Package: pksensi (via r-universe)

October 21, 2026

Title Global Sensitivity Analysis in Physiologically Based Kinetic Modeling

Version 1.4.0.9000

Description Applying the global sensitivity analysis workflow to investigate the parameter uncertainty and sensitivity in physiologically based kinetic (PK) models, especially the physiologically based pharmacokinetic/toxicokinetic model with multivariate outputs. The package also provides some functions to check the convergence and sensitivity of model parameters. The workflow was first mentioned in Hsieh et al., (2018) <doi:10.3389/fphar.2018.00588>, then further refined (Hsieh et al., 2020 <doi:10.1016/j.softx.2020.100609>).

License GPL-3 | file LICENSE

URL https://github.com/nanhung/pksensi, https://nanhung.github.io/pksensi/

BugReports https://github.com/nanhung/pksensi/issues

Depends R (>= 4.1)

Imports ggplot2, data.table, deSolve, foreach, parallel, doParallel

Suggests covr, knitr, httk, rmarkdown, testthat

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Language en-US

Roxygen list(markdown = TRUE)

Repository https://nanhung.r-universe.dev

Date/Publication 2024-11-27 17:31:55 UTC

RemoteUrl https://github.com/nanhung/pksensi

RemoteRef HEAD

RemoteSha a775203eab90fa09059fde7190396d730ae04351

Contents

about-pksensi 2
APAP 3
check 3
compile_model 6
mcsim_install 7
models 8
pksim 9
rfast99 9
solve_fun 11
solve_mcsim 12
tell2 15

Index **16**

`about-pksensi` *About* **pksensi** *package*

Description

Applying a global sensitivity analysis approach to reduce parameter dimensionality in physiologically based kinetic modeling and evaluate the robustness of the algorithm under the given sampling number.

Details

The "extended Fourier amplitude sensitivity testing (eFAST)", a variance-based sensitivity analysis method is used to estimate the parameter impact on model output (Saltelli et al. 1999). The eFAST is the effective algorithm to determine the influential parameter in physiologically-based pharmacokinetic model calibration (Hsieh et al. 2018). The eFAST algorithm is sourced from **sensitivity** package but implemented the random-phase shift to evaluating the robustness of sensitivity measurement under the given sample size.

References

Saltelli, A., Tarantola, S., & Chan, K. S. (1999). A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics*, 41, 39-56.

Hsieh, N. H., Reisfeld, B., Bois, F. Y., & Chiu, W. A. (2018). Applying a global sensitivity analysis workflow to improve the computational efficiencies in physiologically-based pharmacokinetic modeling. *Frontiers in Pharmacology*, 9, 588.

Hsieh, N-H., Reisfeld B., and Chiu W.A., (2020). pksensi: An R package to apply global sensitivity analysis in physiologically based kinetic modeling *SoftwareX*, 12, 100609.

APAP *Pharmacokinetic Dataset of Acetaminophen*

Description

The Acetaminophen dataset contains a human experiment of pharmacokinetic data for acetaminophen and it's major metabolites acetaminophen-glucuronide and acetaminophen-sulfate.

Usage

```
APAP
```

Format

A data frame containing the following columns:

- Wt: averaged weight of the study population (kg).
- Dose: given oral dose of acetaminophen administered (mg/kg).
- Time: time after drug administration (hr).
- APAP: concentration of acetaminophen in the sample (mg/L).
- AG: concentration of acetaminophen-glucuronide in the sample (mg/L).
- AS: concentration of acetaminophen-sulfate in the sample (mg/L).
- Study: sourced reference.

References

Zurlinden, T. J., & Reisfeld, B. (2016). Physiologically based modeling of the pharmacokinetics of acetaminophen and its major metabolites in humans using a Bayesian population approach. *European journal of drug metabolism and pharmacokinetics*, 41(3), 267-280.

check *Check the Parameter Sensitivity*

Description

Visualize and check the sensitivity (or convergence) measurement with the given result.

