Package 'nlsrk'

June 24, 2017

Version 1.1

Date 2017-06-22

Title Runge-Kutta Solver for Function nls()

Author Jean-Sebastien Pierre <jean-sebastien.pierre@univ-rennes1.fr>

Maintainer Jean-Sebastien Pierre <jean-sebastien.pierre@univ-rennes1.fr>

Depends R (>= 3.0.0)

Description Performs univariate or multivariate computation of a single ODE or of a set of ODE (ordinary differential equations).

License GPL (>= 2)

URL none

NeedsCompilation no

Repository CRAN

Date/Publication 2017-06-23 23:01:24 UTC

R topics documented:

`nlsrk-package` *Runge-Kutta package for nls*

Description

Performs univariate or multivariate computation of a single ODE or of a set of ODE (ordinary differential equations)

Details

Package:	nlsrk
Version:	1.0
Date:	2006-09-07
Depends:	R (>= 1.8.0)
Suggests:	
License:	GPL version 2 or newer
URL:	none
Built:	R 2.2.1; ; 2006-02-22 19:52:45; windows

Univariate functions (single ODE)

`dfdt`	Right Side of a first order ODE
`frunge`	Numeric solution of an ODE (univariate)

Multivariate functions (ODEs system)

`multirunge`	Runge-Kutta numerical solver (single point, multivariate)
`evrunge`	Numerical Runge-Kutta Solver (multi point, multivariate)
`prepare`	Prepares a multivariate data set for use in nls / evrunge
`sys`	Right side of a first order differential equation system (multivariate)

Utility

`nlscontour`	Draws a contourplot of the sum of squares function around a couple of parameters
`summary.nlsgrid`	Details on an nlsgrid

data

`logis`	Logistic growth of bacteria
`syslin`	Pharmacokinetics : intramuscular injection
`syslin.don`	Intra muscular injection (columns)

Note

nlsrk provides to nls(stats) an evaluation of functions defined by first order Ordinary Differential Equations (ODEs). For a single ODE use `frunge` and edit `dfdt`. For a system of ODEs, use the multivariate functions `evrunge` and edit `sys`. The method used in both cases is the Runge-Kutta 4 method. The result is not guaranteed especially for "stiff" systems of equations.

Author(s)

Jean-Sebastien Pierre <jean-sebastien.pierre@univ-rennes1.fr> Maintainer: Jean-Sebastien Pierre <jean-sebastien.pierre@univ-rennes1.fr>

References

Numerical Recipes `http://www.nr.com`

See Also

`nls`

`dfdt` *Right Side of a first order ODE*

Description

Gives the value of the right side of a first order Ordinary Differential Equation Used in frunge (Numeric resolution using Runge-Kutta 4 metthod)

Usage

```
dfdt(t, y, param)
```

Arguments

`t`	time, independent variable
`y`	values of the unknown function at t
`param`	numeric vector : the set of parameters defining the function

Details

the unique expression composing the body of dfdt is intended as the right side of the first order differential equation :

$$\frac{dy}{dt} = dfdt(t, y, param)$$

The function body has to be written by the user.
param must take the form c(param1,param2,param3). if the initial value (y0) is used as a parameter (for model fitting for instance), it **must be the last** in the list and **not used** in dfdt.

Value

a single float numeric value : the derivative of the unknown function at time t.

Note

When used as argument for frunge, the initial condition **y0** may be used as a parameter subject to fitting by `nls`, for instance. In this condition, **y0** must appear as the **last** parameter in param and **must not** be used nor modified in `dfdt`

Author(s)

Jean-Sebastien Pierre
<Jean-sebastien.pierre@univ-rennes1.fr>

References

Any textbook in mathematics

See Also

`frunge`, `nls`

Examples

```
## Solves and draws the logistic function using default dfdt as provided in the package

frunge(t=10, param=c(r=0.1,k=100), y0=3, Dfdt = dfdt, dt = 0.01, graph = TRUE)
```

---

evrunge *Numerical Runge-Kutta Solver (multi point, multivariate)*

Description

evrunge evaluates the solutions of a system of first order ordinary equations for a given set of values of the independent variable (generally the time).

