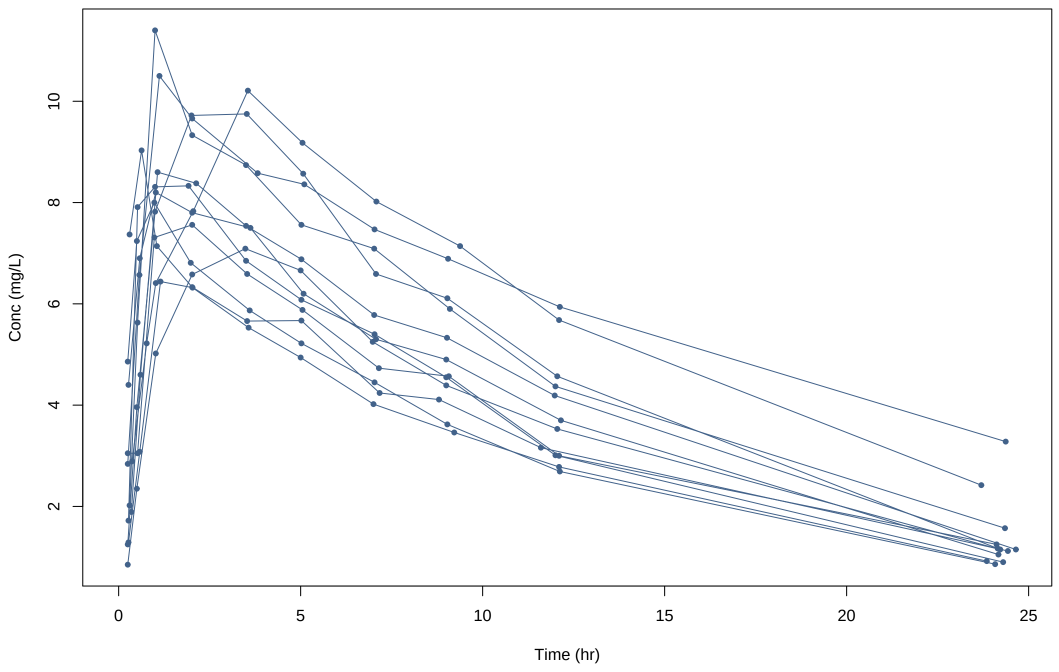

Figure 2 : *Theophylline data.*

4.1.2 Model

The data was analysed with a one-compartment model with first-order absorption and elimination, parameterised in absorption rate constant k$_a$ (units hr^{-1}) volume of distribution V (units L) and elimination rate constant k (units hr^{-1}). Concentrations at time 0 were removed from the dataset. The model did not include covariates. Interindividual variability was modelled using an exponential model for the three pharmacokinetic parameters. A covariance between the parameters k and V was assumed, yielding the following variance-covariance matrix:

$$\Omega = \begin{pmatrix} \omega^2_{k_a} & 0 & 0 \\ 0 & \omega^2_V & omega_{k,V} = \text{cov}(\eta_k, \eta_V) \\ 0 & \text{cov}(\eta_k, \eta_V) & \omega^2_k \end{pmatrix} \tag{12}$$

The residual error model was a combined additive and proportional error model as in equation 2.

These data were analysed with the software NONMEM version 5.1. The ADVAN2 routine was used. The estimation method was the FOCE algorithm with the INTERACTION option. The control file is given in the Appendix (see page 59) and the relevant results in the ouput file runtheo.res are shown on page 60.

The following parameter estimates were obtained:

Population mean		Interindividual variability (CV%)	
k_a (hr^{-1})	1.51	ω_{k_a} (-)	0.67
V (L)	0.46	ω_V (-)	0.12
k ($L.hr^{-1}$)	0.087	ω_k (-)	0.13
σ_{inter} ($mg.L^{-1}$)	0.088	$cor(\eta_k, \eta_V)$ (-)	0.99
σ_{slope} (-)	0.26		

4.1.3 Simulations

The simulations were also performed using NONMEM version 5.1. The control file used for the simulations is given in the Appendix (see page 61). The beginning is identical to the control file used for the analysis (page 60); the initial values in the $THETA, $OMEGA, $SIGMA blocks have been changed to the values estimated with the model, the $ERROR block includes a line to output the simulated data, and the $TABLE block has been changed to output the simulated data in a file.

The number of simulations can be changed with the SUBPROBLEMS options in the $SIMULATION block. Here, we use 100 simulations to compute the npde quickly as an illustration, but larger numbers are more appropriate (we recommend at least 1000 simulations). Simulations were saved in the file simtheopp.tab.

4.1.4 Computing npde

The interactive version of the program was run below. In a first step, the user was prompted to enter all details necessary for the computations (text **in purple** show values entered by the user while text in black is printed by the program):

```
myres<-npde()
Name of the file containing the observed data: theopp.tab
Automatic recognition of columns in the dataset (y/Y) [default=yes] ? n
I'm assuming file theopp.tab has the following structure:
     ID X Y ...
and does not contain a column signaling missing data.
To keep, press ENTER, to change, type any letter: n
     Column with ID information ? 1
     Column with X (eg time) information ? 3
     Column with Y (eg DV) information ? 4
     Column signaling missing data (eg MDV, press ENTER if none) ?
     Column signaling censoring (eg CENS, press ENTER if none) ?
```

Column with individual predictions (eg ipred, press ENTER if none) ?

Columns with covariates (eg WT; enter one at a time, press ENTER if none or when finished) ?

Name of the file containing the simulated data: **simtheopp.tab**

Do you want results and graphs to be saved to files (y/Y) [default=yes] ? **y**

Different formats of graphs are possible:

1. Postscript (extension eps)
2. JPEG (extension jpeg)
3. PNG (extension png)
4. Acrobat PDF (extension pdf)

Which format would you like for the graph (1-4) ? **1**

Name of the file (extension will be added, default=output): **theophylline**

Do you want to compute npde (y/Y) [default=yes] ? **y**

Do you want to compute pd (y/Y) [default=yes] ? **y**

Different decorrelation methods are available:

1. Cholesky decomposition (default)
2. Inverse using diagonalisation (as in Monolix and Nonmem)
3. Cholesky followed by polar decomposition

Which method should be used for the decorrelation (1-3) ? **1**

Method used to handle censored observations:

1. omit: pd will be set to NaN for missing data
2. cdf: pd will be imputed using a random sample from U(0,p_LOQ) where p_LOQ is the probability, according to the model, that a given observation is less than LOQ (default)
3. loq: an observation below the LOQ will be imputed to the LOQ
4. ypred: an observation below the LOQ will be imputed to the population model prediction
5. ipred: an observation below the LOQ will be imputed to the individual model prediction

Which method should be used (1-5) ? **2**

Do you want a message printed as the computation of npde begins in a new subject (y/Y) [default=no] ? **y**

Do you want the function to return an object (y/Y) [default=yes] ? **y**

In the second step, the program computed the normalised prediction distribution errors, plotted the corresponding graphs and performed the statistical tests for npde, then computed the prediction discrepancies (for which no tests are reported). A warning is issued here because the number of simulations is considered too small.

Here we see that the test of the mean (`t-test`) and variance (`Fisher variance test`) don't shown any significant departure from the theoretical values of 0 and 1 respectively, on the other hand, the normality test (`SW test of normality`) indicates a departure from the normal distribution, so that the global test (`Global adjusted p-value`), consisting of a Bonferroni-corrected combination of the three test, also shows a significant departure from the theoretical distribution. However, the results of the tests don't necessarily reflect model adequacy in this case, because of the small number of simulations used.

```
Automatic detection of variables is ON. The program will attempt to detect
both mandatory variables (ID, X, Y) and optional variables (IPRED, MDV, CENS)
when they are not specifically given or when the user-specified names are not
found in the dataset, by looking in the names of the columns (to override this
behaviour, please use argument detect=FALSE in the call to npdeData().
Reading data from file ../data/theopp.tab
These are the first lines of the dataset as read into R. Please check the format
of the data is appropriate, if not, modify the na and/or sep items and retry:
  ID Dose Time  Conc   Wt
1  1 4.02 0.00    NA 79.6
2  1   NA 0.25  2.84   NA
3  1   NA 0.57  6.57   NA
4  1   NA 1.12 10.50   NA
5  1   NA 2.02  9.66   NA
6  1   NA 3.82  8.58   NA

The following NpdeData object was successfully created:
Object of class NpdeData
    longitudinal data
Dataset ../data/theopp.tab
    Structured data: Conc ~ Time | ID
    predictor: Time (hr)
NpdeDataReading data from file ../data/simtheopp.tab
These are the first lines of the dataset as read into R. Please check the format
of the data is appropriate, if not, modify the na and/or sep items and retry:
  ID xsim      ysim
1  1 0.00 -0.090212
2  1 0.25  2.289200
3  1 0.57  4.227900
4  1 1.12  5.497900
5  1 2.02  7.917300
```

```
6  1 3.82  5.394300
There are rows with MDV=1 in the original dataset, the corresponding rows
will be removed from the simulated dataset.

Warning: the number of simulations is 100 which may be too small.
We advise performing at least 1000 simulations to compute npde.
Computing the npde for subject  1
Computing the npde for subject  2
Computing the npde for subject  3
Computing the npde for subject  4
Computing the npde for subject  5
Computing the npde for subject  6
Computing the npde for subject  7
Computing the npde for subject  8
Computing the npde for subject  9
Computing the npde for subject  10
Computing the npde for subject  11
Computing the npde for subject  12
---------------------------------------------
Distribution of npde :
      nb of obs: 120
           mean= 0.0668   (SE= 0.095 )
       variance= 1.074   (SE= 0.14 )
       skewness= 0.511
       kurtosis= 0.2912
---------------------------------------------


Statistical tests
  t-test                     : 0.481
  Fisher variance test       : 0.55
  SW test of normality       : 0.00273 **
Global adjusted p-value      : 0.00818 **
---
Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
---------------------------------------------
Saving results in file ../results/theophylline.npde
Saving graphs in file ../results/theophylline.eps
```

```
Selected plot type: default
Plotting QQ-plot of the distribution
Plotting histogram of the distribution
Plotting scatterplot versus X
Plotting scatterplot versus predictions
```

