Package ‘nonmem2rx’

May 9, 2026

Type Package

Title Converts 'NONMEM' Models to 'rxode2'

Version 0.1.9

Maintainer Matthew Fidler <`matthew.fidler@gmail.com`>

Description 'NONMEM' has been a tool for running nonlinear mixed effects models since the 80s and is still used today (Bauer 2019 <doi:10.1002/psp4.12404>). This tool allows you to convert 'NONMEM' models to 'rxode2' (Wang, Hallow and James (2016) <doi:10.1002/psp4.12052>) and with simple models 'nlmixr2' syntax (Fidler et al (2019) <doi:10.1002/psp4.12445>). The 'nlmixr2' syntax requires the residual specification to be included and it is not always translated. If available, the 'rxode2' model will read in the 'NONMEM' data and compare the simulation for the population model ('PRED') individual model ('IPRED') and residual model ('IWRES') to immediately show how well the translation is performing. This saves the model development time for people who are creating an 'rxode2' model manually. Additionally, this package reads in all the information to allow simulation with uncertainty (that is the number of observations, the number of subjects, and the covariance matrix) with a 'rxode2' model. This is complementary to the 'babelmixr2' package that translates 'nlmixr2' models to 'NONMEM' and can convert the objects converted from 'nonmem2rx' to a full 'nlmixr2' fit.

License GPL (>= 3)

URL `https://nlmixr2.github.io/nonmem2rx/`,
`https://github.com/nlmixr2/nonmem2rx/`

Encoding UTF-8

LinkingTo dparser, Rcpp, rxode2

Imports checkmate, digest, dparser, lotri, Rcpp, rxode2 (> 2.0.13), magrittr, cli, tools, utils, data.table, qs2, xml2, ggplot2, ggforce, crayon

Suggests devtools, testthat (>= 3.0.0), nonmemica, NMdata, nonmem2R, withr, nlme, dplyr, xgxr, vdiffr, knitr, rmarkdown, spelling

Config/testthat/edition 3

RoxygenNote 7.3.3

VignetteBuilder knitr

Language en-US

NeedsCompilation yes

Author Matthew Fidler [aut, cre] (ORCID:
<https://orcid.org/0000-0001-8538-6691>),
Philip Delff [ctb],
Gabriel Staples [ctb] (string insensitive compare)

Repository CRAN

Date/Publication 2025-11-27 06:10:18 UTC

Contents

`as.nonmem2rx` *Convert a model to a nonmem2rx model*

Description

Convert a model to a nonmem2rx model

Usage

```
as.nonmem2rx(model1, model2, compress = TRUE)
```

Arguments

`model1`	Input model 1
`model2`	Input model 2
`compress`	boolean to compress the ui at the end

Value

nonmem2rx model

Author(s)

Matthew L. Fidler

Examples

```
 mod <- nonmem2rx(system.file("mods/cpt/runODE032.ctl", package="nonmem2rx"),
                  determineError=FALSE, lst=".res", save=FALSE)

 mod2 <-function() {
   ini({
     lcl <- 1.37034036528946
     lvc <- 4.19814911033061
     lq <- 1.38003493562413
     lvp <- 3.87657341967489
     RSV <- c(0, 0.196446108190896, 1)
     eta.cl ~ 0.101251418415006
     eta.v ~ 0.0993872449483344
     eta.q ~ 0.101302674763154
     eta.v2 ~ 0.0730497519364148
   })
   model({
     cmt(CENTRAL)
     cmt(PERI)
     cl <- exp(lcl + eta.cl)
     v <- exp(lvc + eta.v)
     q <- exp(lq + eta.q)
     v2 <- exp(lvp + eta.v2)
     v1 <- v
     scale1 <- v
     k21 <- q/v2
     k12 <- q/v
     d/dt(CENTRAL) <- k21 * PERI - k12 * CENTRAL - cl * CENTRAL/v1
     d/dt(PERI) <- -k21 * PERI + k12 * CENTRAL
     f <- CENTRAL/scale1
     f ~ prop(RSV)
   })
 }

new <- try(as.nonmem2rx(mod2, mod))
if (!inherits(new, "try-error")) print(new, page=1)
```

`nmcov` *Read in data file*

Description

Read in data file

Usage

```
nmcov(file, ...)
```

Arguments

`file`	file name to read the results from
`...`	other parameters passed to `data.table::fread`

Value

A matrix with covariance step from NONMEM

Author(s)