Usage

```
evrunge(t, param, y0, sys, dt = 0.01, graph = FALSE, observable = rep(1, length(y0)))
```

Arguments

t	numerical vector : the values of t at which the ODE system must be evaluated
param	numerical vector or numerical objects list. Passed as arguments to sys
y0	numerical vector : the initial values of the unknown functions to solve
sys	the function giving the right side of the system. Must be written by user (see ?sys)
dt	integration step for Runge-Kutta algorithm. Default = 0.01
graph	optionally : graphic representation of the set of solutions. Default = FALSE
observable	A numeric vector coding what trajectories are observable. 1 = observable, 2 = not observable. Default : a vector of 1, nfunct times

Details

This function is intended basically to be used in conjunction with **nls** for non linear least square fit of a set of ODEs on experimental data. This is the reason why the solutions are concatenated in a single column. For fitting with nls, the data have to be organised in the same way. Use prepare(nlsrk) to convert a data frame with observations in separate columns in one where the columns are concatenated. In a further version of this function, the solutions will optionnaly be provided as columns of an data.frame. The system must be provided by user. See sys for an editable example.

t **must be** sorted in ascending order **and** every intervals between two consecutive values of t **must be greater then dt**. Otherwise, the function stops and an error message is displayed.

Although the algorithm works with dt slightly lower than with min(t[i+1] - t[i]) the accuracy of the results are not guaranted. It is recommanded to chose with dt fairly lower than the minimum interval in with t See the note below concerning the parameter 'observable'

Value

A vector of size nfunct*length(t) where nfunct is the number of equations in sys

Note

Calls multirunge for each point t
Uses Runge-Kutta 4 algorithm. The error is of order dt^4. This algorithm is robust and simple but is not guaranteed to work correctly on any ODE system. In particular, RK4 is a fixed step algorithm. "stiff" systems of equations require generally more sophisticated methods with adaptive steps.

Sometimes, especially when the observations are the result of a sampling procedure, it may be useful to treat the initial conditions as parameters injected in the fitting procedure of nls. Note that in these conditions, y0 appears three times in the call to nls: in param, as parameter of evrunge and as starting values for parameters in start

The option observable allows to fit data sets where the data are obzervable only for a subset of the state variables involved in the equations system. There is **no guarantee that nls converges in this case**. The success of the algorithm requires the model to be **identifiable** with the subset of observations. Identifiability is a difficult mathematical topic that filled a lot of books.

Author(s)

Jean-Sebastien Pierre
<Jean-sebastien.pierre@univ-rennes1.fr>

References

~put references to the literature/web site here ~

See Also

multirunge, sys, prepare, nls

Examples

```
##
## example 1 : solving and plotting the system sys provided in the package
##
data(syslin.don)
syslin<-prepare(syslin.don)
  evrunge(t=c(1:30),param=c(1,1),y0=c(1000,0),sys=sys,graph=TRUE)
##
## example 2 : fitting by nls on data \code{syslin} fixed and known initial conditions
##
data(syslin)
attach(syslin)
nls(y~ evrunge(t,param=c(k1,k2),y0=c(1000,0),sys,graph=FALSE),data=syslin,start=list(k1=1,k2=1),
trace=TRUE)->m1
summary(m1)
detach(syslin)
##
## example 3 : fitting by nls on data syslin "unknown" initial conditions:
##                  they have to be fitted as parameters
##
data(syslin)
nls(y~ evrunge(t,param=c(k1,k2,y0),y0,sys,graph=FALSE),data=syslin,start=list(k1=1,k2=1,
y0=c(1000,0)),trace=TRUE)->m2
summary(m2)
plot.nlsrk(m2,syslin)
##
## example 4 : fitting by nls on data syslin with known initial conditions:
##                There are no observations for the trajectory 1 which is not observable.
##     sys is unchanged
##
data(syslin)
##     We eliminate the data corresponding to trajectory 1 and fit only on trajectory 2.
##     Fixed initial conditions
 syslin2<-syslin[syslin$traj==2,]
 attach(syslin2)
 nls(y~ evrunge(t,param=c(k1,k2),y0=c(1000,0),sys,graph=FALSE,observable=c(0,1)),data=syslin2,
     start=list(k1=1,k2=1),trace=TRUE)->m3
 summary(m3)
```

```
plot.nlsrk(m3,syslin2)
detach(syslin2)
```

`frunge` *Numeric solution of an ODE (univariate)*

Description

frunge provides a sampled numeric solution for an ODE for a given set of values of the independent variable (t). Returns a series of values corresponding to each values of t