Alternatively, the first step can be run non-interactively, with the following command:

```
myres<-autonpde("theopp.tab","simtheopp.tab",1,3,4,namsav="theophylline",
verbose=TRUE)
```

4.1.5 Graphs

The graphs in figure 3 are plotted in a window, and saved to a file (unless boolsave=F). The

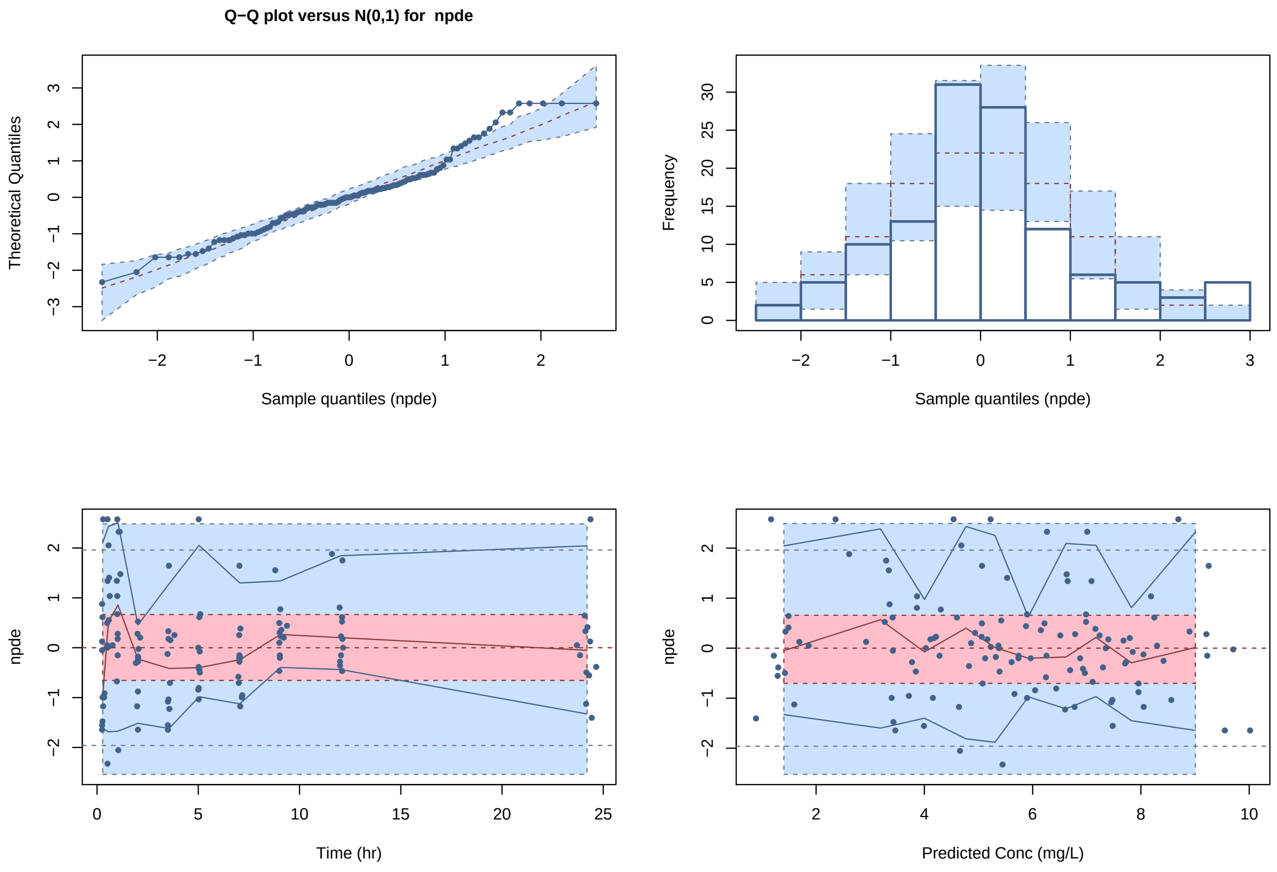

Figure 3 : *Graphs plotted by the* npde() *or* autonpde() *functions.*

quantile-quantile plot and the histogram show a group of values corresponding to npde = 2.33, corresponding to predicted distribution errors set at 0.99, and the prediction bands shows the corresponding departure graphically on the two upper graphs. This indicates observations larger than all the 100 corresponding simulated values. This often happens when K is small as is the case in this example (K=100), and can explain the departure from normality seen in the tests. However, even increasing the number of simulations to 1000 or 2000 does not in this example yield a non-significant test, meaning the model does not describe the data adequately (results not shown).

In the scatterplots (lower two graphs), the pink area is the prediction interval for the median, while the blue areas shows the prediction areas for the boundaries of the 95% prediction intervals. The prediction bands are very large because of the small number of simulations so that model misspecification is not so obvious. By default, the binning on the X-axis uses bins of equal size (number of observations), and the mean of the X values in the bin is used to plot the bin, which explains why the bins do not extend to the largest X-value especially in the lower right plot, but this default behaviour can be tuned in the options.

Figure 4 shows the VPC, where the prediction bands are again very large because of the low number of simulations.

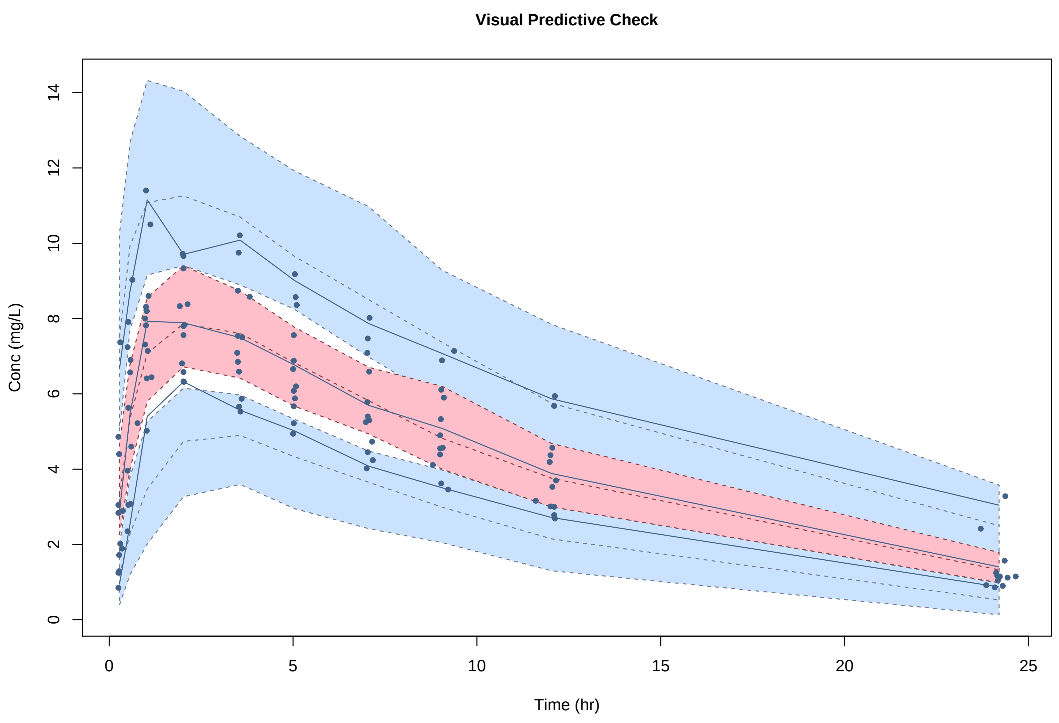

Figure 4 : *VPC of the theophylline data, with 100 simulated datasets.*

4.1.6 Tests

A method gof.test() is available for objects of class `NpdeObject`. When applied to the object resulting from a call to npde() or autonpde(), it will produce the same results as previously:

```
---------------------------------------------
Distribution of npde :
      nb of obs: 120
           mean= 0.0668   (SE= 0.095 )
       variance= 1.074   (SE= 0.14 )
       skewness= 0.511
       kurtosis= 0.2912
---------------------------------------------

Statistical tests
  t-test                    : 0.481
  Fisher variance test      : 0.55
  SW test of normality      : 0.00273 **
Global adjusted p-value     : 0.00818 **
---
Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
---------------------------------------------
```

Here, the four p-values are redirected to the R object y. With the option which="pd", the user can select tests for the pd instead of the npde, but the tests will only be strictly valid when there is only one observation per subject.

A gof.test() method has also been defined for numeric vectors, in which case it will compute the first moments and perform the four tests. The statistical diagnostics can therefore be regenerated easily without running the computation all over again, provided the results have been saved. In the example above, the npde were saved to a file named theophylline.npde. The following code reads the results from this file and computes the same tests as above:

```
dat<-read.table("theophylline.npde",header=T)
y<-gof.test(dat$npde)
```

4.2 Model evaluation for viral loads in HIV (with BQL data)

In this example, we use simulated datasets based on a real study of viral load in HIV patients. The real data and model were obtained in a phase II clinical trial supported by the French Agency for AIDS Research, the COPHAR 3 - ANRS 134 trial [9]. The simulated datasets available in the library were generated in a simulation study designed to evaluate the new method proposed to handle BQL data [16]. Data was simulated using a simple bi-exponential HIV dynamic model describing the two-phase decline of viral load during anti-retroviral treatment; the dataset was then censored at different LOQ levels (LOQ=20 or 50 copies/mL) to generate two datasets containing two different proportions of BQL data.