Usage

```
check(x, times, vars, cutoff, out)

heat_check(
  x,
  order = c("first order", "interaction", "total order"),
  vars = NULL,
  times = NULL,
  index = "SI",
  cutoff = c(0.05, 0.1),
  level = T,
  text = F,
  show.all = FALSE
)

## S3 method for class 'rfast99'
plot(x, vars = 1, cutoff = 0.1, ...)

## S3 method for class 'rfast99'
print(x, ...)
```

Arguments

x	a list of the storing information in the defined sensitivity function.
times	a logical value or character to specify the display time in simulation.
vars	a logical value or character to specify the display variable in simulation.
cutoff	a value or vector to set the cut-off for sensitivity (or convergence) index. The default is 0.05 and 0.1.
out	a logical value to print the checking result to the console.
order	a vector of the interested output index, including `first order`, `interaction`, and `total order`.
index	a character to choose sensitivity index `SI` (default) or convergence index `CI`.
level	a logical value to use continuous or discrete (default) output.
text	a logical value to display the calculated indices in the plot.
show.all	a logical value to show all testing parameters in the heatmap. The default is set to `FALSE` to show only the influential parameters.
...	additional arguments to customize the graphical parameters.

Details

The convergence of sensitivity indices for each parameter is using the approach proposed by Sarrazin et al. (2016). This method quantitatively assesses the convergence by computing the range of 95\ Using the global approach based on the heatmap visualization combined with the index "cut-off," can systematically distinguish between "influential" and "non-influential" parameters (Hsieh et al. 2018).

Value

The `print` function returns sensitivity and convergence indices with given time-step in the console. The `check` method provides the summary of parameter sensitivity and convergence according to the given `cutoff`. It can distinguish the influential and non-influential parameter by the providing value of `cutoff`. The `plot` function can generate the time-course functional outputs of first order and interaction indices for each parameter. The default output is the first model variable. The `heat_check` provides a convenient way to visualize and distinguish the influential and non-influential parameter by the setting cut-off. The convergence index can examine the stability of the sensitivity index. To check convergence, be sure to conduct the replication in `rfast99`.

References

Sarrazin, F., Pianosi, F., & Wagener, T. (2016). Global Sensitivity Analysis of environmental models: Convergence and validation. *Environmental Modelling & Software*, 79, 135–152.

Hsieh, N. H., Reisfeld, B., Bois, F. Y., & Chiu, W. A. (2018). Applying a global sensitivity analysis workflow to improve the computational efficiencies in physiologically-based pharmacokinetic modeling. *Frontiers in Pharmacology*, 9, 588.

See Also

`tell2`

Examples

```
q <- "qunif"
q.arg <- list(list(min = 0.6, max = 1),
  list(min = 0.5, max = 1.5),
  list(min = 0.02, max = 0.3),
  list(min = 20, max = 60))

params <- c("F","KA","KE","V")

set.seed(1234)
x <- rfast99(params = params, n = 200, q = q, q.arg = q.arg, rep = 20)

time <- c(0.25, 0.5, 1, 2, 4, 6, 8, 12, 24)
out <- solve_fun(x, model = FFPK, time = time, vars = "output")


# Check results of sensitivity measures
check(out)
plot(out)
heat_check(out, show.all = TRUE)
heat_check(out, index = "CI")
```

`compile_model` *Model Compiler*

Description

The `compile_model` is used to compile the model code that is written under C or **GNU MCSim** format and generate the executable program in numerical analysis.

Usage

```
compile_model(
  mName,
  application = "mcsim",
  use_model_file = TRUE,
  mcsim_dir = NULL,
  mod_dir = NULL
)
```

Arguments

`mName`	a string giving the name of the model code (without extension).
`application`	a character to assign the specific methods (`mcsim` or `R`) that will be applied to the numerical analysis (default is `mcsim`). The version must be assigned for Windows user (default is `6.2.0`).
`use_model_file`	a logical value to operate the compiler to model or C file, the default is set to `TRUE` to assign the **GNU MCSim**'s model file in compiling.
`mcsim_dir`	a character to assign the installed location of MCSim directory (The default is set to home).
`mod_dir`	a character giving the name of the directory that is used to store the model file.

Details

Generally, the solving function through **GNU MCSim** can provide faster computing speed than exporting C in R. Therefore, this function set `use_model_file = TRUE` and `application = 'mcsim'` as a default setting, suggesting to use **GNU MCSim** as main solver to solve the differential equation. To compile model code in Windows, be sure to install Rtools (rtools40) first. In addition, the `version` of **GNU MCSim** should provide to conduct model compiling in Windows.