Usage

```
frunge(t, param, y0, Dfdt, dt = 0.01, graph = FALSE)
```

Arguments

`t`	The independent variable. Must be sorted in increasing order
`param`	Numeric vector : under the form of `c(param1,param2...)` gives the parameters passed to `dfdt`
`y0`	The initial condition. May appear as a parameter for nls. See details
`Dfdt`	A function giving the right side of the ODE. Default is `dfdt(nlsrk)`
`dt`	Time increment for Runge-Kutta algorithm. **Must** be lower than the smaller difference between two consecutive **t** values
`graph`	If true, plots the graph of the function over the range of t. (For standalone usage)

Details

The vector **t must be** sorted in increasing order. Any badly placed `t` will result in an error and program exit. dt must be lower than `min(diff(t))`. If not, an error will occur. The shorter **dt**, the more accurate the solution but the longer the calculation will be.

Value

A numeric vector of length equal to the length of **t**. **y[1]** is set to **y0**

Author(s)

Jean-Sebastien Pierre
<Jean-sebastien.pierre@univ-rennes1.fr>

References

Numerical recipes http://www.nr.com

See Also

dfdt,nls

Examples

```
v<-frunge(t=seq(0,50,0.5), param=c(r=0.1,k=100), y0=3, Dfdt = dfdt, dt = 0.01, graph = TRUE)
       plot(seq(0,50,0.5),v)
##
##     Example of model fitting with frunge. a : determined initial condition
##
data(logis)
attach(logis)
nls(y~frunge(time,c(r,k),y0=3,dfdt,graph=FALSE),data=logis,start=list(r=0.05,k=100),trace=TRUE)->m1
plot(time,y)
lines(time,fitted(m1))
summary(m1)
detach(logis)
##
##     Example of model fitting with frunge. b : unknown initial condition ;
##     y0 is fitted as a parameter

## Not run: data(logis)
attach(logis)
nls(y~frunge(time,c(r,k,y0),y0,dfdt,graph=FALSE),data=logis,start=list(r=0.05,k=100,y0=y[1]),
    trace=TRUE)->m2
plot(time,y)
lines(time,fitted(m2))
summary(m1)
detach(logis)
## End(Not run)
```

---

kmk *mck-Mc Kendrick SIR system*

Description

a variant of sys to use in evrunge

Usage

```
kmk()
```

Value

see sys Three values for three equations are returned

Author(s)

J.S. Pierre

See Also

sys, evrunge

Examples

```
## The function is currently defined as
function ()
{
    f1 <- function(t, y, param) -param[1] * y[1] * y[2]
    f2 <- function(t, y, param) param[1] * y[1] * y[2] - param[2] * y[2]
    f3 <- function(t, y, param) param[2] * y[2]
    c(f1, f2, f3)
  }
```

---

logis *Logistic growth of bacteria*

Description

Logistic growth in bacteria : time versus Optical Density in arbitrary units (simulated data)

Usage

```
data(logis.don)
```

Format

A data frame with 51 observations on the following 2 variables.

`time` a numeric vector : time in hours

`y` a numeric vector : Optical Density

Details

a dataset suitable to run nls and fit a logistic model, eventually under the form of a differential equation

Source

personnal source J.S. Pierre

Examples

```
data(logis)
## maybe str(logis) ; plot(logis) ...
```

logis.don *Logistic growth of bacteria*

Description

Logistic growth in bacteria : time versus Optical Density in arbitrary units (simulated data)

Usage

```
data(logis.don)
```

Format

A data frame with 51 observations on the following 2 variables.

`time` a numeric vector : time in hours

`y` a numeric vector : Optical Density

Details

a dataset suitable to run nls and fit a logistic model, eventually under the form of a differential equation

Source

personnal source J.S. Pierre

Examples

```
data(logis.don)
## maybe str(logis.don) ; plot(logis.don) ...
```

multirunge *Runge-Kutta numerical solver (single point, multivariate)*

Description

Solves numerically an initial conditions problem for a set of Ordinary Differential Equations (ODE) by Runge-Kutta 4 method. Integrates numerically the equations from **tmin** to **tmax** by steps of **dt**.

