Package ‘PBSddesolve’

September 20, 2016

Version 1.12.2
Date 2016-09-20
Title Solver for Delay Differential Equations
Author Alex Couture-Beil [aut], Jon T. Schnute [aut], Rowan Haigh [aut, cre],
    Simon N. Wood [aut], Benjamin J. Cairns [aut], Nicholas Boers [ctb]
Maintainer Rowan Haigh <rowan.haigh@dfo-mpo.gc.ca>
Copyright 2007-2016, Fisheries and Oceans Canada
Depends R (>= 2.15.0)
Suggests PBSmodelling
NeedsCompilation yes
Description Routines for solving systems of delay differential equations by
    interfacing numerical routines written by Simon N. Wood, with contributions
    by Benjamin J. Cairns. These numerical routines first appeared in Simon
    Wood's 'solv95' program. This package includes a vignette and a complete
    user's guide. 'PBSddesolve' originally appeared on CRAN under the name
    'ddesolve'. That version is no longer supported. The current name emphasizes
    a close association with other PBS packages, particularly 'PBSmodelling'.
License GPL (>= 2)
URL https://github.com/pbs-software/pbs-ddesolve
Repository CRAN
Date/Publication 2016-09-20 22:20:50

R topics documented:

dde ................................................................. 2
pastvalue ...................................................... 4
PBSddesolve .................................................. 5

Index 6
Description

A solver for systems of delay differential equations based on numerical routines from Simon Wood’s solv95 program. This solver is also capable of solving systems of ordinary differential equations.

Please see the included demos for examples of how to use dde.

To view available demos run demo(package="PBSddesolve").
The supplied demos require that the R package PBSmodelling be installed.

Usage

```r
dde(y, times, func, parms=NULL, switchfunc=NULL, mapfunc=NULL, 
    tol=1e-08, dt=0.1, hbsize=10000)
```

Arguments

- **y**: Vector of initial values of the DDE system. The size of the supplied vector determines the number of variables in the system.
- **times**: Numeric vector of specific times to solve.
- **func**: A user supplied function that computes the gradients in the DDE system at time \( t \). The function must be defined using the arguments: \((t, y)\) or \((t, y, \text{parms})\), where \( t \) is the current time in the integration, \( y \) is a vector of the current estimated variables of the DDE system, and \( \text{parms} \) is any R object representing additional parameters (optional).
  
  The argument `func` must return one of the two following return types:
  1) a vector containing the calculated gradients for each variable; or
  2) a list with two elements - the first a vector of calculated gradients, the second a vector (possibly named) of values for a variable specified by the user at each point in the integration.

- **parms**: Any constant parameters to pass to `func`, `switchfunc`, and `mapfunc`.
- **switchfunc**: An optional function that is used to manipulate state values at given times. The switch function takes the arguments \((t, y)\) or \((t, y, \text{parms})\) and must return a numeric vector. The size of the vector determines the number of switches used by the model. As values of `switchfunc` pass through zero (from positive to negative), a corresponding call to `mapfunc` is made, which can then modify any state value.

- **mapfunc**: If `switchfunc` is defined, then a map function must also be supplied with arguments \((t, y, \text{switch\_id})\) or \((t, y, \text{switch\_id}, \text{parms})\), where \( t \) is the time, \( y \) are the current state values, \( \text{switch\_id} \) is the index of the triggered switch, and \( \text{parms} \) are additional constant parameters.

- **tol**: Maximum error tolerated at each time step (as a proportion of the state variable concerned).
Maximum initial time step.

History buffer size required for solving DDEs.

The user supplied function \texttt{func} can access past values (lags) of \texttt{y} by calling the \texttt{pastvalue} function. Past gradients are accessible by the \texttt{pastgradient} function. These functions can only be called from \texttt{func} and can only be passed values of \texttt{t} greater or equal to the start time, but less than the current time of the integration point. For example, calling \texttt{pastvalue(t)} is not allowed, since these values are the current values which are passed in as \texttt{y}.

A data frame with one column for \texttt{t}, a column for every variable in the system, and a column for every additional value that may (or may not) have been returned by \texttt{func} in the second element of the list.

If the initial \texttt{y} values parameter was named, then the solved values column will use the same names. Otherwise \texttt{y1}, \texttt{y2}, ... will be used.

If \texttt{func} returned a list, with a named vector as the second element, then those names will be used as the column names. If the vector was not named, then \texttt{extra1}, \texttt{extra2}, ... will be used.

\texttt{pastvalue}

### Examples

```r
# This is just a single example of using dde.
# For more examples see demo(package="PBSddesolve")
# the demos require the package PBSmodelling

require(PBSddesolve)
local(env=.PBSddeEnv, expr={
  #create a func to return dde gradient
  yprime <- function(t,y,parms) {
    if (t < parms$tau)
      lag <- parms$initial
    else
      lag <- pastvalue(t - parms$tau)
    return(c(y1,y2))
  }

  #define initial values and parameters
  yinit <- c(1,1)
  parms <- list(tau=3, a=2, m=-10, initial=yinit)
})
```
# solve the dde system
yout <- dde(y=yinit, times=seq(0, 30, 0.1), func=yprime, parms=parms)

# and display the results
plot(yout$time, yout$y1, type="l", col="red", xlab="t", ylab="y",
     ylim=c(min(yout$y1, yout$y2), max(yout$y1, yout$y2))
lines(yout$time, yout$y2, col="blue")
legend("topleft", legend = c("y1", "y2"), lwd=2, lty = 1,
      xjust = 1, yjust = 1, col = c("red","blue"))
}

---

pastvalue | Retrieve Past Values (lags) During Gradient Calculation

### Description

These routines provides access to variable history at lagged times. The lagged time \( t \) must not be less than \( t_0 \), nor should it be greater than the current time of gradient calculation. The routine cannot be directly called by a user, and will only work during the integration process as triggered by the dde routine.

### Usage

```r
pastvalue(t)
pastgradient(t)
```

### Arguments

- `t` Access history at time \( t \).

### Value

Vector of variable history at time \( t \).

### See Also

dde
**Description**

A solver for systems of delay differential equations based on numerical routines from Simon Wood’s solv95 program. This solver is also capable of solving systems of ordinary differential equations.

**Details**

Please see the user guide `PBSddesolve-UG.pdf`, located in R’s library directory `./library/PBSddesolve/doc`, for a comprehensive overview.

**Author(s)**

Alex Couture-Beil <alex@mofo.ca>
Jon T. Schnute <schnutej-dfo@shaw.ca>
Rowan Haigh <rowan.haigh@dfo-mpo.gc.ca>

Maintainer: Rowan Haigh <rowan.haigh@dfo-mpo.gc.ca>

**References**


**See Also**

dde
Index

*Topic math
    dde, 2
    pastvalue, 4
*Topic package
    PBSddesolve, 5

dde, 2, 4, 5

pastgradient, 3
pastgradient (pastvalue), 4
pastvalue, 3, 4
PBSddesolve, 5
PBSddesolve-package (PBSddesolve), 5