Package: pCODE (via r-universe)

August 20, 2024

Type Package

Title Estimation of an Ordinary Differential Equation Model by Parameter Cascade Method

Version 0.9.4

Imports fda, pracma, MASS, deSolve, stats

Depends R (>= 3.5.0)

Description An implementation of the parameter cascade method Ramsay, J. O., Hooker,G., Campbell, D., and Cao, J. (2007) <doi:10.1111/j.1467-9868.2007.00610.x> for estimating ordinary differential equation models with missing or complete observations. It combines smoothing method and profile estimation to estimate any non-linear dynamic system. The package also offers variance estimates for parameters of interest based on either bootstrap or Delta method.

URL https://github.com/alex-haixuw/PCODE

License GPL

Encoding UTF-8

Suggests knitr, rmarkdown, Hmisc, testthat (>= 2.1.0)

VignetteBuilder knitr

RoxygenNote 6.1.1

Repository https://alex-haixuw.r-universe.dev

RemoteUrl https://github.com/alex-haixuw/pcode

RemoteRef HEAD

RemoteSha ecbfc5ef69fcb5293d91c7368da9273f3cde8ffd

Contents

bootsvar 2
deltavar 3
innerobj 4

innerobj_lkh 4
innerobj_lkh_1d 5
innerobj_multi 6
innerobj_multi_missing 6
nls_optimize 7
nls_optimize.inner 8
outterobj 8
outterobj_lkh 9
outterobj_lkh_1d 9
outterobj_multi_missing 10
outterobj_multi_nls 11
pcode 11
pcode_1d 13
pcode_lkh 14
pcode_lkh_1d 15
pcode_missing 16
prepare_basis 18
tunelambda 18

Index **20**

`bootsvar` *Bootstrap variance estimator of structural parameters.*

Description

Obtaining an estimate of variance for structural parameters by bootstrap method.

Usage

```
bootsvar(data, time, ode.model,par.names,state.names, likelihood.fun = NULL,
       par.initial, basis.list, lambda = NULL,bootsrep,controls = NULL)
```

Arguments

`data`	A data frame or a matrix contain observations from each dimension of the ODE model.
`time`	A vector contain observation times or a matrix if time points are different between dimensions.
`ode.model`	An R function that computes the time derivative of the ODE model given observations of states variable and structural parameters.
`par.names`	The names of structural parameters defined in the 'ode.model'.
`state.names`	The names of state variables defined in the 'ode.model'.
`likelihood.fun`	A likelihood function passed to PCODE in case of that the error termsdevtools::document()do not have a Normal distribution.
`par.initial`	Initial value of structural parameters to be optimized.

basis.list	A list of basis objects for smoothing each dimension's observations. Can be the same or different across dimensions.
lambda	Penalty parameter.
bootsrep	Bootstrap sample to be used for estimating variance.
controls	A list of control parameters. Same as the controls in pcode.

Value

boots.var The bootstrap variance of each structural parameters.

`deltavar` *Numeric estimation of variance of structural parameters by Delta method.*

Description

Obtaining variance of structural parameters by Delta method.

Usage

```
deltavar(data, time, ode.model,par.names,state.names,
     likelihood.fun, par.initial, basis.list, lambda,stepsize,y_stepsize,controls)
```

Arguments

data	A data frame or a matrix contain observations from each dimension of the ODE model.
time	A vector contain observation times or a matrix if time points are different between dimensions.
ode.model	An R function that computes the time derivative of the ODE model given observations of states variable and structural parameters.
par.names	The names of structural parameters defined in the 'ode.model'.
state.names	The names of state variables defined in the 'ode.model'.
likelihood.fun	A likelihood function passed to PCODE in case of that the error termsdevtools::document()do not have a Normal distribution.
par.initial	Initial value of structural parameters to be optimized.
basis.list	A list of basis objects for smoothing each dimension's observations. Can be the same or different across dimensions.
lambda	Penalty parameter.
stepsize	Stepsize used in estimating partial derivatives with respect to structural parameters for the Delta method.
y_stepsize	Stepsize used in estimating partial derivatives with respect to observations for the Delta method.
controls	A list of control parameters. Same as the controls in pcode.

Value

par.var The variance of structural parameters obtained by Delta method.

`innerobj` *Inner objective function (Single dimension version)*

Description

An objective function combines the sum of squared error of basis expansion estimates and the penalty controls how those estimates fail to satisfies the ODE model

Usage

```
innerobj(basis_coef, ode.par, input, derive.model,NLS)
```

Arguments

`basis_coef`	Basis coefficients for interpolating observations given a basis object.
`ode.par`	Structural parameters of the ODE model.
`input`	Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..
`derive.model`	The function defines the ODE model and is the same as the ode.model in 'pcode'
`NLS`	Default is `TRUE` so the function returns vector of residuals, and otherwise returns sum of squared errors.

Value

`residual.vec`	Vector of residuals and evaluation of penalty function on quadrature points for approximating the integral.

`innerobj_lkh` *Inner objective function (likelihood and multiple dimension version)*

Description

An objective function combines the likelihood or loglikelihood of errors from each dimension of state variables and the penalty controls how the state estimates fail to satisfy the ODE model.

Usage

```
innerobj_lkh(basis_coef, ode.par, input, derive.model, likelihood.fun)
```

Arguments

basis_coef	Basis coefficients for interpolating observations given a basis boject.
ode.par	Structural parameters of the ODD model.
input	Contains dependencies for the optimization, including observations, ode penalty, and etc..
derive.model	The function defines the ODE model and is the same as the ode.model in 'pcode'.
likelihood.fun	The likelihood or loglikelihood function of the errors.

Value

obj.eval The evaluation of the inner objective function.