Usage

```
multirunge(y0, tmin, tmax, dt, param, sys)
```

Arguments

y0	Numerical vector : initial conditions (as many elements as equations in sys)
tmin	Minimum value of the independent variable (generally the time)
tmax	Maximum value of the independent variable
dt	Time increment (default = 0.01)
param	Numerical vector providing the parameters for sys
sys	The set of functions giving the right sides of the ODEs

Details

sys must be provided by the user. Please edit the object sys (see ?sys and examples)

Value

A numerical vector of **nfunct** elements. Nfunct is the number of unknown functions of sys determined by the function as length(sys())

Note

Should seldom be used independently. Most often it will be called by **evrunge**

Author(s)

Jean-Sebastien Pierre
<Jean-sebastien.pierre@univ-rennes1.fr>

See Also

evrunge, sys, nls

Examples

```
multirunge(y0=c(1000,0),tmin=0,tmax=30,dt=0.01,param=c(1,1),sys=sys)
```

---

nlscontour *Contour plot for nonlinear least squares*

Description

Takes as central values the results of non-linear least square fitting by the nls(base, stats) function, then draws a contour plot of the residual sum of squares function around these values.

Usage

```
nlscontour(x, param1 = 1, param2 = 2, range1 = NULL, range2 = NULL,
npoints = 100, filled = FALSE,colored=FALSE)
```

Arguments

x	a nls model
param1	The number of the first parameter to plot in the parameters list of model formula
param2	The number of the second parameter to plot
range1	The range (min and max) for plotting parameter param1. Default = +/-4*Standard Error of the parameter
range2	The range (min and max) for plotting parameter param2. Default = +/-4*Standard Error of the parameter
npoints	Number of points of the grid for contour. Default = 100
filled	Defines the style of the contour plot FALSE: lines (default), TRUE: filled contours (shades of gray)
colored	Defines the style of the contour plot FALSE: lines (default), TRUE: Colored contours (red= lower)

Details

npoints defines the total number of points of the square grid used for drawing the contour plot. Thus, the grid will be of size round(sqrt(npoints))*round(sqrt(npoints))

Value

an object of class nlsgrid with three components:

$x	the values used for the first parameter
$y	the values used for the second parameter
$grid	a square matrix: The values of the residual sum of squares for each combination of the parameters

...

Author(s)

Jean-Sebastien Pierre
<Jean-sebastien.pierre@univ-rennes1.fr>

References

~put references to the literature/web site here ~

See Also

nls logis summary.nlsgrid

Examples

```
data(logis)
attach(logis)
m1<-nls(y~k/(1+c*exp(-r*time)),data=logis,start=c(k=100,r=0.1,c=40))
nlscontour(m1)
detach(logis)
```

`nlsModel` *calculate a model as defined in nls for the set of parameters* `start` *on the values* `data`

Description

Recovered from older versions of R (<= 1.8), obsolete in 2.10

Usage

```
nlsModel(form, data, start)
```

Arguments

`form`	A nls model object
`data`	a data frame of independent variables
`start`	a list of parameter values

Value

a vector of estimates

Author(s)

R development core team

`plot.nlsrk` *Plot of multivariate data fitted by the nlsrk package*

Description

Plot the experimental points of each trajectory with the lines fitted by nls / nlsrk

Usage

```
 ## S3 method for class 'nlsrk'
plot(x, data, maintitle = "Ordinary Differential Equations system",...)
```

Arguments

x	A "nls" object obtained with evrunge
data	a data.frame with only three columns named (imperatively) t, y, traj in that order
maintitle	Title for the graph
...	optional parameters for plot

Details

syslin Minimal and preliminary implementation of a plot method for "nls" objects obtained by Runge-Kutta 4 resolution of a set of ODE. The symbols of the experimental points are chosen autom)tically according to the formula pch=19+data$traj. Then, the first trajectory is drawn with pch=20 (black dot), the next with pch=21 (empty circles), etc... see plot(options) and par for details. Similarily, the lines of the fitted solutions are drawn in a color defined automatically according to the formula col=data$traj. The first trajectory is drawn in black (col=1), the next in red (col=2), etc... Any object of type "nls" is accepted by the program, but if it does not result of the use of evrunge, inpredictible errors will occur.

