

Package ‘babelmixr2’

May 28, 2023

Type Package

Title Use 'nlmixr2' to Interact with Open Source and Commercial Software

Version 0.1.1

Description Run other estimation and simulation software via the 'nlmixr2' (Fidler et al (2019) <[doi:10.1002/psp4.12445](https://doi.org/10.1002/psp4.12445)>) interface including 'PKNCA', 'NONMEM' and 'Monolix'. While not required, you can get/install the 'lixoftConnectors' package in the 'Monolix' installation, as described at the following url <https://monolix.lixoft.com/monolix-api/lixoftconnectors_installation/>. When 'lixoftConnectors' is available, 'Monolix' can be run directly instead of setting up command line usage.

License GPL (>= 3)

URL <https://nlmixr2.github.io/babelmixr2/>,
<https://github.com/nlmixr2/babelmixr2/>

NeedsCompilation yes

Encoding UTF-8

Suggests testthat, nlmixr2data, withr, lixoftConnectors, PKNCA (>= 0.10.0), knitr, rmarkdown, spelling, units

Depends R (>= 3.5), nlmixr2 (>= 2.0.8)

Imports checkmate, cli, digest, lotri, nlmixr2est, nonmem2rx, methods, qs, rex, rxode2

RoxygenNote 7.2.3

Config/testthat/edition 3

LinkingTo Rcpp, rxode2, RcppArmadillo, RcppEigen, rxode2parse

Language en-US

VignetteBuilder knitr

Author Matthew Fidler [aut, cre] (<<https://orcid.org/0000-0001-8538-6691>>),
Bill Denney [aut] (<<https://orcid.org/0000-0002-5759-428X>>),
Nook Fulloption [ctb] (goldfish art)

Maintainer Matthew Fidler <matt.fidler@novartis.com>

Repository CRAN

Date/Publication 2023-05-27 22:40:05 UTC

R topics documented:

as.nlmixr2	2
bblDatToMonolix	4
getStandardColNames	6
modelUnitConversion	6
monolixControl	7
nlmixr2Est.pknca	10
nonmemControl	11
pkncaControl	14
rxToMonolix	15
rxToNonmem	15
simplifyUnit	16
Index	17

as.nlmixr2	<i>Convert an object to a nlmixr2 fit object</i>
------------	--

Description

Convert an object to a nlmixr2 fit object

Usage

```
as.nlmixr2(x, ..., table = nlmixr2est::tableControl())
```

```
as.nlmixr(x, ..., table = nlmixr2est::tableControl())
```

Arguments

x	Object to convert
...	Other arguments
table	is the 'nlmixr2est::tableControl()' options

Value

nlmixr2 fit object

Author(s)

Matthew L. Fidler

Examples

```

# First read in the model (but without residuals)
mod <- nonmem2rx(system.file("mods/cpt/runODE032.ct1", package="nonmem2rx"),
                 determineError=FALSE, lst=".res", save=FALSE)

# define the model with residuals (and change the name of the
# parameters) In this step you need to be careful to not change the
# estimates and make sure the residual estimates are correct (could
# have to change var to sd).

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)
  })
}

# now we create another nonmem2rx object that validates the model above:

new <- as.nonmem2rx(mod2, mod)

# once that is done, you can translate to a full.nlmixr2 fit (if you wish)

fit <- as.nlmixr2(new)

print(fit)

```

bbIDatToMonolix *Convert nlmixr2-compatible data to other formats (if possible)*

Description

Convert nlmixr2-compatible data to other formats (if possible)

Usage

```
bbIDatToMonolix(model, data, table = nlmixr2est::tableControl(), env = NULL)
```

```
bbIDatToNonmem(model, data, table = nlmixr2est::tableControl(), env = NULL)
```

```
bbIDatToRxode(model, data, table = nlmixr2est::tableControl(), env = NULL)
```

```
bbIDatToMrgsolve(model, data, table = nlmixr2est::tableControl(), env = NULL)
```

```
bbIDatToPknca(model, data, table = nlmixr2est::tableControl(), env = NULL)
```

Arguments

model	rxode2 model for conversion
data	Input dataset.
table	is the table control; this is mostly to figure out if there are additional columns to keep.
env	When 'NULL' (default) nothing is done. When an environment, the function 'nlmixr2est::foceiPreProcessData(data, env, model)' is called on the provided environment.

Value

With the function 'bbIDatToMonolix()' return a list with: - Monolix compatible dataset (\$monolix)
- Monolix ADM information (\$adm)

With the function 'nlmixrDataToNonmem()' return a dataset that is compatible with NONMEM.