Philip Delff and Matthew L. Fidler

Examples

```
nmcov(system.file("mods/cpt/runODE032.cov", package="nonmem2rx"))
```

`nmext` *Reads the NONMEM* `.ext` *file for final parameter information*

Description

Reads the NONMEM `.ext` file for final parameter information

Usage

```
nmext(file)
```

Arguments

`file`	File where the list is located

Value

return a list with `$theta`, `$eta` and `$eps`

Author(s)

Matthew L. Fidler

Examples

```
nmext(system.file("run001.ext", package="nonmem2rx"))
```

`nmgrd` *Reads the NONMEM `.grd` file for final parameter gradient*

Description

Reads the NONMEM `.grd` file for final parameter gradient

Usage

```
nmgrd(file)
```

Arguments

`file`	File where the list is located

Value

return a list with `$rawGrad`

Author(s)

Matthew L. Fidler

Examples

```
nmgrd(system.file("mods/cpt/runODE032.grd", package="nonmem2rx"))
```

`nminfo` *Get the most accurate information you can get from NONMEM*

Description

Get the most accurate information you can get from NONMEM

Usage

```
nminfo(
  file,
  mod = ".mod",
  xml = ".xml",
  ext = ".ext",
  cov = ".cov",
  phi = ".phi",
  lst = ".lst",
  grd = ".grd",
```

```
  useXml = TRUE,
  useExt = TRUE,
  useCov = TRUE,
  usePhi = TRUE,
  useLst = TRUE,
  strictLst = FALSE,
  verbose = FALSE
)
```

Arguments

`file`	nonmem file, like control stream, phi. This function will remove the extension to try to get the right information. It preferentially selects the most accurate estimates from the file.
`mod`	the NONMEM output extension, defaults to `.mod`
`xml`	the NONMEM xml file extension , defaults to `.xml`
`ext`	the NONMEM ext file extension, defaults to `.ext`
`cov`	the NONMEM covariance file extension, defaults to `.cov`
`phi`	the NONMEM eta/phi file extension, defaults to `.phi`
`lst`	the NONMEM output extension, defaults to `.lst`
`grd`	the NONMEM gradient file extension, defaults to `.grd`
`useXml`	if present, use the NONMEM xml file to import much of the NONMEM information
`useExt`	if present, use the NONMEM ext file to extract parameter estimates (default `TRUE`), otherwise defaults to parameter estimates extracted in the NONMEM output
`useCov`	if present, use the NONMEM cov file to import the covariance, otherwise import the covariance with list file
`usePhi`	if present, use the NONMEM phi file to extract etas (default `TRUE`), otherwise defaults to etas in the tables (if present)
`useLst`	if present, use the NONMEM lst file to extract NONMEM information
`strictLst`	The list parsing needs to be correct for a successful load (default `FALSE`).
`verbose`	this is a flag to be more verbose when reading information in, by default this is `FALSE`

Value

list of NONMEM information

Author(s)

Matthew L. Fidler

Examples

```
nminfo(system.file("mods/cpt/runODE032.res", package="nonmem2rx"))
```

`nmlst` *Reads the NONMEM `.lst` file for final parameter information*

Description

Reads the NONMEM `.lst` file for final parameter information

Usage

```
nmlst(file, strictLst = FALSE)
```

Arguments

`file`	File where the list is located
`strictLst`	The list parsing needs to be correct for a successful load (default `FALSE`).

Value

return a list with `$theta`, `$eta` and `$eps` and other information about the control stream

Author(s)

Matthew L. Fidler

Examples

```
nmlst(system.file("mods/DDMODEL00000322/HCQ1CMT.lst", package="nonmem2rx"))
nmlst(system.file("mods/DDMODEL00000302/run1.lst", package="nonmem2rx"))
nmlst(system.file("mods/DDMODEL00000301/run3.lst", package="nonmem2rx"))
nmlst(system.file("mods/cpt/runODE032.res", package="nonmem2rx"))
```

`nmtab` *Read nonmem table file*

Description

Read nonmem table file

Usage

```
nmtab(file, ...)
```

Arguments

`file`	file name to read the results from
`...`	other parameters passed to `data.table::fread`

Value

data frame of the read table

Author(s)

Philip Delff, Matthew L. Fidler

Examples

```
nmtab(system.file("mods/cpt/runODE032.csv", package="nonmem2rx"))
```