Value

The default `application` is set to `'mcsim'` to generate the executable program to solve differential equations by **GNU MCSim**. If `application = 'R'`, the function will compile and create dynamic-link libraries (.dll) on Windows and shared objects (.so) on Unix-likes systems (e.g., Linux and MacOS) that can link with **deSolve** solver.

`mcsim_install` *Download and install* **GNU MCSim**

Description

Download the latest or specific version of **GNU MCSim** from the official website (https://www.gnu.org/software/mcsim/) and install it to the system directory.

Usage

```
mcsim_install(version = "6.2.0", install_dir = NULL, mxstep = 5000)

mcsim_version()

set_rtools40_path()
```

Arguments

`version`	a character of version number.
`install_dir`	a character to assign the installed directory.
`mxstep`	a numeric value to assign the maximum number of (internally defined) steps allowed during one call to the solver.

Details

This function aims to help users download (source: https://ftp.gnu.org/gnu/mcsim/) and install **GNU MCSim** more easily. However, if you can not install it through this function. The additional way is to follow the instruction and install it manually: https://www.gnu.org/software/mcsim/mcsim.html#Installation

The default `mxstp` is setting to 5000. The user can increase `mxstp` to avoid possible error return. If you meet any error when conduct sensitivity analysis, you can use this function to re-install **GNU MCSim** and set the higher `mxstp`. The default installed `directory` is under `/home/username` (Linux), `/Users/username` (MacOS), and `C:/Users/username/Documents` (windows). To install **GNU MCSim** in Windows, be sure to install Rtools first. The current suggested Rtools version is 4.0.

Functions

- `mcsim_version()`: Return the version number.
- `set_rtools40_path()`: the function to set Rtools40 path

References

Bois, F. Y., & Maszle, D. R. (1997). MCSim: a Monte Carlo simulation program. *Journal of Statistical Software*, 2(9): 1–60.

https://www.gnu.org/software/mcsim/

`models` *Example PK Model for Sensitivity Analysis*

Description

The example PK model that were used in sensitivity testing. Three examples are included: Flip-flop pharmacokinetic model, one-compartment toxicokinetic model from **httk** (Pearce et al. 2017), and acetaminophen pharmacokinetic model (Zurlinden et al. 2016).

Usage

```
FFPK(params, time, dose = 1)

pbtk1cpt_model()

pbpk_apap_model()
```

Arguments

`params`	a numeric vector to define the input parameter value.
`time`	a numeric vector to define the given time point(s).
`dose`	a numeric value to define the given dose in flip-flop model.

Functions

- `pbtk1cpt_model()`: Download pbtk1cpt.model file.
- `pbpk_apap_model()`: Download pbpk_apap.model file.

References

Pearce, R. G., Setzer, R. W., Strope, C. L., Wambaugh, J. F., & Sipes, N. S. (2017). httk: R package for high-throughput toxicokinetics. *Journal of Statistical Software*, 79(4), 1-26.

Zurlinden, T. J., & Reisfeld, B. (2016). Physiologically based modeling of the pharmacokinetics of acetaminophen and its major metabolites in humans using a Bayesian population approach. *European journal of drug metabolism and pharmacokinetics*, 41(3), 267-280.

Examples

```
params <- c(F = 0.9, KA = 1.2, KE = 0.2, V = 1.5)
t <- seq(0, 12, 0.1)
C <-FFPK(params = params, time = t)
plot(t, C, type = "l", xlab = "time", ylab = "concentration")
```

pksim *PK Simulation from Sampling Parameter*

Description

Create the PK profile of the output results based on the given parameter (uncertainty analysis). This function is used to visualize the corresponding output PK profile from the input parameters in sensitivity analysis. If there are multiple outputs in model, the generated result will be the first model variable (default). A PK plot is generated with median, 25\

Usage

```
pksim(y, vars = 1, log = F, legend = T, ...)
```

Arguments

y	a numeric array created from solve_fun or solve_mcsim function.
vars	a logical value or character to specific the display variable in simulation (default 1).
log	a logical value to transform the y-axis to log scale.
legend	a logical value to display the legend in the created plot.
...	additional arguments to customize the graphical parameters.

rfast99 *Extended Fourier Amplitude Sensitivity Test with Random Phase Shift*

Description

Applying the extended Fourier amplitude sensitivity Test algorithm to create the numeric sequences for each parameter (Saltelli et al. 1999). Each sequence is random generated based on the random phase shift approach. It is an extension based on the fast99 function in **sensitivity** package.