The data subdirectory contains the following files:

virload.tab	simulated validation dataset without BQL observations
virload20.tab	simulated validation dataset with LOQ=20 copies/mL
virload50.tab	simulated validation dataset with LOQ=50 copies/mL
simvirload.tab	simulated Monte Carlo samples (with K=1000)

The actual proportion of BQL data in the real dataset was 48%, while the proportion of BQL data in the censored datasets is respectively 26 and 44% for virload20.tab and virload50.tab.

4.2.1 Data

Each dataset contains 50 patients having 6 sampling times over a treatment period of 24 weeks (day 0, 28, 56, 84, 112, 168). The data file has the following structure:

Column number	1	2	3	4	5
Column name	ID	Time	Log_VL	cens	ipred
Item meaning	ID	Time	$\log_{10}$(viral load)	Censored or not	individual predictions

Time is in days and viral loads are reported in $\log_{10}$ of the values measured in copies/mL.

4.2.2 Model and parameters for simulation

The model used in this example is a bi-exponential model which was proposed by Ding and colleagues [5], where the model f used to describe the evolution of $\log_{10}$(viral load) is given by:

$$f(t_{ij}, \theta_i) = \log_{10}(P_{1i} e^{-\lambda_{1i} t_{ij}} + P_{2i} e^{-\lambda_{2i} t_{ij}}) \tag{13}$$

This model contains four individual parameters θ_i: P_{1i}, P_{2i} are the baseline values of viral load and the λ_{1i}, λ_{2i} represent the biphasic viral decline rates. These parameters are positive and assumed to follow a log-normal distribution with fixed effects $\mu = (P_1, P_2, \lambda_1, \lambda_2)$. A correlation between the random effects of P_1, P_2 and a constant error model were found on real data. The values used for simulation are inspired by the parameters obtained when analysing real data collected in the COPHAR 3 - ANRS 134 trials [9], and are given in the following table:

Population mean		Interindividual variability (CV%)	
P_1 (copies/mL)	25000	ω_{P_1} (-)	2.1
P_2 (copies/mL)	250	ω_{P_1} (-)	1.4
λ_1 (day^{-1})	0.2	ω_{λ_1} (-)	0.3
λ_2 (day^{-1})	0.02	ω_{λ_2} (-)	0.3
σ (log$_{10}$(copies/mL))	0.14	cor(η_{P_1}, η_{P_2}) (-)	0.8

4.2.3 Computing npde in the presence of BQL data

The following code was used to compute the npde for the 3 datasets. For the two censored datasets, censored data were either omitted (option `cens.method="omit"`) or imputed as described in Methods (default, option `cens.method="cdf"`).

```
data(virload)
data(simvirload)
xviroad<-autonpde(namobs=virload,namsim=simvirload,boolsave=FALSE,
units=list(x="days",y="copies/mL"))

data(virload20)
x20<-autonpde(namobs=virload20,namsim=simvirload,boolsave=FALSE,
units=list(x="days",y="copies/mL"))
x20.omit<-autonpde(namobs=virload20,namsim=simvirload,boolsave=FALSE,
units=list(x="days",y="copies/mL"),cens.method="omit")

data(virload50)
x50<-autonpde(namobs=virload50,namsim=simvirload,boolsave=FALSE,
units=list(x="days",y="copies/mL"))
x50.omit<-autonpde(namobs=virload50,namsim=simvirload,boolsave=FALSE,
units=list(x="days",y="copies/mL"),cens.method="omit")
```

A simulation study was performed in [16], extending results from [16], to assess the performance of the method implemented in the new version of `npde`. The simulation study showed increased power to detect model misspecification, compared to simply omitting BQL data from the dataset. As expected, we observe a decrease in power when the proportion of BQL increases, since the imputation is based on the model.

4.2.4 Graphs

The plot function produces by default the 4 graphs shown in figure 3. A number of graphs have been defined, which can be individually accessed through the `plot.type` option.

Data: Figure 5 shows a plot of the data for the dataset with the highest level of censoring (LOQ=50 cp/mL) obtained by selecting the 'data' `plot.type`. By default, the imputed data will be plotted as red stars, completing the observed dataset (top left plot). Options are available to remove the BQL data from the plot (option `plot.loq=FALSE`, top right) or to plot them at the censoring value (option `impute.loq=FALSE`, bottom left).

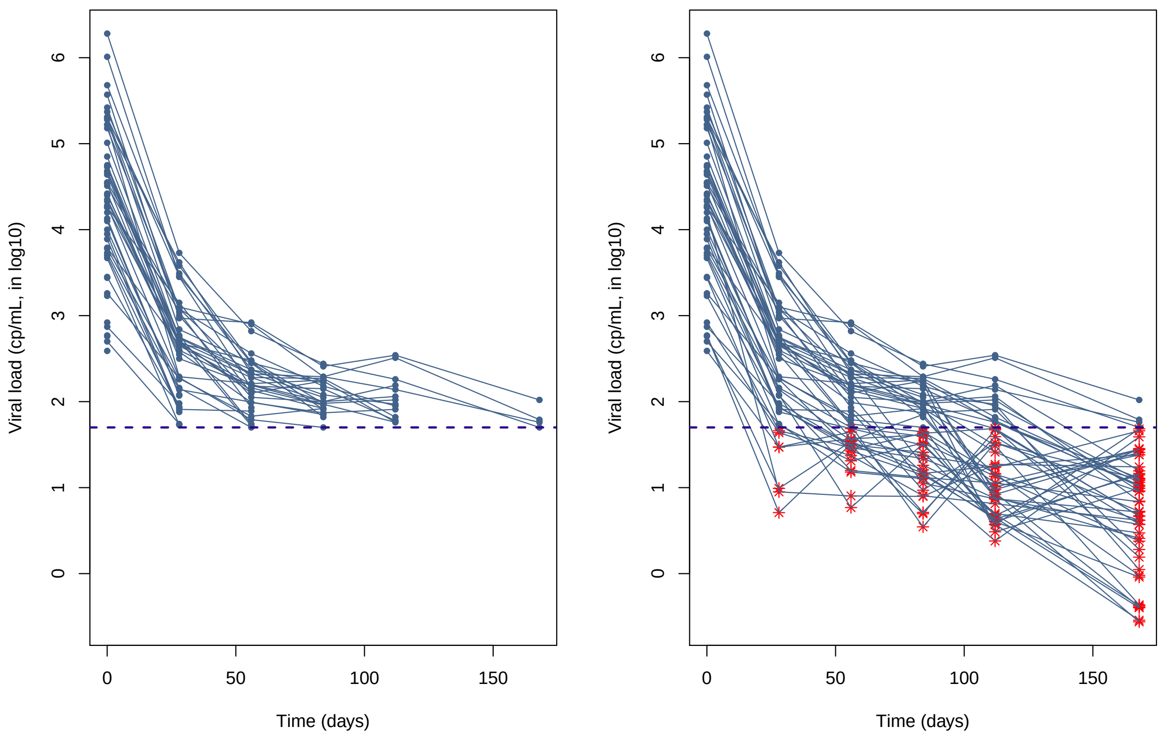

Figure 5 : *Figure showing the data, with different options. The default is shown top left, and plots the BQL data using the imputed values. Other options are to remove the BQL data from the plot (top right) or to plot them at the value used for censoring (bottom left). A dotted line shows the LOQ.*

The following code was used to produce this figure:

```
par(mfrow=c(2,2))
plot(x50,plot.type="data",new=F,ylab="Viral load (cp/mL, in log10)")
plot(x50,plot.type="data",new=F,ylab="Viral load (cp/mL, in log10)",plot.loq=FALSE)
plot(x50,plot.type="data",new=F,ylab="Viral load (cp/mL, in log10)",impute.loq=FALSE
```

Comparing the censoring methods: in this section, we compare the different censoring methods for the dataset with a high censoring value (virload50). By default, the imputation method described in section 2 is used, yielding the plots in figure 6. With the omit censoring method, censored values are omitted from the graph altogether (figure 7). With the ipred or ppred censoring methods, censored values are replaced by population or individual predictions (figure 8 and 9).

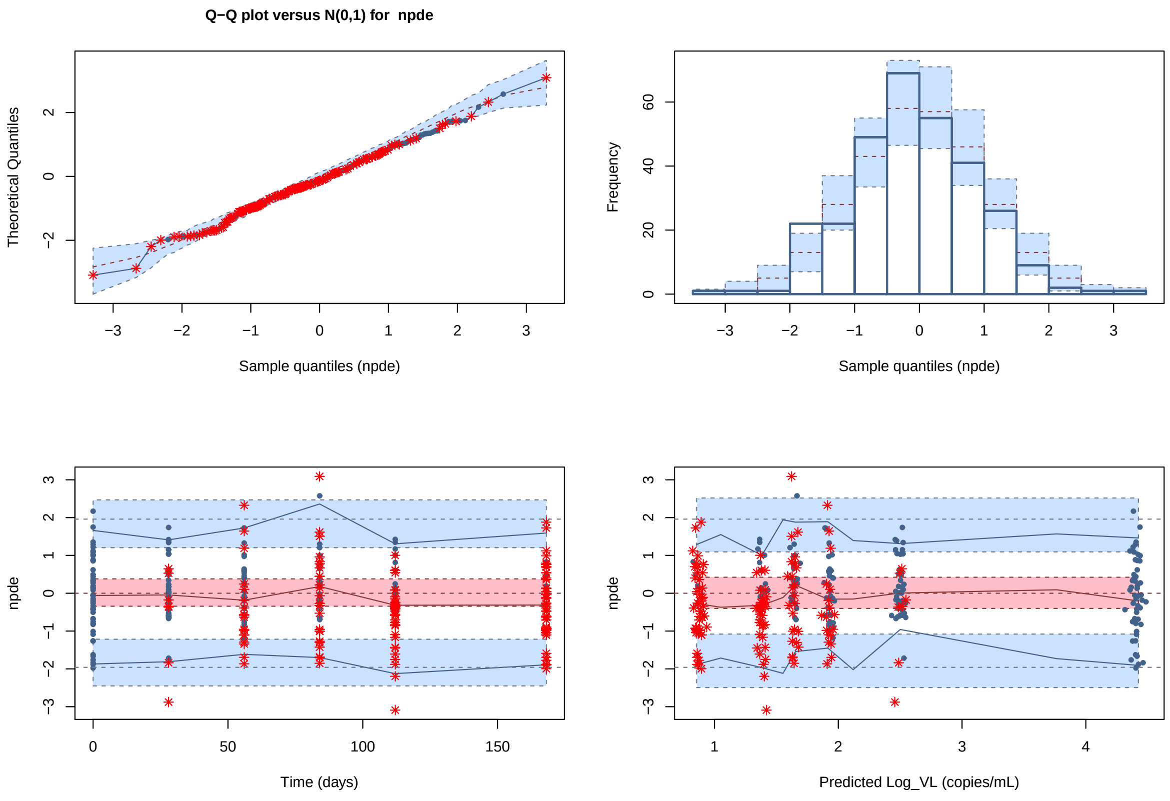

Figure 6 : *Default graphs for the virload50 dataset, default censoring method* ("cdf")*.*