`nmxml` *Read a nonmem xml and create output similar to the* `nmlst()`

Description

Read a nonmem xml and create output similar to the `nmlst()`

Usage

```
nmxml(xml)
```

Arguments

`xml`	xml file

Value

list of nonmem information

Author(s)

Matthew L. Fidler

Examples

```
nmxml(system.file("mods/cpt/runODE032.xml", package="nonmem2rx"))
```

`nonmem2rx` *Convert a NONMEM source file to a rxode model (nlmixr2-syle)*

Description

Convert a NONMEM source file to a rxode model (nlmixr2-syle)

Usage

```
nonmem2rx(
  file,
  inputData = NULL,
  nonmemOutputDir = NULL,
  rename = NULL,
  tolowerLhs = TRUE,
  thetaNames = TRUE,
  etaNames = TRUE,
  cmtNames = TRUE,
  updateFinal = TRUE,
  determineError = TRUE,
  validate = getOption("nonmem2rx.validate", TRUE),
  nonmemData = FALSE,
  strictLst = FALSE,
  unintFixed = FALSE,
  extended = getOption("nonmem2rx.extended", FALSE),
  nLinesPro = 20L,
  delta = 1e-04,
  usePhi = TRUE,
  useExt = TRUE,
  useCov = TRUE,
  useXml = TRUE,
  useLst = TRUE,
  mod = ".mod",
  cov = ".cov",
  phi = ".phi",
  lst = getOption("nonmem2rx.lst", ".lst"),
  xml = ".xml",
  ext = ".ext",
  grd = ".grd",
  scanLines = getOption("nonmem2rx.scanLines", 50L),
  save = getOption("nonmem2rx.save", NA),
  saveTime = getOption("nonmem2rx.saveTime", 15),
  overwrite = getOption("nonmem2rx.overwrite", TRUE),
  load = getOption("nonmem2rx.load", TRUE),
  compress = getOption("nonmem2rx.compress", TRUE),
  keep = getOption("nonmem2rx.keep", c("dfSub", "dfObs", "thetaMat", "sigma"))
)
```

Arguments

`file`	NONMEM run file, like an `.xml` or `.lst` file or even a control stream
`inputData`	this is a path to the input dataset (or `NULL` to determine from the dataset). Often the input dataset may be different from the place it points to in the control stream because directories can be created to run NONMEM from a script. Because of this, when this is specified the input data will be assumed to be from here instead.
`nonmemOutputDir`	This is a path the the nonmem output directory. When not `NULL` it will assume that the diretory for the output files is located here instead of where the control stream currently exists.
`rename`	When not `NULL` this should be a named character vector that contains the parameters that should be renamed. For example, if the model uses the variable `YTYPE` and has `CMT` it isn't compatible with `rxode2/nlmixr2`. You can change this for the input dataset and the model to create a new model that still reproduces the NONMEM output by specifying `rename=c(dvid="YTYPE")`
`tolowerLhs`	Boolean to change the lhs to lower case (default: `TRUE`)
`thetaNames`	this could be a boolean indicating that the theta names should be changed to the comment-labeled names (default: `TRUE`). This could also be a character vector of the theta names (in order) to be replaced.
`etaNames`	this could be a boolean indicating that the eta names should be changed to the comment-labeled names (default: `TRUE`). This could also be a character vector of the theta names (in order) to be replaced.
`cmtNames`	this could be a boolean indicating that the compartment names should be changed to the named compartments in the `$MODEL` by `COMP = (name)` (default: `TRUE`). This could also be a character vector of the compartment names (in order) to be replaced.
`updateFinal`	Update the parsed model with the model estimates from the `.lst` output file.
`determineError`	Boolean to try to determine the `nlmixr2`-style residual error model (like `ipred ~ add(add.sd)`), otherwise endpoints are not defined in the `rxode2/nlmixr2` model (default: `TRUE`)
`validate`	Boolean that this tool will attempt to "validate" the model by solving the derived model under pred conditions (etas are zero and eps values are zero)
`nonmemData`	Boolean that tells `nonmem2rx` to read in the nonmem data (if possible) even if the model will not be validated (like if it is a simulation run or missing final parameter estimates). By default this is `FALSE`, nonmem data will not be integrated into the nonmem2rx ui.
`strictLst`	The list parsing needs to be correct for a successful load (default `FALSE`).
`unintFixed`	Treat uninteresting values as fixed parameters (default `FALSE`)
`extended`	Translate extended control streams from tools like wings for NONMEM
`nLinesPro`	The number of lines to check for the $PROBLEM statement.
`delta`	this is the offset for NONMEM times that are tied
`usePhi`	if present, use the NONMEM phi file to extract etas (default `TRUE`), otherwise defaults to etas in the tables (if present)

