Package 'simode'

July 26, 2018

Type Package

Title Statistical Inference for Systems of Ordinary Differential Equations using Separable Integral-Matching

Version 1.0.1

Author Itai Dattner, Rami Yaari

Maintainer Itai Dattner <idattner@stat.haifa.ac.il>

Description Implements statistical inference for systems of ordinary differential equations, that uses the integral-matching criterion and takes advantage of the separability of parameters, in order to obtain initial parameter estimates for nonlinear least squares optimization.

Depends R (>= 3.4.0)

Imports deSolve, pracma, quadprog

Suggests parallel, Rcgmin, Rvmmin, R.rsp

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

VignetteBuilder R.rsp

NeedsCompilation no

R topics documented:

`confint.profile.simode` *Calculates confidence intervals for the model parameters*

Description

Calculates confidence intervals for the model parameters, based on the given likelihood profiles

Usage

```
## S3 method for class 'profile.simode'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

`object`	A fitted model object: `profile.simode` object returned by a call to `profile`.
`parm`	A specification of which parameters are to be given confidence intervals (named vector). If missing, all parameters are considered.
`level`	The confidence level required.
`...`	Additional argument(s) for methods.

Value

The confidence intervals.

`plot.confint.simode` *Plot confidence intervals for the model parameters*

Description

Plot confidence intervals for the model parameters of a `simode` object, calculated based on likelihood profiles

Usage

```
## S3 method for class 'confint.simode'
plot(x, which = NULL, pars_true = NULL,
  legend = F, cols = list(fit = "blue", true = "black"), ...)
```

Arguments

x	confint.simode object returned by a call to confint
which	Which parameters to plot the confidence intervals for. If empty, the plot will include all of the parameters in x.
pars_true	The true parameter values (if are known).
legend	Whether or not to add a figure legend.
cols	List of colors for each element of the plot.
...	Additional argument(s) for methods.

---

plot.list.simode *Plot the fit/estimates of a* list.simode *object*

Description

Plot the fit or parameter estimates obtained from a call to simode with mc_sets>1. Plots the mean and standard deviation obtained from the multiple fits.

Usage

```
## S3 method for class 'list.simode'
plot(x, type = c("fit", "est"), show = c("nls", "im",
  "both"), which = NULL, pars_true = NULL, time = NULL,
  plot_im_smooth = F, legend = F, mfrow = par("mfrow"),
  cols = list(nls_fit = "blue", im_fit = "green", true = "black", obs = "red",
  im_smooth = "magenta"), ...)
```

Arguments

x	simode object returned by a call to simode.
type	Type of plot - 'fit' to plot the fitted equations and 'est' to plot the parameter estimates.
show	Whether to plot the fit/estimates obtained using nonlinear least squares ('nls'), integral-matching ('im') or both ('both').
which	Which variables to plot in case type='fit', or which parameters to plot in case type='est'. If empty, the plot will include all of the variables/parameters in x.
pars_true	The true parameter values (if are known). Should be named using the parameter names. If given, the true values for the variables/parameters will be added to the plot.
time	The time points to use for the fitted curves (relevant only for type='fit'). If not given then the time points of the observations in x will be used.
plot_im_smooth	Whether or not to plot the smoothed curves created and used by the integral-matching procedure (relevant only for type='fit').
legend	Whether or not to add a figure legend.
mfrow	A vector of the form c(nr,nc) setting the layout of subplots in one plot (see also par).
cols	List of colors for each element of the plot.
...	Additional argument(s) for methods.

plot.profile.simode *Plot the likelihood profiles for the model parameters*

Description

Plot the likelihood profiles for the model parameters

Usage

```
## S3 method for class 'profile.simode'
plot(x, which = NULL, mfrow = par("mfrow"),
  cols = list(fit = "blue", threshold = "red"), ...)
```

Arguments

x	profile.simode object returned by a call to profile.
which	Which parameters to plot the likelihood profiles for. If empty, the plot will include all of the parameters in x.
mfrow	A vector of the form c(nr,nc) setting the layout of subplots in one plot (see also par).
cols	List of colors for each element of the plot.
...	Additional argument(s) for methods.

plot.simode *Plot the fit/estimates of a* simode *object*

Description

Plot the fit or parameter estimates obtained from a call to simode.

Usage