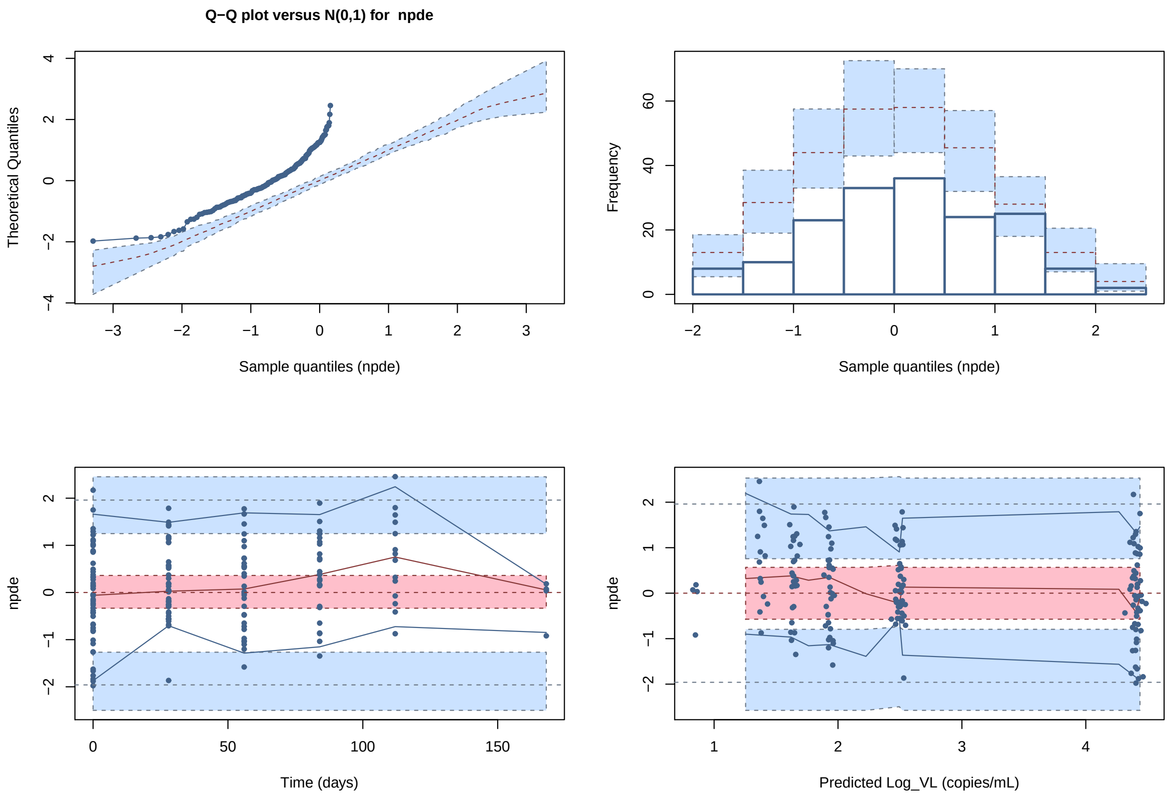

Figure 7 : *Default graphs for the virload50 dataset, censoring method* "omit".

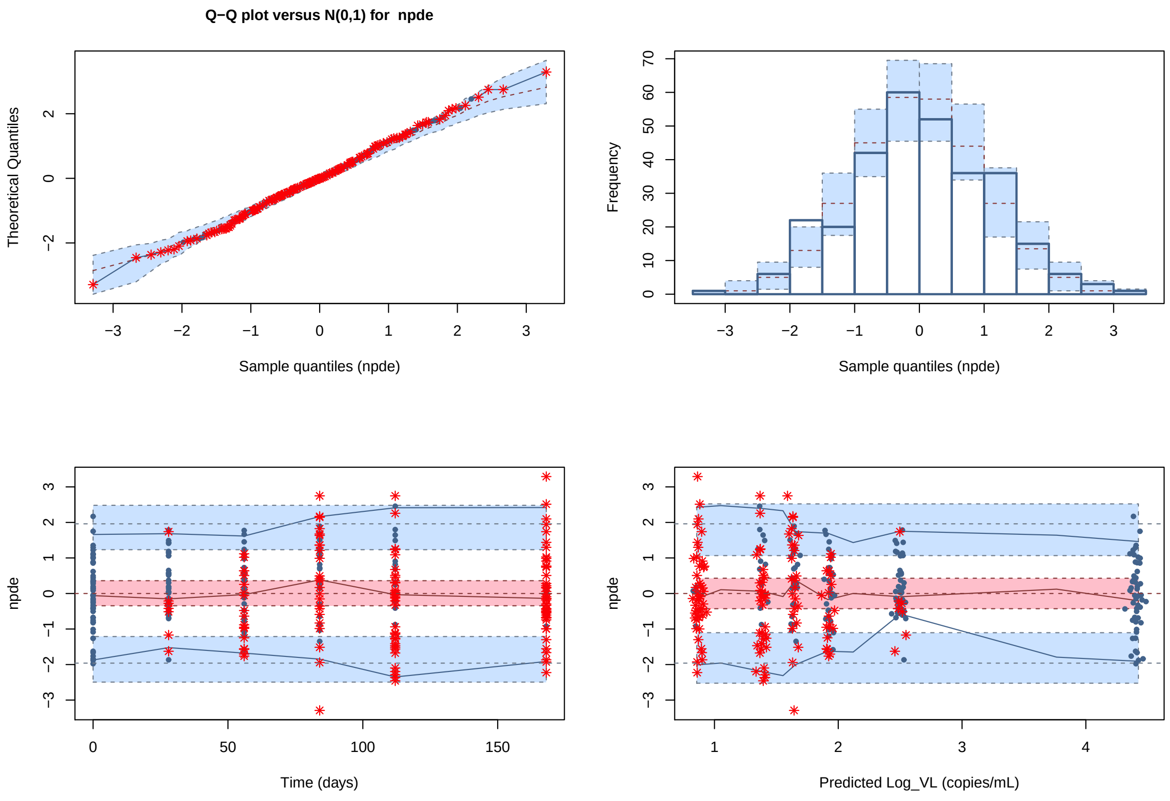

Figure 8 : *Default graphs for the virload50 dataset, censoring method* "ipred".

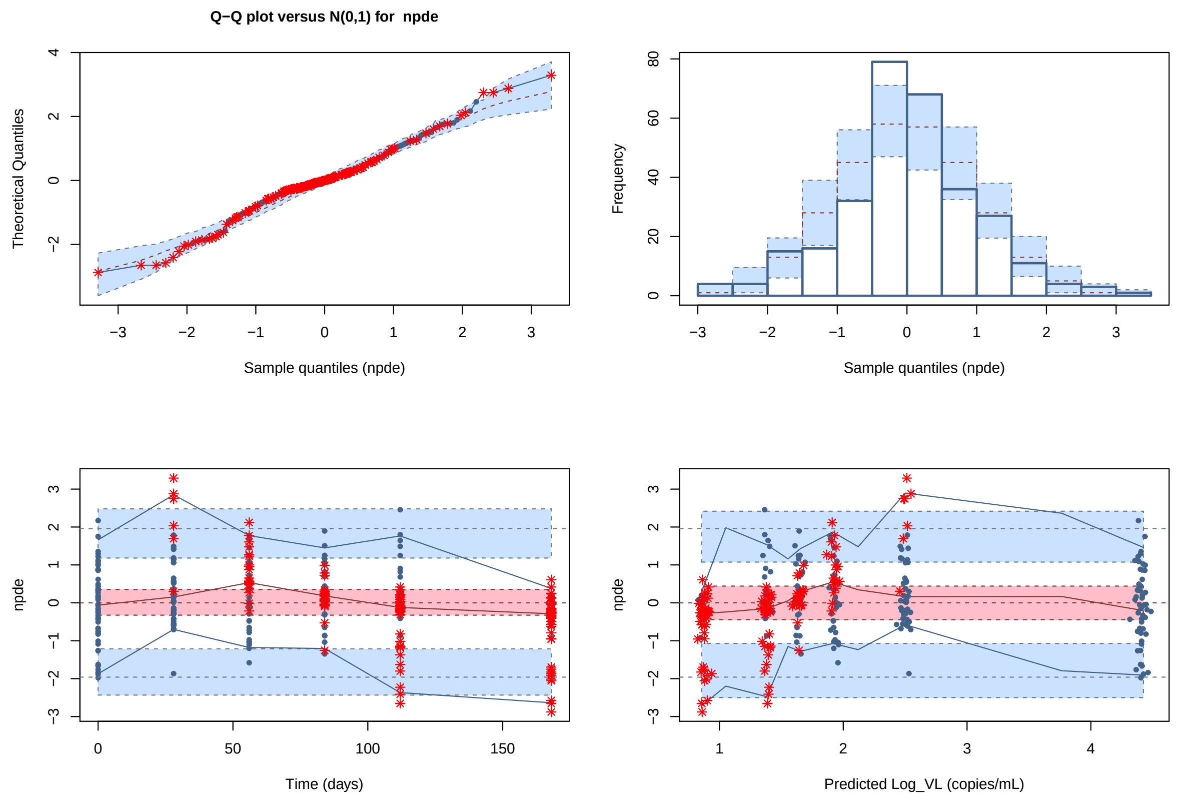

Figure 9 : *Default graphs for the virload50 dataset, censoring method* "ppred".