Value

none

Author(s)

jean-sebastien pierre
<jean-sebastien.pierre@univ-rennes1.fr>

See Also

evrunge

Examples

```
##
## example 1 : solving and plotting the system sys provided in the package
##
data(syslin.don)
syslin<-prepare(syslin.don)
attach(syslin)
nls(y~ evrunge(t,param=c(k1,k2),y0=c(1000,0),sys,graph=FALSE),data=syslin,
start=list(k1=1,k2=1),trace=TRUE)->m1
plot.nlsrk(m1,syslin)
##
```

prepare *Prepares a multivariate data set for use in nls / evrunge*

Description

Utility :
Modifies a multivariate `data.frame` for fitting a nonlinear model defined by a system of first order Ordinary Differential Equations. In the original `data.frame`, the data must be arrange in **nfunct + 1** columns, **nfunct** being the dimension of the ODE system. One of the column must be the observation times (or in general the observations of the independent variable). The others colums are the observations of the functions. `prepare` returns the same data set with a single column for time (repeated **nfunct** times), and a single column for the concatenated observations. A third column is created as a `factor` indexing the original column (1 for the first, 2 for the second, etc)

Usage

```
prepare(x = data.frame, ntime = 1, cols = c(2:(length(x[1, ]))))
```

Arguments

`x`	a data.frame : one column for time and one for each function
`ntime`	numeric : the number of the column coding for time (independent variable)
`cols`	numeric vector : the numbers of the columns corresponding to each dependent variable

Details

Prepares the data set as an entry for `nls` which is basically univariate. The formula in `nls` must involve `evrunge`. As `nls` considers the observations as univariate, the criterion used in the least-square process is that of the **trace** of the variance-covariance matrix of the residuals.

Value

A data.frame with columns :

`t`	time repeated as many times as necessary
`y`	concatenated observed trajectories
`traj`	factor : number of the trajectory

Author(s)

Jean-Sebastien Pierre <Jean-sebastien.pierre@univ-rennes1.fr>

See Also

`frunge`, `dfdt`,`multirunge`,`evrunge`,`sys`,`nls`

Examples

```
data(syslin.don)
syslin<-prepare(syslin.don)
summary(syslin)
```

`summary.nlsgrid` *Summary method for objects of class nlsgrid(nlsrk)*

Description

Prints the characteristics of an nlsgrid

Usage

```
## S3 method for class 'nlsgrid'
summary(object, ...)
```

Arguments

`object`	An object of class nlsgrid
`...`	Any parameters for summary

Value

NULL

Author(s)

Jean-Sebastien Pierre
<Jean-sebastien.pierre@univ-rennes1.fr>

References

none

See Also

`nlsrk`, `nlsrk-package`, `nls`, `nlscontour`

Examples

```
data(logis)
m1<-nls(y~k/(1+c*exp(-r*time)),data=logis,start=list(k=100,r=0.1,c=45))
gr12<-nlscontour(m1)
summary(gr12)
```

sys *Right side of a first order differential equation system (multivariate)*

Description

sys gives the set of derivatives of a first order differential equation system. The functions must be provided by user. Edit and change **sys**.

Usage

```
sys()
```

Details

sys has no arguments but the componing functions must have three : **t, y param**. sys returns the rights sides of the set of **nfonct** functions.

Value

A numeric vector of the **nfonct** functions at time **t**

Author(s)

Jean-Sebastien Pierre
<Jean-sebastien.pierre@univ-rennes1.fr>

References

any mathematical textbook

See Also

evrunge, multirunge, nls

Examples

```
evrunge(seq(0,10,0.1),c(1,1),y0=c(1000,0),sys=sys,graph=TRUE)
```

syslin *Pharmacokinetics : intramuscular injection*

Description

A survey in time of muscle and blood concentrations of a drug following an intramuscular injection.

Usage

```
data(syslin)
```

Format

A data frame with 100 observations on the following 3 variables.

`t` a numeric vector : time in hours

`y` a numeric vector : blood or muscle concentration

`traj` a factor : number of the compartments. 1=muscle, 2= blood

Source

simulated data (J.S. Pierre)

Examples

```
data(syslin)
## maybe str(syslin) ; plot(syslin) ...
```

syslin.don *Intra muscular injection (columns)*

Description

Concentration of a drug in the muscle and in the blood at various times after injection

Usage

```
data(syslin.don)
```

Format

A data frame with 50 observations on the following 3 variables.

`t` a numeric vector: time

`y1` a numeric vector : muscle concentration

`y2` a numeric vector : blood concentration

Details

must be modified by prepare to be fitted by nls / evrunge

Source

Simulated data (J.S. Pierre)

Examples

```
## concatenation of muscle and blood concentration by prepare

data(syslin.don)
summary(syslin.don)
syslin<-prepare(syslin.don)
summary(syslin)
```

Index