```
## S3 method for class 'simode'
plot(x, type = c("fit", "est"), show = c("nls", "im",
  "both"), which = NULL, pars_true = NULL, time = NULL,
  plot_im_smooth = F, legend = F, mfrow = par("mfrow"),
  cols = list(nls_fit = "blue", im_fit = "green", true = "black", obs = "red",
  im_smooth = "magenta"), ...)
```

Arguments

x	simode object returned by a call to simode.
type	Type of plot - 'fit' to plot the fitted variables and 'est' to plot the parameter estimates.
show	Whether to plot the fit/estimates obtained using nonlinear least squares ('nls'), integral-matching ('im') or both ('both').
which	Which variables to plot in case type='fit', or which parameters to plot in case type='est'. If empty, the plot will include all of the variables/parameters in x.

pars_true	The true parameter values (if are known). Should be named using the parameter names. If given, the true values for the variables/parameters will be added to the plot.
time	The time points to use for the fitted curves (relevant only for type='fit'). If not given then the time points of the observations in x will be used.
plot_im_smooth	Whether or not to plot the smoothed curves created and used by the integral-matching procedure (relevant only for type='fit').
legend	Whether or not to add a figure legend.
mfrow	A vector of the form c(nr,nc) setting the layout of subplots in one plot (see also par).
cols	List of colors for each element of the plot.
...	Additional argument(s) for methods.

---

plot_trace *Plot optimization trace of a call to* simode

Description

Plot a trace of the loss values and parameter estimates during the integral-matching/nonlinear least squares optimization within a call to simode. For the traces to exist, the arguments save_im_trace and/or save_nls_trace in simode.control should be set to true, when calling simode.

Usage

```
plot_trace(x, show = c("nls", "im", "both"), which = NULL,
  mfrow = par("mfrow"), cols = list(nls_fit = "blue", im_fit = "green"),
  ...)
```

Arguments

x	simode object returned by a call to simode.
show	Whether to plot the estimates obtained using nonlinear least squares ('nls'), integral-matching ('im') or both ('both').
which	Which parameters' traces to plot. If NULL, the trace for all the parameters in x will be plotted.
mfrow	A vector of the form c(nr,nc) setting the layout of subplots in one plot (see also par).
cols	List of colors for each element of the plot.
...	Additional argument(s) for methods.

print.confint.simode *Print method for* confint.simode *objects*

Description

Print method for confint.simode objects

Usage

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

Arguments

x	The confint.simode object.
...	Additional argument(s) for methods.

print.profile.simode *Print method for* profile.simode *objects*

Description

Print method for profile.simode objects

Usage

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

Arguments

x	The profile.simode object.
...	Additional argument(s)for methods.

print.simode *Print method for* simode *objects*

Description

Print method for simode objects

Usage

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

Arguments

x	The simode object.
...	Additional argument(s) for methods.

print.summary.simode *Print method for* summary.simode *objects*

Description

Print method for summary.simode objects

Usage

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

Arguments

x	The summary.simode object.
...	Additional argument(s) for methods.

profile.simode *Calculate likelihood profiles for the model parameters*

Description

Calculate likelihood profiles for the model parameters

Usage

```
## S3 method for class 'simode'
profile(fitted, which = NULL, optim_type = c("nls",
  "both"), step_size, max_steps, alpha = 0.05, skip_err = T, trace = 0,
  save_to_log = F, ...)
```

Arguments

fitted	simode object returned by a call to simode.
which	Which parameters to estimate the profile for.
optim_type	Whether to calculate the profiles based on maximum-likelihood optimization only ('nls') or based on integral-matching followed by maximum-likelihood optimization ('both').
step_size	Step size for profiling (one value for all parameters or a value for each parameter in which).
max_steps	Maximum number of steps to take in each direction.
alpha	Maximum (two-sided) likelihood ratio test confidence level to find.
skip_err	Whether on not to stop the calculation if encountering a problem with one point in the profile.
trace	Report level (0-4), with higher values producing more tracing information.
save_to_log	Whether to redirect output to log file. The log file will be saved to tempdir().
...	Additional argument(s) for methods.

Details

If the call to simode, which returned the fitted object given to this method, included a user-defined likelihood function (with the calc_nll argument), then the likelihood profiles will be calculated using this function. Otherwise, the profiles will be calculated using a likelihood based on a Gaussian distribuion with fixed sigma, where sigma will be estimated in the background together with the rest of the model parameters.

Value

The likelihood profiles.

---

simode *Statistical inference of ordinary differential equations using separable integral-matching*

Description

Estimating the parameters of an ODE system in two stages: 1) Estimate the parameters using separable integral-matching, 2) Estimate the parameters using nonlinear least squares starting from the values obtained in stage 1.

Usage