With the function 'nlmixrDataToMrgsolve()' return a dataset that is compatible with 'mrgsolve'. Unlike NONMEM, it supports replacement events with 'evid=8' (note with 'rxode2' replacement 'evid' is '5').

With the function 'nlmixrDataToRxode()' this will normalize the dataset to use newer 'evid' definitions that are closer to NONMEM instead of any classic definitions that are used at a lower level

Author(s)

Matthew L. Fidler

Examples

```

pk.turnover.emax3 <- function() {
  ini({
    tktr <- log(1)
    tka <- log(1)
    tc1 <- log(0.1)
    tv <- log(10)
    ##
    eta.ktr ~ 1
    eta.ka ~ 1
    eta.cl ~ 2
    eta.v ~ 1
    prop.err <- 0.1
    pkadd.err <- 0.1
    ##
    temax <- logit(0.8)
    tec50 <- log(0.5)
    tkout <- log(0.05)
    te0 <- log(100)
    ##
    eta.emax ~ .5
    eta.ec50 ~ .5
    eta.kout ~ .5
    eta.e0 ~ .5
    ##
    pdadd.err <- 10
  })
  model({
    ktr <- exp(tktr + eta.ktr)
    ka <- exp(tka + eta.ka)
    cl <- exp(tc1 + eta.cl)
    v <- exp(tv + eta.v)
    emax = expit(temax+eta.emax)
    ec50 = exp(tec50 + eta.ec50)
    kout = exp(tkout + eta.kout)
    e0 = exp(te0 + eta.e0)
    ##
    DCP = center/v
    PD=1-emax*DCP/(ec50+DCP)
    ##
    effect(0) = e0
    kin = e0*kout
    ##
    d/dt(depot) = -ktr * depot
    d/dt(gut) = ktr * depot -ka * gut
    d/dt(center) = ka * gut - cl / v * center
    d/dt(effect) = kin*PD -kout*effect
    ##
    cp = center / v
    cp ~ prop(prop.err) + add(pkadd.err)
    effect ~ add(pdadd.err) | pca
  })
}

```

```
  })  
}  
  
bblDatToMonolix(pk.turnover.emax3, nlmixr2data::warfarin)  
  
bblDatToNonmem(pk.turnover.emax3, nlmixr2data::warfarin)  
  
bblDatToMrgsolve(pk.turnover.emax3, nlmixr2data::warfarin)  
  
bblDatToRxode(pk.turnover.emax3, nlmixr2data::warfarin)
```

`getStandardColNames` *Determine standardized rxode2 column names from data*

Description

Determine standardized rxode2 column names from data

Usage

```
getStandardColNames(data)
```

Arguments

`data` A data.frame as the source for column names

Value

A named character vector where the names are the standardized names and the values are either the name of the column from the data or NA if the column is not present in the data.

Examples

```
getStandardColNames(data.frame(ID=1, DV=2, Time=3, CmT=4))
```

`modelUnitConversion` *Unit conversion for pharmacokinetic models*

Description

Unit conversion for pharmacokinetic models

Usage

```
modelUnitConversion(  
  dvu = NA_character_,  
  amtu = NA_character_,  
  timeu = NA_character_,  
  volumeu = NA_character_  
)
```

Arguments

dvu, amtu, timeu The units for the DV, AMT, and TIME columns in the data

volumeu The units for the volume parameters in the model

Value

A list with names for the units associated with each parameter ("amtu", "clearanceu", "volumeu", "timeu", "dvu") and the numeric value to multiply the modeled estimate (for example, cp) so that the model is consistent with the data units.

See Also

Other Unit conversion: [simplifyUnit\(\)](#)

Examples

```
modelUnitConversion(dvu = "ng/mL", amtu = "mg", timeu = "hr", volumeu = "L")
```

monolixControl

Monolix Controller for nlmixr2

Description

Monolix Controller for nlmixr2

Usage

```
monolixControl(  
  nbSSDoses = 7,  
  useLinearization = FALSE,  
  stiff = FALSE,  
  addProp = c("combined2", "combined1"),  
  exploratoryAutoStop = FALSE,  
  smoothingAutoStop = FALSE,  
  burnInIterations = 5,  
  smoothingIterations = 200,  
  exploratoryIterations = 250,
```

```

simulatedAnnealingIterations = 250,
exploratoryInterval = 200,
exploratoryAlpha = 0,
omegaTau = 0.95,
errorModelTau = 0.95,
variability = c("none", "firstStage", "decreasing"),
runCommand = getOption("babelmixr2.monolix", ""),
rxControl = NULL,
sumProd = FALSE,
optExpression = TRUE,
calcTables = TRUE,
compress = TRUE,
ci = 0.95,
sigdigTable = NULL,
absolutePath = FALSE,
modelName = NULL,
...
)