VPC: The Visual Predictive Check (VPC) for the data without and with imputation is shown in figure 10.

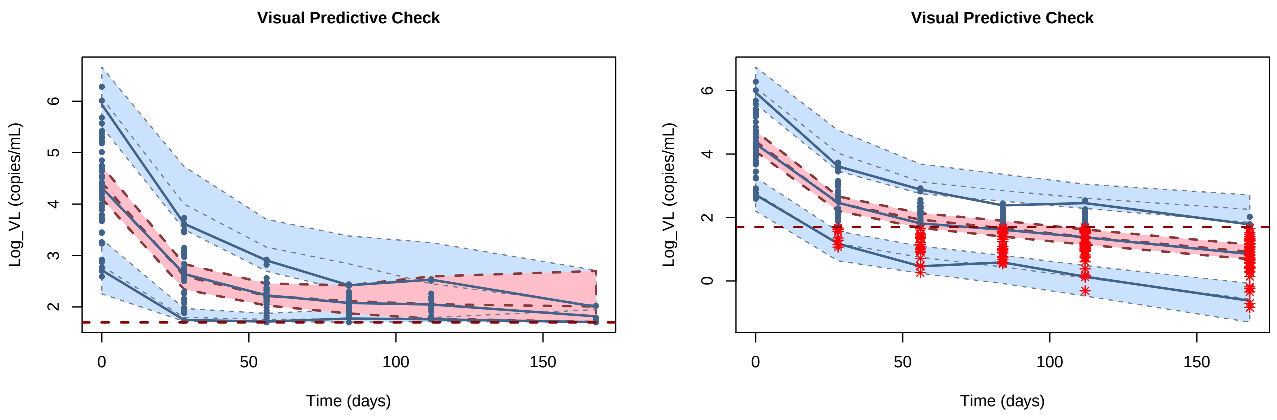

Figure 10 : *VPC obtained by removing BQL values (left) and by imputing them (right), for the dataset where LOQ=50 cp/mL. A dotted line shows the LOQ.*

Figure 11 shows the probability of being LOQ according to the model, with the corresponding prediction interval.

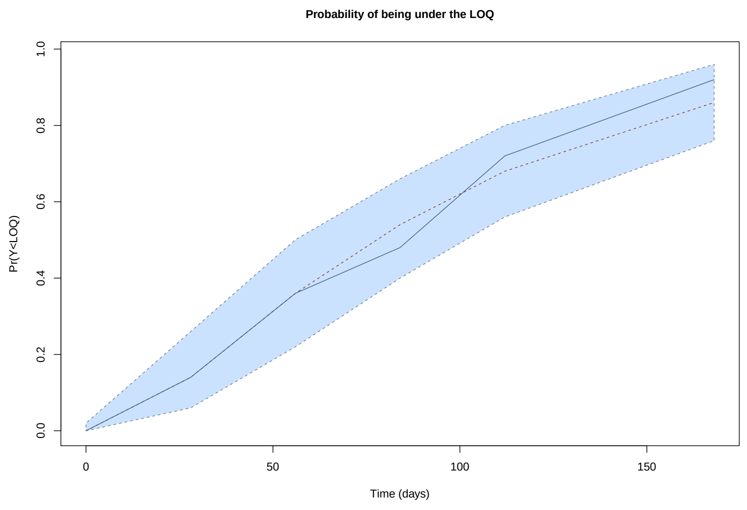

Figure 11 : *Probability of being LOQ according to the model.*

These two figures can be obtained by the following code:

```
par(mfrow=c(1,2))
plot(x50.omit,plot.type="vpc",new=F)
plot(x50,plot.type="vpc",new=F)

plot(x50,plot.type="loq")
```

Scatterplots: Scatterplots of npde or pd versus time or predictions are available. An example is given in figure 12 for npde versus time, with or without imputation. Here imputing the censored values works very well, as can be expected given that the true model was used to simulate the data.

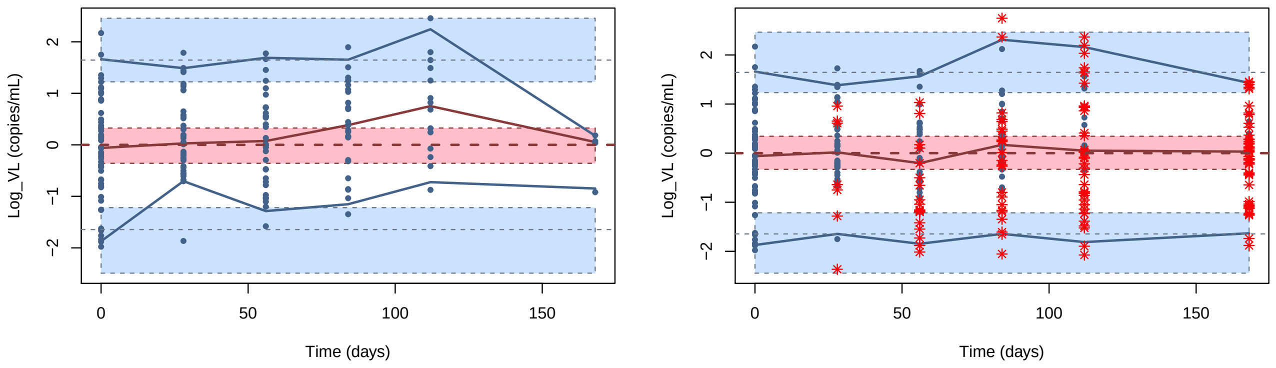

Figure 12 : *VPC obtained by removing BQL values (left) and by imputing them (right), for the dataset where LOQ=50 cp/mL. A dotted line shows the LOQ.*

This figure can be obtained by the following code:

```
par(mfrow=c(1,2))
plot(x50.omit,plot.type="x.scatter",new=F)
plot(x50,plot.type="x.scatter",new=F)
```

4.3 Model evaluation for remifentanil PK

4.3.1 Data

This dataset is one of the datasets distributed in R in the `nlme` library. The original data was collected in a study by Minto et al. [14, 15], who studied the pharmacokinetics and pharmacodynamics of remifentanil in 65 healthy volunteers. Remifentanil is a synthetic opioid derivative, used as a major analgesic before surgery or in critical care. In the study, the subjects were given remifentanil as a continuous infusion over 4 to 20 min, and measurements were collected over a period of time varying from 45 to 230 min (mean 80 min), along with EEG measurements. The following covariates were recorded: gender, age, body weight, height, body surface area and lean body mass. The recruitment was specifically designed to investigate the effect of age, with recruitment over 3 age groups (young (20-40 yr), middle-aged (40-65 yr) and elderly (over 65 yr)).

This dataset is used to illustrate the new covariate graphs. The data is not included by default in the library because it makes the package too large, however it is available from the author on request, and is also downloadable on the npde website (`http://www.npde.biostat.fr/`) in the Download page.

We modified the dataset to add a column with age group, and include the predictors Rate (rate of infusion) and Amt (dose) to generate simulations from the model.

```
data(remifent)
head(remifent)
```

4.3.2 Model

Minto et al. analysed this data using a 3-compartment model with a proportional error model and log-normal distribution for the parameters. In their base model (without covariates), they estimated the following parameters:

Population mean		Interindividual variability (CV%)	
CL (L.min^{-1})	2.46	ω_{CL} (-)	23
V_1 (L)	4.98	ω_{V_1} (-)	37
Q_2 (L.min^{-1})	1.69	ω_{Q_2} (-)	52
V_2 (L)	9.01	ω_{V_2} (-)	39
Q_3 (L.min^{-1})	0;065	ω_{Q_3} (-)	56
V_3 (L)	6.54	ω_{V_3} (-)	63
σ (-)	0.204		

Because of the size of the dataset, only 200 replications of the base model without covariates are used here but of course we advise a larger number of replications.

```
xrem<-autonpde(namobs="remifent.tab",namsim="simremifent_base.tab",
  iid=1,ix=2,iy=3,icov=c(6:12),namsav="remibase",units=list(x="hr",y="ug/L",
  covariates=c("yr","-","cm","kg","m2","kg","yr")))
```

Computing the npde yields the following results:

```
---------------------------------------------
Distribution of npde :
      nb of obs: 1992
           mean= -0.02578   (SE= 0.015 )
       variance= 0.4637   (SE= 0.015 )
       skewness= 0.2545
       kurtosis= 2.077
---------------------------------------------
Statistical tests
  t-test                    : 0.0912 .
  Fisher variance test      : 1.88e-102 ***
  SW test of normality      : 8e-18 ***
Global adjusted p-value     : 5.65e-102 ***
---
Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
---------------------------------------------
```

Here the model is strongly rejected by the test on npde: the t-test does not show any trend in the data, but the variance of npde is significantly smaller than 1. This appears clearly in the plots in figure 13 when comparing the distribution of npde to the theoretical distribution.