```
simode(equations, pars, time, obs, mc_sets = 1, nlin_pars = NULL,
  likelihood_pars = NULL, fixed = NULL, start = NULL, lower = NULL,
  upper = NULL, im_method = c("separable", "non-separable"),
  gen_obs = NULL, calc_nll = NULL, simode_ctrl = simode.control(), ...)
```

Arguments

equations	Named vector. The equations describing the ODE system. Each element of the vector should contain a character representation of the right-hand side of an equation, and should be named according to the left-hand side of the equation (i.e., the variable name). An equation can contain parameters appearing in pars, variables appearing in the equations names and/or any function of 't', which is a reserved symbol for the time domain.
pars	The names of the parameters and initial conditions to be estimated. An initial condition name for a certain variable is the name given to the relevant equation in equations (e.g., if an equation is named 'x' than its initial condition should be named 'x' as well). Note: The symbol 't' is reserved for the time domain and cannot be used as a parameter name.
time	Time points of the observations. Either a vector, if the same time points were used for observing all variables, or a list of vectors the length of obs, of which each element is the length of the relevant element in obs.
obs	Named list. The observations. When mc_sets=1, obs should contain a list of vectors with length <= equations (either partial or fully observed system). Each list member should be named according to the relevant equation name and should be the length of the relevant time vector. When mc_sets>1, obs should contain a list the length of mc_sets, where each list member is a list that fits the description in the case of mc_sets=1.

mc_sets	Number of monte-carlo expriments. When mc_sets>1, the function will fit each set of observations separately either sequentially or in parallel, according to the value of parallel in simode_ctrl.
nlin_pars	Names of parameters or initial conditions that will be estimated in stage 1 using nonlinear least squares optimization. The parameter names in nlin_pars must appear in pars.
likelihood_pars	Names of likelihood parameters not appearing in the ODE system, which are needed for the the user-defined function calc_nll. The parameter names in likelihood_pars must appear in pars.
fixed	Named vector. Fixed values for one or more of the ODE system parameters or initial conditions. Parameters in this list will not be estimated.
start	Named vector. Starting values for optimization of parameters/initial conditions. Must contain starting values for all the parameters in nlin_pars and likelihood_pars. If im_method="non-seperable", can optionally contain starting values for any other parameter/initial condition.
lower	Named vector. Lower bounds for any parameter/initial condition.
upper	Named vector. Upper bounds for any parameter/initial condition.
im_method	The method to use for integral-matching. Default "separable" means that linear parameters are estimated directly while "non-separable" means that linear parameters are estimated using nonlinear least squares optimization. If none of the parameters are linear then the default can be used.
gen_obs	A user-defined function for completing missing observations (see Details).
calc_nll	A user-defined function for calculating negative log-likelihood for the model (see Details).
simode_ctrl	Various control parameters. See simode.control.
...	Additional arguments passed to optim, gen_obs and calc_nll

Details

gen_obs can be used in cases of a partially observed system, for which observations of the missing variables can be generated given values for the system parameters. The function will be called during the optimization using integral-matching.
It must be defined as gen_obs <- function(equations, pars, x0, time, obs, ...), where:

- equations the ODE equations
- pars the parameter values
- x0 the initial conditions
- time the timing of the observations (vector or list)
- obs the observations
- ... additional parameters passed from the call to simode

The function should return a list with two items:

- time the vector or list of time points of the observations
- obs the list of observations with the newly generated observations

calc_nll allows the user to pass his own likelihood function to be used in the optimization in the second stage (if not defined, the default nonlinear least squares optimization will be used). The likelihood function will also be used in a following call to profile, for the calculation of likelihood profiles. It must be defined as calc_nll <- function(pars, time, obs, model_out, ...), where:

- pars the parameter values
- time the timing of the observations (vector or list)
- obs the observations
- model_out the model output returned from a call to solve_ode. If time is a list with possibly different times for each variable then model_out will contain a union of all these times.
- ... additional parameters passed from the call to simode

The function should return the negative log-likelihood.

Value

If mc_sets=1, returns a simode object containing the parameter estimates after integral-matching (stage 1) and after nonlinear least squares optimization (stage 2). If mc_sets>1 returns a list.simode object which is a list of simode objects the length of mc_sets.

References

Dattner & Klaassen (2015). Optimal Rate of Direct Estimators in Systems of Ordinary Differential Equations Linear in Functions of the Parameters, Electronic Journal of Statistics, Vol. 9, No. 2, 1939-1973.

Dattner, Miller, Petrenko, Kadouriz, Jurkevitch & Huppert (2017). Modelling and Parameter Inference of Predator-prey Dynamics in Heterogeneous Environments Using The Direct Integral Approach, Journal of The Royal Society Interface 14.126: 20160525.

Examples