```

Arguments

nbSSDoses Number of steady state doses (default 7)

useLinearization
Use linearization for log likelihood and fim.

stiff boolean for using the stiff ODE solver

addProp specifies the type of additive plus proportional errors, the one where standard deviations add (combined1) or the type where the variances add (combined2).
The combined1 error type can be described by the following equation:

$$y = f + (a + b \cdot f^c) \cdot \text{err}$$
The combined2 error model can be described by the following equation:

$$y = f + \sqrt{a^2 + b^2 \cdot (f^c)^2} \cdot \text{err}$$
Where:
- y represents the observed value
- f represents the predicted value
- a is the additive standard deviation
- b is the proportional/power standard deviation
- c is the power exponent (in the proportional case c=1)

exploratoryAutoStop
logical to turn on or off exploratory phase auto-stop of SAEM (default 250)

smoothingAutoStop
Boolean indicating if the smoothing should automatically stop (default 'FALSE')

burnInIterations
Number of burn in iterations

smoothingIterations
Number of smoothing iterations

exploratoryIterations	Number of iterations for exploratory phase (default 250)
simulatedAnnealingIterations	Number of simulating annealing iterations
exploratoryInterval	Minimum number of iterations in the exploratory phase (default 200)
exploratoryAlpha	Convergence memory in the exploratory phase (only used when 'exploratoryAutoStop' is 'TRUE')
omegaTau	Proportional rate on variance for simulated annealing
errorModelTau	Proportional rate on error model for simulated annealing
variability	This describes the methodology for parameters without variability. It could be: - Fixed throughout (none) - Variability in the first stage (firstStage) - Decreasing until it reaches the fixed value (decreasing)
runCommand	is a shell command or function to run monolix; You can specify the default by options("babelmixr2.monolix"="runMonolix"). If it is empty and 'lixoftConnectors' is available, use lixoftConnectors to run monolix. See details for function usage.
rxControl	'rxode2' ODE solving options during fitting, created with 'rxControl()'
sumProd	Is a boolean indicating if the model should change multiplication to high precision multiplication and sums to high precision sums using the PreciseSums package. By default this is FALSE.
optExpression	Optimize the rxode2 expression to speed up calculation. By default this is turned on.
calcTables	This boolean is to determine if the fociFit will calculate tables. By default this is TRUE
compress	Should the object have compressed items
ci	Confidence level for some tables. By default this is 0.95 or 95% confidence.
sigdigTable	Significant digits in the final output table. If not specified, then it matches the significant digits in the 'sigdig' optimization algorithm. If 'sigdig' is NULL, use 3.
absolutePath	Boolean indicating if the absolute path should be used for the monolix runs
modelName	Model name used to generate the NONMEM output. If 'NULL' try to infer from the model name (could be 'x' if not clear). Otherwise use this character for outputs.
...	Ignored parameters

Details

If runCommand is given as a string, it will be called with the system() command like:

```
runCommand mlxtran.
```

For example, if runCommand="/path/to/monolix/mlxbsub2021' -p " then the command line used would look like the following:

```
 '/path/to/monolix/mlxbsub2021' monolix.mlxtran
```

If `runCommand` is given as a function, it will be called as `FUN(mlxtran, directory, ui)` to run Monolix. This allows you to run Monolix in any way that you may need, as long as you can write it in R. `babelmixr2` will wait for the function to return before proceeding.

If `runCommand` is NA, `nlmixr()` will stop after writing the model files and without starting Monolix.

Value

A monolix control object

Author(s)

Matthew Fidler

nlmixr2Est.pknca	<i>Estimate starting parameters using PKNCA</i>
------------------	---

Description

Estimate starting parameters using PKNCA

Usage

```
## S3 method for class 'pknca'
nlmixr2Est(env, ...)
```

Arguments

<code>env</code>	Environment for the <code>nlmixr2</code> estimation routines. This needs to have: - <code>rxode2</code> ui object in <code>‘\$ui‘</code> - data to fit in the estimation routine in <code>‘\$data‘</code> - control for the estimation routine’s control options in <code>‘\$ui‘</code>
<code>...</code>	Other arguments provided to <code>‘nlmixr2Est()‘</code> provided for flexibility but not currently used inside <code>nlmixr</code>

Details

Parameters are estimated as follows:

- `ka4` half-lives to `Tmax` but not higher than 3: $\log(2)/(t_{max}/4)$
- `vcInverse` of dose-normalized `Cmax`
- `cl` Estimated as the median clearance
- `vp, vp22-` and 4-fold the `vc`, respectively by default, controlled by the `vpMult` and `vp2Mult` arguments to `pkncaControl`

- $q, q_{20.5}$ - and 0.25-fold the c_l , respectively by default, controlled by the `qMult` and `q2Mult` arguments to `pkncaControl`

The bounds for the parameter estimates are set to 10 and 10 times the 99th percentile. (For k_a , the lower bound is set to the lower of 10 modified from 10 times the 99th percentile.)