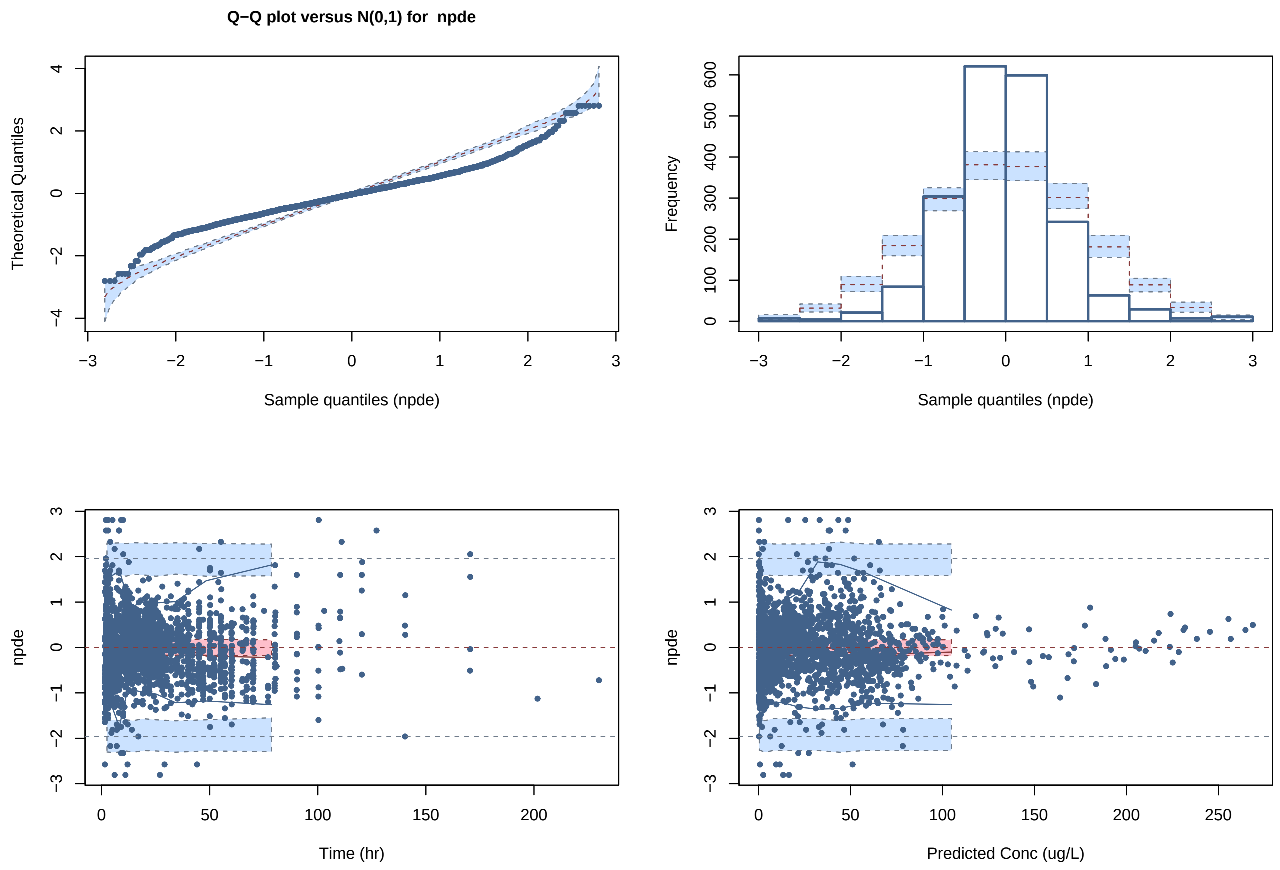

Figure 13 : *Graphs plotted by the* npde() *or* autonpde() *functions.*

Note: the very small p-value we find here is partly due to the size of the dataset (1992 observations here). Tests on npde, especially the variance and normality tests, are almost systematically positive in large real-life datasets as the power to detect model misspecification is very large and even a small number of outliers will cause the test to fail; it is generally more informative with large datasets to just use diagnostic graphs to help diagnose model misspecifications without attaching too much importance to the p-value.

4.3.3 Covariate graphs

Scatterplots: Scatterplots of npde versus a covariate can be used to assess trends, as in figure 14 for npde versus lean body mass. Here we removed the observed data for clarity.

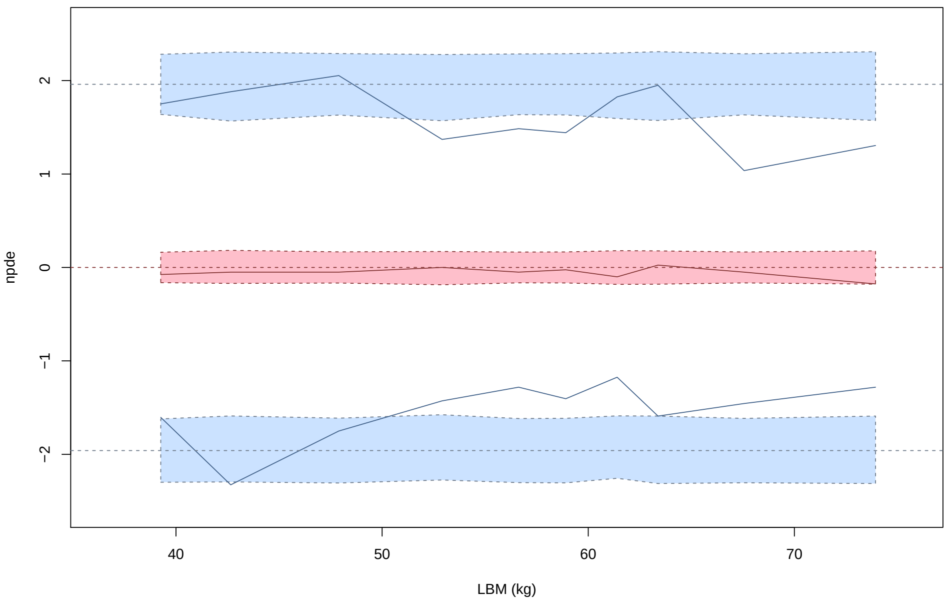

Figure 14 : *Scatterplot of npde versus lean body mass.*

Figure 15 shows the plot of cumulative density function, split by age group.

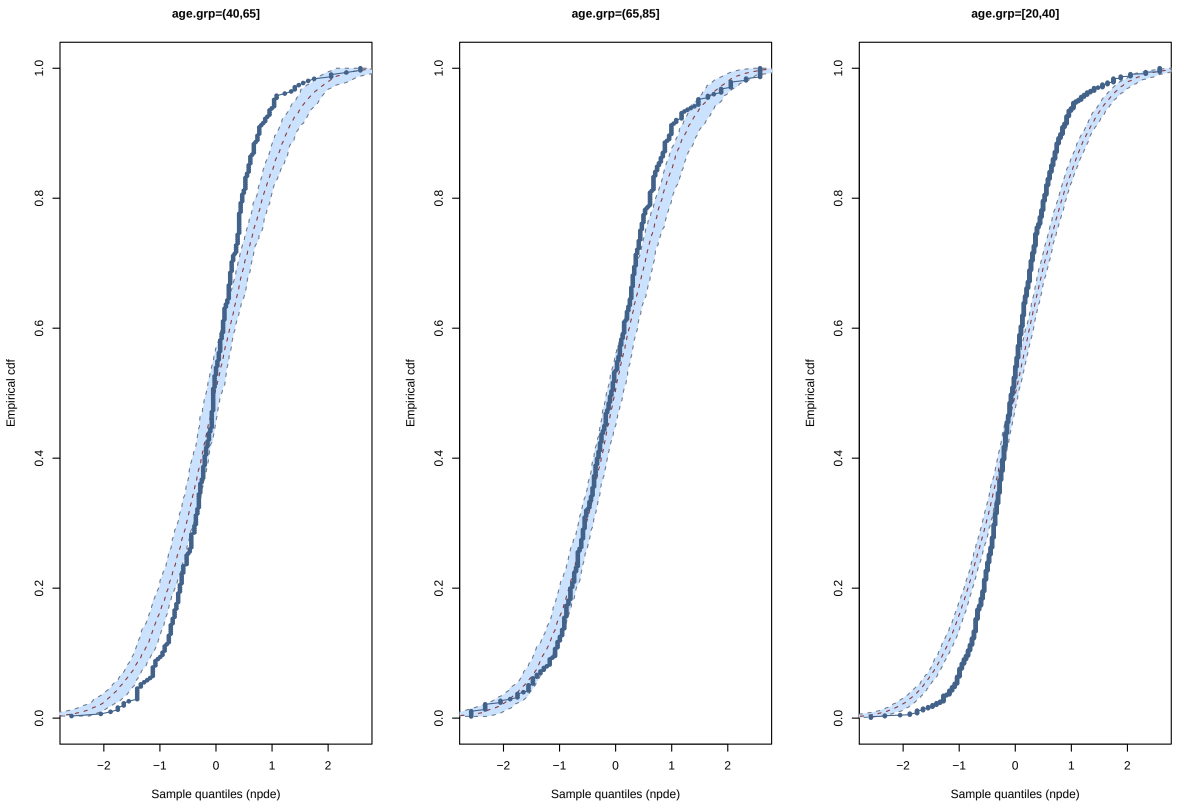

Figure 15 : *Empirical cumulative function of npde, split by quantiles of age.*

These figures can be obtained by the following code:

```
plot(xrem,plot.type="cov.scatter",which.cov="LBM",plot.obs=FALSE)
plot(xrem,plot.type="ecdf",covsplit=TRUE,which.cov="age.grp",bands=TRUE,
plot.obs=FALSE)
```

4.4 Types of graphs

Table II shows which plot types are available (some depend on whether for instance covariates or data below the limit of quantification are present in the dataset) for a NpdeObject object. Given an object x resulting from a call to npde or autonpde, default plots can be produced using the following command:

```
plot(x)
```

Different plots are also available using the option plot.type, as in:

```
plot(x,plot.type="data")
```

The final five plots can also be accessed with the base plot and the option covsplit=TRUE. For instance,
`plot(x,plot.type="cov.x.scatter")` is equivalent to `plot(x,plot.type="x.scatter",covspli`

4.5 Options for graphs

Default layout for graphs in the npde library can be modified through the use of many options. An additional document, `demo_npde2.0.pdf`, is included in the `inst` directory of the package, presenting additional examples of graphs and how to change the options.