```
## Not run:
## ================================================
## Predator-Prey Lotka-Volterra model
## ================================================

## generate model equations and parameters (X=Prey,Y=Predator)
pars <- c('alpha','beta','gamma','delta')
vars <- c('X','Y')
eq_X <- 'alpha*X-beta*X*Y'
eq_Y <- 'delta*X*Y-gamma*Y'
equations <- c(eq_X,eq_Y)
names(equations) <- vars
x0 <- c(0.9,0.9)
names(x0) <- vars
theta <- c(2/3,4/3,1,1)
names(theta) <- pars

## generate observations
n <- 100
time <- seq(0,25,length.out=n)
model_out <- solve_ode(equations,theta,x0,time)
x_det <- model_out[,vars]
```

```
set.seed(1000)
sigma <- 0.1
obs <- list()
for(i in 1:length(vars)) {
  obs[[i]] <- pmax(0, rnorm(n,x_det[,i],sigma))
}
names(obs) <- vars

## estimate model parameters with known initial conditions
simode_fit1 <- simode(equations=equations, pars=pars, fixed=x0, time=time, obs=obs)
plot(simode_fit1, type='fit', pars_true=theta, mfrow=c(2,1))
plot(simode_fit1, type='est', pars_true=theta)

## estimate model parameters and initial conditions
simode_fit2 <- simode(equations=equations, pars=c(pars,vars), time=time, obs=obs)
plot(simode_fit2, type='fit', pars_true=c(theta,x0), mfrow=c(2,1))
plot(simode_fit2, type='est', pars_true=c(theta,x0))

profiles_fit2 <- profile(simode_fit2,step_size=0.01,max_steps=50)
plot(profiles_fit2,mfrow=c(2,3))
ci_fit2 <- confint(profiles_fit2)
ci_fit2
plot(ci_fit2,pars_true=c(theta,x0),legend=T)

## End(Not run)
```

`simode.control` *Class containing control parameters for a call to* `simode`

Description

Class containing control parameters for a call to `simode`

Usage