Usage

```
rfast99(
  params,
  n,
  M = 4,
  omega = NULL,
  q = NULL,
  q.arg = NULL,
  replicate = 5,
  conf = 0.95
)
```

Arguments

params	an integer for the giving number of parameters, or a vector of character strings giving their names.
n	an integer for the sampling size.
M	an integer specifying the interference parameter. The default is 4.
omega	a vector giving the set of frequencies.
q	a vector of quantile functions names corresponding to wanted parameters distributions.
q.arg	a list of quantile functions parameters.
replicate	an integer to define the number of replication. The default is 5.
conf	the confidence level for replication confidence intervals. The default is 0.95.

Value

The returned parameter value will be stored in an array with c(model evaluation, replication, parameters).

References

Saltelli, A., Tarantola, S., & Chan, K. S. (1999). A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics*, 41, 39-56.

Cukier, R. I., Levine, H. B., & Shuler, K. E. (1978). Nonlinear sensitivity analysis of multiparameter model systems. *Journal of Computational Physics*, 26, 1–42.

Examples

```
# Generate the parameter matrix with 20 replications
q <- "qunif"
q.arg <- list(min = 0, max = 1)

set.seed(1234)
x <- rfast99(params = 3, n = 100, replicate = 20, q = q, q.arg = q.arg)
dim(x$a) # the array of c(model evaluation, replication, parameters).

## Not run:
save(x, file = "input_parameters.rda")

## End(Not run)
```

solve_fun *Solve PK Model Through* **deSolve** *Package or Analytical Function*

Description

Solve time-dependent quantities/concentrations of different variables in PK model through the imported ode function in **deSolve** package. It can also be used to solve the function with analytical solution.

Usage

```
solve_fun(
  x,
  time = NULL,
  initParmsfun = "initParms",
  initState,
  dllname = NULL,
  func = "derivs",
  initfunc = "initmod",
  outnames,
  method = "lsode",
  rtol = 1e-08,
  atol = 1e-12,
  model = NULL,
  lnparam = F,
  vars = NULL,
  tell = T,
  ...
)
```

Arguments

x	a list of storing information in the defined sensitivity function.
time	a vector to define the given time sequence.
initParmsfun	a character for the given specific initial parameter function.
initState	a vector that define the initial values of state variables for the ODE system.
dllname	a string giving the name of the shared library (without extension) that contains the compiled function.
func	the name of the function in the dynamically loaded shared library.
initfunc	the name of the initialization function (which initialises values of parameters), as provided in dllname.
outnames	the names of output variables calculated in the compiled function func.
method	method used by integrator (**deSolve**).
rtol	argument passed to integrator (**deSolve**).

atol	argument passed to integrator (**deSolve**).
model	the defined analytical equation with functional output.
lnparam	a logical value that make the statement of the log-transformed parameter (default FALSE).
vars	a character for the selected output.
tell	a logical value to automatically combine the result y to decoupling simulation x.
...	additional arguments for deSolve::ode method.

References

Soetaert, K. E., Petzoldt, T., & Setzer, R. W. (2010). Solving differential equations in R: package deSolve. *Journal of Statistical Software*, 33(9), 1–25.

See Also

pksim

Examples

```
q <- "qunif"
q.arg <- list(list(min = 0.6, max = 1.0),
 list(min = 0.5, max = 1.5),
 list(min = 0.02, max = 0.3),
 list(min = 20, max = 60))

params <- c("F","KA","KE","V")

set.seed(1234)
x <- rfast99(params = params, n = 200, q = q, q.arg = q.arg, rep = 20)

time <- seq(from = 0.25, to = 12.25, by = 0.5)
y <- solve_fun(x, model = FFPK, time = time, vars = "output")

pksim(y) # Visualize uncertainty of model output
```

---

solve_mcsim *Solve PK Model Through* **GNU MCSim**

Description

Solve the differential equations of time-dependent quantity/concentration in different tissues/compartments through **GNU MCSim**.