Parameter estimation methods may be changed in a future version.

Value

A model with updated starting parameters. In the model a new element named "nca" will be available which includes the PKNCA results used for the calculation.

nonmemControl	<i>NONMEM estimation control</i>
---------------	----------------------------------

Description

NONMEM estimation control

Usage

```
nonmemControl(
  est = c("focei", "imp", "its", "posthoc"),
  advanOde = c("advan13", "advan8", "advan6"),
  cov = c("r,s", "r", "s", ""),
  maxeval = 1e+05,
  tol = 6,
  atol = 12,
  sstol = 6,
  ssatol = 12,
  sigl = 12,
  sigdig = 3,
  print = 1,
  extension = getOption("babelmixr2.nmModelExtension", ".nmctl"),
  outputExtension = getOption("babelmixr2.nmOutputExtension", ".lst"),
  runCommand = getOption("babelmixr2.nonmem", ""),
  iniSigDig = 5,
  protectZeros = TRUE,
  muRef = TRUE,
  addProp = c("combined2", "combined1"),
  rxControl = NULL,
  sumProd = FALSE,
  optExpression = TRUE,
  calcTables = TRUE,
  compress = TRUE,
  ci = 0.95,
  sigdigTable = NULL,
```

```

    readRounding = FALSE,
    readBadOpt = FALSE,
    niter = 100L,
    isample = 1000L,
    iaccept = 0.4,
    iscaleMin = 0.1,
    iscaleMax = 10,
    df = 4,
    seed = 14456,
    mapiter = 1,
    mapinter = 0,
    noabort = TRUE,
    modelName = NULL,
    ...
)

```

Arguments

est	NONMEM estimation method
advanOde	The ODE solving method for NONMEM
cov	The NONMEM covariance method
maxeval	NONMEM's maxeval (for non posthoc methods)
tol	NONMEM tolerance for ODE solving advan
atol	NONMEM absolute tolerance for ODE solving
sstol	NONMEM tolerance for steady state ODE solving
ssatol	NONMEM absolute tolerance for steady state ODE solving
sigl	NONMEM sigl estimation option
sigdig	the significant digits for NONMEM
print	The print number for NONMEM
extension	NONMEM file extensions
outputExtension	Extension to use for the NONMEM output listing
runCommand	Command to run NONMEM (typically the path to "nmfe75") or a function. See the details for more information.
iniSigDig	How many significant digits are printed in \$THETA and \$OMEGA when the estimate is zero. Also controls the zero protection numbers
protectZeros	Add methods to protect divide by zero
muRef	Automatically mu-reference the control stream
addProp, sumProd, optExpression, calcTables, compress, ci, sigdigTable	Passed to nlmixr2est::foceiControl
rxControl	Options to pass to rxode2::rxControl for simulations
readRounding	Try to read NONMEM output when NONMEM terminated due to rounding errors

readBadOpt	Try to read NONMEM output when NONMEM terminated due to an apparent failed optimization
niter	number of iterations in NONMEM estimation methods
isample	Isample argument for NONMEM ITS estimation method
iaccept	Iaccept for NONMEM ITS estimation methods
iscaleMin	parameter for IMP NONMEM method (ISCALE_MIN)
iscaleMax	parameter for IMP NONMEM method (ISCALE_MAX)
df	degrees of freedom for IMP method
seed	is the seed for NONMEM methods
mapiter	the number of map iterations for IMP method
mapinter	is the MAPINTER parameter for the IMP method
noabort	Add the 'NOABORT' option for '\$EST'
modelName	Model name used to generate the NONMEM output. If 'NULL' try to infer from the model name (could be 'x' if not clear). Otherwise use this character for outputs.
...	optional genRxControl argument controlling automatic rxControl generation.