```
simode.control(optim_type = c("both", "im", "nls"),
  im_optim_method = c("BFGS", "Nelder-Mead", "CG", "L-BFGS-B", "SANN",
  "Brent", "Rcgmin", "Rvmmin"), nls_optim_method = c("BFGS", "Nelder-Mead",
  "CG", "L-BFGS-B", "SANN", "Brent", "Rcgmin", "Rvmmin"),
  im_optim_control = list(), nls_optim_control = list(),
  ode_control = list(method = "lsoda"), im_smoothing = c("splines",
  "kernel", "none"), im_grid_size = 0, bw_factor = 1.5,
  use_pars2vars_mapping = F, trace = 0, save_im_trace = F,
  save_nls_trace = F, save_to_log = F, parallel = F)
```

Arguments

`optim_type`	Controls what optimization will be performed: either only integral-matching ('im'), only nonlinear least squares ('nls') or both (the default, i.e., first integral-matching then nonlinear least squares starting from the integral-matching estimates).

`im_optim_method`
: Method for optimization during the integral-matching stage. Accepted values are any method supported by the `method` argument in `optim`, as well as "Rvmmin" and "Rcgmin", if the relevant packages are installed.

`nls_optim_method`
: Method for optimization during the nonlinear least squares stage. Accepted values are the same as in `im_optim_method`.

`im_optim_control`
: A list with control parameters for optimization during the integral-matching stage. Can include anything that would appear in the `control` argument in `optim`/`Rvmmin`/`Rcgmin` (depending on the choice of `im_optim_method`). See `optim`, `Rvmmin`, `Rcgmin`.

`nls_optim_control`
: Control parameters for optimization during the nonlinear least squares stage (as in `im_optim_control`)

`ode_control`
: A list with control parameters for the ODE solver. Can include the argument `method` appearing in the arguments to `ode`, as well as any other control parameters accepted as additional parameters in the call to `ode`.

`im_smoothing`
: Choice of type of smoothing during the integral-matching stage (see Details).

`im_grid_size`
: Number of points used in integral-matching grid (not relevant when `im_smoothing='kernel'`). Value <=0 means the grid size will be set according to maximum number of observations for any of the equations in the call to `simode`.

`bw_factor`
: Controls the bandwidth when `im_smoothing='kernel'`. The bandwidth for each equation will be bw_factor*the maximum time interval between two observations (should be >= 1).

`use_pars2vars_mapping`
: Whether to use pars2vars mapping (see Details).

`trace`
: Report level (0-4), with higher values producing more tracing information (see Details).

`save_im_trace`
: Whether to save trace information of integral-matching optimization, which can then be plotted using `plot_trace`.

`save_nls_trace`
: Whether to save trace information of nonlinear least squares optimization, which can then be plotted using `plot_trace`.

`save_to_log`
: Controls whether to redirect output to a log file. If true, output will be saved to the file 'simode.log' in tempdir.

`parallel`
: Controls whether to fit multiple `mc_sets` in the call to `simode` sequentially or in parallel. Fitting in parallel requires that the parallel package will be installed. When running in parallel, output will not be displayed regardless of the trace level. Instead, one can set `save_to_log` to true to save the output to a log file.

Details

Possible values for `im_smoothing` are "splines" (the default), in which case smoothing will be performed using `smooth.spline` with generalized cross-validation, "kernel", using own kernel smoother function, or "none" (using the observations as is, with interpolation if necessary).

`use_pars2vars_mapping` controls whether to use a mapping of which equations are affected by each of the parameters. When set to true, previous matrices computed as part of the integral-matching estimation are stored during the integral-matching optimization, and are updated only for the equations that were affected by the change in the parameter estimates from the previous iteration.

When the number of equations is large and some of the parameters affect only a few equations, setting this option to true can significantly reduce the optimization time during the integral-matching stage (while increasing the storage usage). This is especially true with derivative based optimization methods (such as "BFGS" of optim) which updates only one of the optimized parameters in each iteration.

`trace` has 5 possible levels:
With trace=0, there would be no output displayed if there are no errors.
With trace=1, a message will be displayed at the beginning and end of each optimization stage.
With trace=2, non-critical errors occurring during the optimization iterations will be displayed.
With trace=3, non-critical warnings occurring during the optimization iterations will be displayed.
With trace=4, the calculated loss value for each iteration of the integral-matching and nonlinear least squares optimizations will be displayed.

`sir_example` *Example dataset for a multi-group SIR model*

Description

A model for the spread of a seasonal influenza epidemics in two groups over five seasons.

Usage

```
sir_example
```

Format

A list containing the following variables:

equations The ODE equations describing the system, including ten equations for the susceptible dynamics (2 groups * 5 seasons) and ten equations for the infected dynamics.

beta The 2x2 transmission matrix parameters (in time unit of weeks).

gamma The recovery rate parameter (in time unit of weeks).

kappa The relative infectiousness of seasons 2-5 compared to season 1.

S0 The initial conditions for the susceptible variables.

I0 The initial conditions for the infected variables.

time Times in which the observations were made (in weeks).

obs A list of observations of the infected variables, generated using Gaussian measurement error with sigma=1e-3.

solve_ode *Ordinary differential equations solver*

Description

A wrapper for the ode function that solves a system of ordinary differential equations described using symbolic equations.

Usage

```
solve_ode(equations, pars, x0, time, ...)
```

Arguments

equations	The equations describing the ODE system. See simode.
pars	The parameter values. Named according to their names in equations.
x0	The initial conditions. Named accroding to the names of equations.
time	The time points for which the ODE variables' values will be computed.
...	Additional argument(s) for methods.

Value

A matrix whose first column contains the given time points and subsequent columns hold the computed ODE equations' values at these time points.

summary.list.simode *Summary method for* list.simode *objects*

Description

Summary method for list.simode objects

Usage

```
## S3 method for class 'list.simode'
summary(object, pars_true = NULL, digits = max(3,
  getOption("digits") - 3), ...)
```

Arguments

object	list.simode object returned by a call to simode with mc_sets>1
pars_true	The true parameter values (if known).
digits	The number of significant digits to use.
...	Additional argument(s) for methods.

Value

The mean and standard deviation for the loss values and parameter estimates obtained from the integral-matching and nonlinear least squares optimizations. If pars_true is given, then will also calculate bias and RMSE for the parameter estimates.

summary.simode *Summary method for* simode *objects*

Description

Summary method for simode objects

Usage

```
## S3 method for class 'simode'
summary(object, digits = max(3, getOption("digits") - 3),
  ...)
```

Arguments

object	The simode object.
digits	The number of significant digits to use.
...	Additional argument(s) for methods.

Index