Usage

```
solve_mcsim(
  x,
  mName,
  infile.name = NULL,
  outfile.name = NULL,
  setpoint.name = NULL,
  params = NULL,
  vars = NULL,
  times = NULL,
  condition = NULL,
  generate.infile = T,
  tell = T,
  rtol = 1e-06,
  atol = 1e-06,
  monte_carlo = NULL,
  dist = NULL,
  q.arg = NULL,
  parallel = 1
)

generate_infile(
  mod = NULL,
  infile.name = NULL,
  outfile.name = NULL,
  params = NULL,
  vars,
  times,
  condition,
  rtol = 1e-06,
  atol = 1e-06,
  monte_carlo = NULL,
  dist = NULL,
  q.arg = NULL
)
```

Arguments

x	a list of storing information in the defined sensitivity function.
mName	a string giving the name of the model or C file (without extension).
infile.name	a character to assign the name of input file.
outfile.name	a character to assign the name of output file.
setpoint.name	a character to assign the name of file for parameter matrix.
params	a character to assign the testing parameters.
vars	a character or a vector to assign the selected output(s).
times	a numeric vector to define the given time point(s).

condition	a character to set the specific parameter value in the input file.
generate.infile	a logical value to automatically generate the input file.
tell	a logical value to automatically combine the result y to decoupling simulation x.
rtol	an argument passed to the integrator (default 1e-6).
atol	an argument passed to the integrator (default 1e-6).
monte_carlo	a numeric value to define the sample size in Monte Carlo simulation.
dist	a vector of distribution names corresponding to <distribution-name> in **GNU MCSim**.
q.arg	a list of shape parameters in the sampling distribution (dist).
parallel	a numeric value to assign the number of cores in parallel computing (default is set to 1).
mod	a list of model and parameters information that is used in solving differential equation.

Details

This function allows users to use external data file that assigned in setpoint.name as parameter matrix. If you want to use it, be sure to define n and setpoint.name.

Value

The output result is the 4-dimension array with c(model evaluations, replications, time-points, output variables).

Functions

- solve_mcsim(): Numerical analysis for the PK model by **GNU MCSim**.
- generate_infile(): Generate the **GNU MCSim** input file.

Examples

```
## Not run:
pbtk1cpt_model()
mName <- "pbtk1cpt"
compile_model(mName)

q <- "qunif"
q.arg <- list(list(min = 0.4, max = 1.1),
              list(min = 0.1, max = 0.4),
              list(min = 1.0, max = 3.0))

params <- c("vdist", "ke", "kgutabs")

set.seed(1234)
x <- rfast99(params = params, n = 200, q = q, q.arg = q.arg, rep = 20)

infile.name <- "example.in"
```

```
outfile.name <- "example.csv"
vars <- "Ccompartment"

t <- seq(from = 0.25, to = 12.25, by = 0.5)

y <- solve_mcsim(x, mName = mName, infile.name = infile.name,
                 setpoint.name = "setpoint.dat",
                 outfile.name = outfile.name, params = params, vars = vars, time = t,
                 condition = "Agutlument = 10")
pksim(y)

## End(Not run)
```

`tell2` *The Decoupling Simulations*

Description

Integrate the decoupling simulations (parameter sequences) and estimation results to compute the sensitivity measures.

Usage

```
tell2(x, y)
```

Arguments

`x`	a list of storing information in the defined sensitivity function.
`y`	a numeric array generated from the solutions of `solve_fun` or `solve_mcsim` function.

Source

This function is based on `tell` function in **sensitivity** package, which is an S3 generic method to estimate sensitivity measures by combining sensitivity object (`rfast99`) and external simulation results.

Index

∗ **datasets**
APAP, 3

about-pksensi, 2
APAP, 3

check, 3
compile_model, 6

FFPK *(*models*)*, 8

generate_infile *(*solve_mcsim*)*, 12

heat_check *(*check*)*, 3

mcsim_install, 7
mcsim_version *(*mcsim_install*)*, 7
models, 8

pbpk_apap_model *(*models*)*, 8
pbtk1cpt_model *(*models*)*, 8
pksensi-package *(*about-pksensi*)*, 2
pksim, 9, *12*
plot.rfast99 *(*check*)*, 3
print.rfast99 *(*check*)*, 3

rfast99, 9

set_rtools40_path *(*mcsim_install*)*, 7
solve_fun, 11
solve_mcsim, 12

tell2, *5*, 15