`useExt`	if present, use the NONMEM ext file to extract parameter estimates (default `TRUE`), otherwise defaults to parameter estimates extracted in the NONMEM output
`useCov`	if present, use the NONMEM cov file to import the covariance, otherwise import the covariance with list file
`useXml`	if present, use the NONMEM xml file to import much of the NONMEM information
`useLst`	if present, use the NONMEM lst file to extract NONMEM information
`mod`	the NONMEM output extension, defaults to `.mod`
`cov`	the NONMEM covariance file extension, defaults to `.cov`
`phi`	the NONMEM eta/phi file extension, defaults to `.phi`
`lst`	the NONMEM output extension, defaults to `.lst`
`xml`	the NONMEM xml file extension , defaults to `.xml`
`ext`	the NONMEM ext file extension, defaults to `.ext`
`grd`	the NONMEM gradient file extension, defaults to `.grd`
`scanLines`	number of lines to scan for comment chars when `IGNORE=@`, default is 50
`save`	This can be: • a `NULL` (meaning don't save), • a logical (default `FALSE`, don't save) that when `TRUE` will use the base name of the control stream, append `.qs` and save the file using `qs2::qs_save()` • A path to a file to write Note that this file will be saved with qs2::qs_save() and can be loaded with qs2::qs_read() • A `NA` value which means save if the whole process (including validation) takes too much time
`saveTime`	The time that the translation/validation needs (in secs) before it will save to avoid having to rerun the model (default 15 for 15 seconds)
`overwrite`	is a boolean to allow overwriting the save file (see `load` for more information).
`load`	a boolean that says to load the save file (if it exists) instead of re-running the translation and validation. Note if `overwrite=TRUE` and `load=TRUE` then this will overwrite based on time stamp of the files. If the save file is newer than the input file, then load that file, otherwise regenerate and overwrite. This works best if you point to an output file, like a `.xml` or listing file instead of the control stream
`compress`	a boolean indicating if the UI should be a compressed UI. If you are using this for simulation with old versions of rxode2, the compressed ui is not supported, so this should be `FALSE`. Otherwise use `TRUE` if you are using a newer rxode2.
`keep`	is a character vector of imported model items that are kept in the model itself; The defaults is "sigma" which keeps the sigma matrix in the model itself. You can add rxode2 solving options that are imported from NONMEM to keep in the model.

Details

Since some of these options you may want to set per project, the following options are queried:

- `nonmem2rx.validate` - boolean to validate the model (default: `TRUE`)
- `nonmem2rx.lst` - default extension for output (default: `.lst`)
- `nonmem2rx.save` - should nonmem2rx save the model output?
- `nonmem2rx.overwrite` - should nonmem2rx save output be overwritten (default `TRUE`)
- `nonmem2rx.load` - should nonmem2rx load a saved model instead of translating and validating again? (default `TRUE`)
- `nonmem2rx.extended` - should nonmem2rx support extended control streams? (default `FALSE`)
- `nonmem2rx.compress` - should the ui be compressed or uncompressed (default: `TRUE`)

Value

rxode2 function

Examples

```
# You can run a translation without validating the input.  This is
# a faster way to import a dataset (and allows the CRAN machines to
# run a quick example)

mod <- nonmem2rx(system.file("mods/cpt/runODE032.ctl", package="nonmem2rx"), lst=".res",
                save=FALSE, validate=FALSE, compress=FALSE)



# Though by default you likely wish to validate the input

mod <- nonmem2rx(system.file("mods/cpt/runODE032.ctl", package="nonmem2rx"),
                 lst=".res", save=FALSE)

mod

# you can plot to compare the pred/ipred differences

plot(mod)

# if you want to see the individual differences
# you can by plotting by page of plots

plot(mod, nrow=2, ncol=2, page=1, log="y")

# or select which pages you want to print

plot(mod, nrow=2, ncol=2, page=c(1,3), log="y")


#' or even  all the individuals with
# plot(page=TRUE)
```

```
plot(mod, nrow=5, ncol=5, page=TRUE, log="y")

# you can also convert to a nlmixr2 object, but need babelmixr2 for
# that conversion
```

Index