Table III following table shows the options that can be set, either by specifying them on the fly in a call to plot applied to a NpdeObject object, or by storing them in the prefs component of the object.

Note that not all of the graphical parameters in `par()` can be used, but it is possible for instance to use the xaxt="n" option below to suppress plotting of the X-axis, and to then add back the axis with the R function axis() to tailor the tickmarks or change colours as wanted. It is also possible of course to extract npde, fitted values or original data to produce any of these plots by hand if the flexibility provided in the library isn't sufficient. Please refer to the document `demo_npde2.0.pdf` for examples of graphs using these options.

Table II: *Types of plots available.*

Plot type	Description
data	Plots the observed data in the dataset
x.scatter	Scatterplot of the npde versus the predictor X (optionally can plot pd or npd instead)
pred.scatter	Scatterplot of the npde versus the population predicted values
cov.scatter	Scatterplot of the npde versus covariates
vpc	Plots a Visual Predictive Check
loq	Plots the probability for an observation to be BQL, versus the predictor X
ecdf	Empirical distribution function of the npde(optionally pd or npd)
hist	Histogram of the npde(optionally pd or npd)
qqplot	QQ-plot of the npdeversus its theoretical distribution (optionally pd or npd)
cov.x.scatter	Scatterplot of the npde versus the predictor X, split by covariate
cov.pred.scatter	Scatterplot of the npde versus the population predicted values, split by covariate
cov.ecdf	Empirical distribution function of the npde(optionally pd or npd), split by covariate
cov.hist	Histogram of the npde(optionally pd or npd), split by covariate
cov.qqplot	QQ-plot of the npdeversus its theoretical distribution (optionally pd or npd) , split by covariate

Table III: *Default graphical parameters. Any option not defined by the user is automatically set to its default value.*

Parameter	Description	Default value
General graphical options		
new	Whether a new plot should be produced	TRUE
ask	Whether users should be prompted before each new plot (if TRUE)	FALSE
interactive	Output is produced for some plots (most notably when binning is used, this prints out the boundaries of the binning intervals) if TRUE	FALSE
xaxt	A character which specifies the x axis type. Specifying "n" suppresses plotting of the axis	empty
yaxt	A character which specifies the y axis type. Specifying "n" suppresses plotting of the axis	empty
frame.plot	If TRUE, a box is drawn around the current plot	TRUE
main	Title	empty
xlab	Label for the X-axis	empty
ylab	Label for the Y-axis	empty
xlog	Scale for the X-axis (TRUE: logarithmic scale)	FALSE
ylog	Scale for the Y-axis (TRUE: logarithmic scale)	FALSE
cex	A numerical value giving the amount by which plotting text and symbols should be magnified relative to the default	1
cex.axis	Magnification to be used for axis annotation relative to the current setting of ’cex’	1
cex.lab	Magnification to be used for x and y labels relative to the current setting of ’cex’	1
cex.main	Magnification to be used for main titles relative to the current setting of ’cex’	1
mfrow	Page layout (NA: layout set by the plot function or before)	NA
xlim	Range for the X-axis (NA: ranges set by the plot function)	NA
ylim	Range for the Y-axis (NA: ranges set by the plot function)	NA
type	Type of plot ("b": both, "p": points, "l": lines). Defaults to b for data and p for other plots	b/p

Parameter	Description	Default value
Options controlling the type of plots		
plot.type	Type of plot (see documentation for list)	default
ilist	List of subjects to include in the individual plots	1:N
smooth	Whether a smooth should be added to certain plots	FALSE
line.smooth	Type of smoothing (l=line, s=spline)	s
which.cov	Which covariates to use for the plot	all
ncat	Number of categories in which to split continuous covariates for graphs Defaults to 3, splitting in $<Q_1$, Q_1-Q_3, $>Q_3$	3
which.resplot	Type of residual plot ("res.vs.x": scatterplot versus X, "res.vs.pred": scatterplot versus predictions, "dist.hist": histogram, "dist.qqplot": QQ-plot)	c("res.vs.x","res.vs.pred", "dist.qqplot","dist.hist")
box	If TRUE, boxplots are produced instead of scatterplots	FALSE
Options for colours and line types		
col	Default symbol and line colour	black
lty	Default line type	1 (straight line)
lwd	Default line width	1
pch.pobs	Default symbol type	20 (dot)
pch.pcens	Default symbol type for censored observations	8 ()
col.pobs	Symbol colour to use for observations (points)	steelblue4
col.lobs	Symbol colour to use for observations (lines)	steelblue4
col.pcens	Symbol colour to use for censored observations	red
lty.lobs	Line type for observations	1
lwd.lobs	Line width for observations	1
col.abline	Colour of the horizontal/vertical lines added to the plots	"DarkBlue"
lty.abline	Type of the lines added to the plots	2 (dashed)
lwd.abline	Width of the lines added to the plots	2
col.fillpi	Colour used to fill histograms andprediction bands	slategray1
col.fillmed	Colour used to fill prediction band on the median (VPC, npde)	pink
col.lmed	Colour used to plot the predicted median (VPC, npde)	indianred4
col.lpi	Colour used to plot lower and upper quantiles	slategray4

Parameter	Description	Default value
lty.lmed	Line type used to plot the predicted median (VPC, npde)	2
lty.lpi	Line type used to plot lower and upper quantiles	2
lwd.lmed	Line width used to plot the predicted median (VPC, npde)	1
lwd.lpi	Line width used to plot lower and upper quantiles	1
Graphical options for VPC and residual plots		
bands	Whether prediction intervals should be plotted	TRUE
approx.pi	If TRUE, samples from $\mathcal{N}(0,1)$ are used to plot prediction intervals, while if FALSE, prediction bands are obtained using pd/npde computed for the simulated data	TRUE
vpc.method	Method used to bin points (one of "equal", "width", "user" or "optimal"); at least the first two letters of the method need to be specified	"equal"
vpc.bin	Number of binning intervals	10
vpc.interval	Size of interval	0.95
vpc.breaks	Vector of breaks used with user-defined breaks (vpc.method="user")	NULL
vpc.extreme	Can be set to a vector of 2 values to fine-tune the behaviour of the binning algorithm at the boundaries; specifying c(0.01,0.99) with the "equal" binning method and vpc.bin=10 will create 2 extreme bands containing 1% of the data on the X-interval, then divide the region within the two bands into the remaining 8 intervals each containing the same number of data; in this case the intervals will all be equal except for the two extreme intervals, the size of which is fixed by the user; complete fine-tuning can be obtained by setting the breaks with the vpc.method="user"	NULL
pi.size	Width of the prediction interval on the quantiles	0.95
vpc.lambda	Value of lambda used to select the optimal number of bins through a penalised criterion	0.3
vpc.beta	Value of beta used to compute the variance-based criterion (Jopt,beta(I)) in the clustering algorithm	0.2
bands.rep	Number of simulated datasets used to compute prediction bands	200

References

[1] Brendel, K., Comets, E., Laffont, C., Laveille, C., and Mentré, F. Metrics for external model evaluation with an application to the population pharmacokinetics of gliclazide. *Pharmaceutical Research 23* (2006), 2036–49.

[2] Brendel, K., Comets, E., Laffont, C., and Mentré, F. Evaluation of different tests based on observations for external model evaluation of population analyses. *Journal of Pharmacokinetics and Pharmacodynamics 37* (2010), 49–65.

[3] Comets, E., Brendel, K., and Mentré, F. Computing normalised prediction distribution errors to evaluate nonlinear mixed-effect models: the npde add-on package for R. *Computer Methods and Programs in Biomedicine 90* (2008), 154–66.

[4] Comets, E., Brendel, K., and Mentré, F. Model evaluation in nonlinear mixed effect models, with applications to pharmacokinetics. *Journal de la Société Française de Statistique 151* (2010), 106–28.

[5] Ding, A. A., and Wu, H. Relationships between antiviral treatment effects and biphasic viral decay rates in modeling HIV dynamics. *Mathematical Biosciences 160* (1999), 63–82.

[6] Fraley, C., and Raftery, A. E. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association 97* (2002), 611–31.

[7] Fraley, C., and Raftery, A. E. MCLUST version 3 for R: normal mixture modeling and model-based clustering. *Technical Report No. 504, Department of Statistics, University of Washington (revised 2009)* (2006).

[8] Genolini, C. *Construire un Package - Classic et S4*. INSERM U669, Paris, France, 2010.

[9] Goujard, C., Barrail-Tran, A., Duval, X., Nembot, G., Panhard, X., Savic, R., Descamps, D., Vrijens, B., Taburet, A., Mentré, F., and the ANRS 134 study group. Virological response to atazanavir, ritonavir and tenofovir/emtricitabine: relation to individual pharmacokinetic parameters and adherence measured by medication events monitoring system (MEMS) in naïve HIV-infected patients (ANRS134 trial). *International AIDS Society 2010* (2010), Abstr WEPE0094.

[10] Higham, N. J. Computing the polar decomposition – with applications. *SIAM Journal on Scientific and Statistical Computing 7* (1986), 1160–74.

[11] LAVIELLE, M. *MONOLIX (MOdèles NOn LInéaires à effets miXtes)*. MONOLIX group, Orsay, France, 2005.

[12] MENTRÉ, F., AND ESCOLANO, S. Prediction discrepancies for the evaluation of nonlinear mixed-effects models. *Journal of Pharmacokinetics and Pharmacodynamics 33* (2006), 345–67.

[13] MESNIL, F., MENTRÉ, F., DUBRUC, C., THÉNOT, J., AND MALLET, A. Population pharmacokinetics analysis of mizolastine and validation from sparse data on patients using the non-parametric maximum likelihood method. *Journal of Pharmacokinetics and Biopharmaceutics 26* (1998), 133–161.