Details

If runCommand is given as a string, it will be called with the system() command like:

```
runCommand controlFile outputFile.
```

For example, if runCommand="" /path/to/nmfe75 "" then the command line used would look like the following:

```
'/path/to/nmfe75' one.cmt.nmctl one.cmt.lst
```

If runCommand is given as a function, it will be called as FUN(ctl, directory, ui) to run NONMEM. This allows you to run NONMEM in any way that you may need, as long as you can write it in R. babelmixr2 will wait for the function to return before proceeding.

If runCommand is NA, nlmixr() will stop after writing the model files and without starting NONMEM.

Value

babelmixr2 control option for generating NONMEM control stream and reading it back into 'babelmixr2'/'nlmixr2'

Author(s)

Matthew L. Fidler

Examples

```
nonmemControl()
```

pkncaControl

*PKNCA estimation control***Description**

PKNCA estimation control

Usage

```

pkncaControl(
  concu = NA_character_,
  doseu = NA_character_,
  timeu = NA_character_,
  volumeu = NA_character_,
  vpMult = 2,
  qMult = 1/2,
  vp2Mult = 4,
  q2Mult = 1/4,
  dvParam = "cp",
  groups = character(),
  sparse = FALSE,
  ncaData = NULL,
  ncaResults = NULL
)

```

Arguments

concu, doseu, timeu	concentration, dose, and time units from the source data (passed to <code>PKNCA::pknca_units_table()</code>).
volumeu	compartment volume for the model (if <code>NULL</code> , simplified units from source data will be used)
vpMult, qMult, vp2Mult, q2Mult	Multipliers for <code>vc</code> and <code>cl</code> to provide initial estimates for <code>vp</code> , <code>q</code> , <code>vp2</code> , and <code>q2</code>
dvParam	The parameter name in the model that should be modified for concentration unit conversions. It must be assigned on a line by itself, separate from the residual error model line.
groups	Grouping columns for NCA summaries by group (required if <code>sparse = TRUE</code>)
sparse	Are the concentration-time data sparse PK (commonly used in small nonclinical species or with terminal or difficult sampling) or dense PK (commonly used in clinical studies or larger nonclinical species)?
ncaData	Data to use for calculating NCA parameters. Typical use is when a subset of the original data are informative for NCA.
ncaResults	Already computed NCA results (a <code>PKNCAresults</code> object) to bypass automatic calculations. At least the following parameters must be calculated in the NCA: <code>tmax</code> , <code>cmax.dn</code> , <code>cl.last</code>

Value

A list of parameters

rxToMonolix	<i>Convert RxODE syntax to monolix syntax</i>
-------------	---

Description

Convert RxODE syntax to monolix syntax

Usage

```
rxToMonolix(x, ui)
```

Arguments

x	Expression
ui	rxode2 ui

Value

Monolix syntax

Author(s)

Matthew Fidler

rxToNonmem	<i>Convert RxODE syntax to NONMEM syntax</i>
------------	--

Description

Convert RxODE syntax to NONMEM syntax

Usage

```
rxToNonmem(x, ui)
```

Arguments

x	Expression
ui	rxode2 ui

Value

NONMEM syntax

Author(s)

Matthew Fidler

simplifyUnit	<i>Simplify units by removing repeated units from the numerator and denominator</i>
--------------	---

Description

Simplify units by removing repeated units from the numerator and denominator

Usage

```
simplifyUnit(numerator = "", denominator = "")
```

Arguments

numerator	The numerator of the units (or the whole unit specification)
denominator	The denominator of the units (or NULL if numerator is the whole unit specification)

Details

NA or "" for numerator and denominator are considered unitless.

Value

The units specified with units that are in both the numerator and denominator cancelled.

See AlsoOther Unit conversion: [modelUnitConversion\(\)](#)**Examples**

```
simplifyUnit("kg", "kg/mL")  
# units that don't match exactly are not cancelled  
simplifyUnit("kg", "g/mL")
```


Index

* Unit conversion

- modelUnitConversion, 6
- simplifyUnit, 16

- as.nlmixr (as.nlmixr2), 2
- as.nlmixr2, 2

- bb1DatToMonolix, 4
- bb1DatToMrgsolve (bb1DatToMonolix), 4
- bb1DatToNonmem (bb1DatToMonolix), 4
- bb1DatToPknca (bb1DatToMonolix), 4
- bb1DatToRxode (bb1DatToMonolix), 4

- getStandardColNames, 6

- modelUnitConversion, 6, 16
- monolixControl, 7

- nlmixr2Est.pknca, 10
- nonmemControl, 11

- pkncaControl, 14

- rxToMonolix, 15
- rxToNonmem, 15

- simplifyUnit, 7, 16