[14] MINTO, C., SCHNIDER, T., EGAN, T., YOUNGS, E., LEMMENS, H., PL, P. G., BILLARD, V., HOKE, J., MOORE, K., HERMANN, D., MUIR, K., MANDEMA, J., AND SHAFER, S. Influence of age and gender on the pharmacokinetics and pharmacodynamics of remifentanil. I. Model development. *Anesthesiology 86* (1997), 10–23.

[15] MINTO, C., SCHNIDER, T., AND SHAFER, S. Pharmacokinetics and pharmacodynamics of remifentanil. II. Model application. *Anesthesiology 86* (1997), 24–33.

[16] NGUYEN, T. H. T., COMETS, E., AND MENTRÉ, F. Prediction discrepancies (pd) for evaluation of models with data under limit of quantification. *Journal of Pharmacokinetics and Pharmacodynamics 39* (2012), 499–518.

[17] R DEVELOPMENT CORE TEAM. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2006. ISBN 3-900051-07-0.

[18] SHEINER, L., AND BEAL, S. *NONMEM Version 5.1*. University of California, NONMEM Project Group, San Francisco, 1998.

[19] WRIGHT, S. Adjusted p-values for simultaneous inference. *Biometrics 48* (1992), 1005–13.

Appendix - Theophylline example

R commands to run the theophylline example

The R commands used to run the theophylline example (with the dataframes included in the npde library) are given below. Simply copy and paste to your R session to execute.

```
library(npde)

data(theopp)
data(simtheopp)
xtheo<-autonpde(namobs=theopp,namsim=simtheopp,
  iid=1,ix=3,iy=4,namsav="results/theo_nocov",units=list(x="hr",y="mg/L"))

plot(xtheo)
plot(xtheo,plot.type="data")
plot(xtheo,plot.type="vpc")
```

The data

Below, the data for the first two subjects in the NONMEM data file is shown:

```
1      4.02      0.          .       79.6
1       .        0.25      2.84        .
1       .        0.57      6.57        .
1       .        1.12     10.5         .
1       .        2.02      9.66        .
1       .        3.82      8.58        .
1       .        5.1       8.36        .
1       .        7.03      7.47        .
1       .        9.05      6.89        .
1       .       12.12      5.94        .
1       .       24.37      3.28        .
2      4.4       0.          .       72.4
2       .         .27      1.72        .
2       .         .52      7.91        .
2       .        1.        8.31        .
2       .        1.92      8.33        .
2       .        3.5       6.85        .
2       .        5.02      6.08        .
2       .        7.03      5.4         .
2       .        9.        4.55        .
2       .       12.        3.01        .
2       .       24.3        .90        .
```

Control file used for the NONMEM analysis

```
$PROB  THEOPHYLLINE   POPULATION DATA
$INPUT      ID DOSE=AMT TIME DV WT
$DATA       theopp.tab
$SUBROUTINES  ADVAN2 TRANS2

$PK
;THETA(1)=MEAN ABSORPTION RATE CONSTANT (1/HR)
;THETA(2)=MEAN ELIMINATION RATE CONSTANT (1/HR)
;THETA(3)=SLOPE OF CLEARANCE VS WEIGHT RELATIONSHIP (LITERS/HR/KG)
   CALLFL=1
   KA=THETA(1)*EXP(ETA(1))
   V=THETA(2)*EXP(ETA(2))
   K=THETA(3)*EXP(ETA(3))
   CL=K*V
   S2=V

$ERROR
SLOP=THETA(4)
SINT=THETA(5)
IPRED=F
W=SLOP*F+SINT
Y=F+W*EPS(1)
IRES=IPRED-DV
IWRES=IRES/W

$THETA  (.1,3,5) (0,0.5,) (.004,.1,2) (0,0.2,) (0,0.1,)
$OMEGA 0.2
$OMEGA BLOCK(2) 0.2 0.05 0.2
$SIGMA 1 FIX

$EST NOABORT METHOD=COND INTERACTION MAXEVAL=2000  PRINT=5
$COV
```

Results obtained with NONMEM

```
 ************************************************************************************
 **********                                                                **********
 **********                 MINIMUM VALUE OF OBJECTIVE FUNCTION            **********
 **********                                                                **********
 ************************************************************************************
 **************************           86.664     ***********************************
 ************************************************************************************
 **********                                                                **********
 **********                       FINAL PARAMETER ESTIMATE                 **********
 **********                                                                **********
 ************************************************************************************

 THETA - VECTOR OF FIXED EFFECTS PARAMETERS   *********

          TH 1      TH 2      TH 3      TH 4      TH 5
       1.51E+00  4.60E-01  8.73E-02  8.81E-02  2.58E-01

 OMEGA - COV MATRIX FOR RANDOM EFFECTS - ETAS  ********

          ETA1      ETA2      ETA3
 ETA1
+        4.43E-01
 ETA2
+        0.00E+00  1.45E-02
 ETA3
+        0.00E+00  1.62E-02  1.82E-02

 SIGMA - COV MATRIX FOR RANDOM EFFECTS - EPSILONS  ****

          EPS1
 EPS1
+        1.00E+00
```

NONMEM control file used for the simulations

```
$PROB  THEOPHYLLINE   POPULATION DATA
$INPUT      ID DOSE=AMT TIME DV WT
$DATA       theopp.tab
$SUBROUTINES  ADVAN2 TRANS2

$PK
;THETA(1)=MEAN ABSORPTION RATE CONSTANT (1/HR)
;THETA(2)=VOLUME OF DISTRIBUTION (LITERS)
;THETA(3)=MEAN ELIMINATION RATE CONSTANT (1/HR)
   CALLFL=1
   KA=THETA(1)*EXP(ETA(1))
   V=THETA(2)*EXP(ETA(2))
   K=THETA(3)*EXP(ETA(3))
   CL=K*V
   S2=V
$ERROR
SLOP=THETA(4)
SINT=THETA(5)
IPRED=F
W=SLOP*F+SINT
Y=F+W*EPS(1)
FSIM=Y

$THETA 1.51 0.46 0.0873 0.0881 0.258
$OMEGA 0.443
$OMEGA BLOCK(2) 0.0145 0.0162 0.0182
$SIGMA 1 FIX

$SIMULATION (82015831)  ONLYSIM SUBPROBLEMS=100
$TABLE ID TIME FSIM IPRED NOPRINT NOHEADER NOAPPEND FILE=simtheopp.tab
```

Simulated data

Below, the first few lines of the simulated data file created by the previous control file are shown:

```
 1.0000E+00  0.0000E+00 -9.0212E-02  0.0000E+00
 1.0000E+00  2.5000E-01  2.2892E+00  2.5691E+00
 1.0000E+00  5.7000E-01  4.2279E+00  4.6287E+00
 1.0000E+00  1.1200E+00  5.4979E+00  6.3074E+00
 1.0000E+00  2.0200E+00  7.9173E+00  6.8719E+00
 1.0000E+00  3.8200E+00  5.3943E+00  6.1422E+00
 1.0000E+00  5.1000E+00  4.3926E+00  5.4793E+00
 1.0000E+00  7.0300E+00  5.0335E+00  4.5902E+00
 1.0000E+00  9.0500E+00  3.3301E+00  3.8114E+00
 1.0000E+00  1.2120E+01  3.3686E+00  2.8730E+00
 1.0000E+00  2.4370E+01  6.0324E-01  9.3011E-01
 2.0000E+00  0.0000E+00  2.0597E-01  0.0000E+00
 2.0000E+00  2.7000E-01  2.3492E+00  3.1259E+00
 2.0000E+00  5.2000E-01  4.4722E+00  5.1687E+00
 2.0000E+00  1.0000E+00  5.7317E+00  7.5603E+00
 2.0000E+00  1.9200E+00  8.6685E+00  9.1770E+00
 2.0000E+00  3.5000E+00  9.6393E+00  8.9901E+00
 2.0000E+00  5.0200E+00  7.9179E+00  8.1473E+00
 2.0000E+00  7.0300E+00  6.7075E+00  7.0363E+00
 2.0000E+00  9.0000E+00  5.4641E+00  6.0802E+00
 2.0000E+00  1.2000E+01  5.0094E+00  4.8659E+00
 2.0000E+00  2.4300E+01  2.0761E+00  1.9518E+00
```

This dataset contains the following columns: patient ID (idsim), time (xsim), simulated data (ysim), individual predictions. The program only uses the first 3 columns.

Saved results

In the example, the results from the R script are saved to a file called theophylline.npde. The first lines of this file, corresponding to the first 2 subjects are shown below:

```
id xobs  yobs  ypred          npde              pd
1  0.25  2.84  2.9238643    0.125661346855074  0.55
1  0.57  6.57  4.6822991    2.05374891063182   0.85
1  1.12  10.5  6.264357     2.32634787404084   0.99
1  2.02  9.66  6.986255     0.524400512708041  0.98
1  3.82  8.58  6.511039     0.253347103135800  0.93
1  5.1   8.36  5.895675     0.674489750196082  0.96
1  7.03  7.47  5.064736     1.64485362695147   0.97
1  9.05  6.89  4.302909     0.772193214188685  0.99
1  12.12 5.94  3.29402      1.75068607125217   0.99
1  24.37 3.28  1.16874348   2.32634787404084   0.99
2  0.27  1.72  3.39568076  -0.994457883209753  0.16
2  0.52  7.91  5.222963     2.32634787404084   0.9
2  1     8.31  6.984615     0.674489750196082  0.71
2  1.92  8.33  7.707843    -0.305480788099397  0.64
2  3.5   6.85  7.47791     -1.55477359459685   0.33
2  5.02  6.08  6.43454     -0.80642124701824   0.43
2  7.03  5.4   5.612031    -0.279319034447454  0.48
2  9     4.55  4.862751     0.100433720511470  0.43
2  12    3.01  3.771684    -0.279319034447454  0.26
2  24.3  0.9   1.2906205   -0.553384719555673  0.31
```