---

innerobj_lkh_1d *Inner objective function (Likelihood and Single dimension version)*

Description

An objective function combines the likelihood or loglikelihood of errors from each dimension of state variables and the penalty controls how the state estimates fail to satisfy the ODE model.

Usage

```
innerobj_lkh_1d(basis_coef, ode.par, input, derive.model, likelihood.fun)
```

Arguments

basis_coef	Basis coefficients for interpolating observations given a basis boject.
ode.par	Structural parameters of the ODD model.
input	Contains dependencies for the optimization, including observations, ode penalty, and etc..
derive.model	The function defines the ODE model and is the same as the ode.model in 'pcode'.
likelihood.fun	The likelihood or loglikelihood function of the errors.

Value

obj.eval The evaluation of the inner objective function.

innerobj_multi *Inner objective function (multiple dimension version)*

Description

An objective function combines the sum of squared error of basis expansion estimates and the penalty controls how those estimates fail to satisfies the ODE model

Usage

```
innerobj_multi(basis_coef, ode.par, input, derive.model,NLS)
```

Arguments

basis_coef	Basis coefficients for interpolating observations given a basis object.
ode.par	Structural parameters of the ODE model.
input	Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..
derive.model	The function defines the ODE model and is the same as the ode.model in pcode.
NLS	Default is TRUE so the function returns vector of residuals, and otherwise returns sum of squared errors.

Value

residual.vec	Vector of residuals and evaluation of penalty function on quadrature points for approximating the integral.

innerobj_multi_missing *Inner objective function (multiple dimension version with unobserved state variables)*

Description

An objective function combines the sum of squared error of basis expansion estimates and the penalty controls how those estimates fail to satisfies the ODE model

Usage

```
innerobj_multi_missing(basis_coef, ode.par, input, derive.model,NLS)
```

Arguments

basis_coef	Basis coefficients for interpolating observations given a basis object.
ode.par	Structural parameters of the ODE model.
input	Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..
derive.model	The function defines the ODE model and is the same as the ode.model in 'pcode'
NLS	Default is TRUE so the function returns vector of residuals, and otherwise returns sum of squared errors.

Value

residual.vec	Vector of residuals and evaluation of penalty function on quadrature points for approximating the integral.

---

nls_optimize *Optimizer for non-linear least square problems*

Description

Obtain the solution to minimize the sum of squared errors of the defined function fun by levenberg-marquardt method. Adapted from PRACMA package.

Usage

```
nls_optimize(fun, x0, ..., options,verbal)
```

Arguments

fun	The function returns the vector of weighted residuals.
x0	The initial value for optimization.
...	Parameters to be passed for fun
options	Additional optimization controls.
verbal	Default = 1 for printing iteration and other for suppressing

Value

par	The solution to the non-linear least square problem, the same size as x0

nls_optimize.inner *Optimizer for non-linear least square problems (for inner objective functions)*

Description

Obtain the solution to minimize the sum of squared errors of the defined function `fun` by levenberg-marquardt method. Adapted from PRACMA package.

Usage

```
nls_optimize.inner(fun, x0, ..., options)
```

Arguments

`fun`	The function returns the vector of weighted residuals.
`x0`	The initial value for optimization.
`...`	Parameters to be passed for `fun`
`options`	Additional optimization controls.

Value

`par`	The solution to the non-linear least square problem, the same size as `x0`

outterobj *Outter objective function (Single dimension version)*

Description

An objective function of the structural parameter computes the measure of fit.

Usage

```
outterobj(ode.parameter, basis.initial, derivative.model, inner.input, NLS)
```

Arguments

`ode.parameter`	Structural parameters of the ODE model.
`basis.initial`	Initial values of the basis coefficients for nonlinear least square optimization.
`derivative.model`	The function defines the ODE model and is the same as the ode.model in 'pcode'
`inner.input`	Input that will be passed to the inner objective function. Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..
`NLS`	Default is `TRUE` so the function returns vector of residuals, and otherwise returns sum of squared errors.

Value

residual	Vector of residuals and evaluation of penalty function on quadrature points for approximating the integral.

---

outterobj_lkh *Outter objective function (likelihood and multiple dimension version)*

Description

An objective function of the structural parameter computes the measure of fit.

Usage

```
outterobj_lkh(ode.parameter, basis.initial, derivative.model, likelihood.fun, inner.input)
```

Arguments

ode.parameter	Structural parameters of the ODE model.
basis.initial	Initial values of the basis coefficients for nonlinear least square optimization.
derivative.model	The function defines the ODE model and is the same as the ode.model in 'pcode'
likelihood.fun	The likelihood or loglikelihood function of the errors.
inner.input	Input that will be passed to the inner objective function. Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..

Value

neglik The negative of the likelihood or the loglikelihood function that will be passed further to the 'optim' function.

---

outterobj_lkh_1d *Outter objective function (likelihood and single dimension version)*

Description

An objective function of the structural parameter computes the measure of fit.

Usage

```
outterobj_lkh_1d(ode.parameter, basis.initial,
                        derivative.model, likelihood.fun, inner.input)
```

Arguments

ode.parameter	Structural parameters of the ODE model.
basis.initial	Initial values of the basis coefficients for nonlinear least square optimization.
derivative.model	The function defines the ODE model and is the same as the ode.model in 'pcode'
likelihood.fun	The likelihood or loglikelihood function of the errors.
inner.input	Input that will be passed to the inner objective function. Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..

Value

neglik The negative of the likelihood or the loglikelihood function that will be passed further to the 'optim' function.

---

outterobj_multi_missing

Outter objective function (multiple dimension version with unobserved state variables)

Description

An objective function of the structural parameter computes the measure of fit for the basis expansion.

Usage

```
outterobj_multi_missing(ode.parameter, basis.initial, derivative.model, inner.input, NLS)
```

Arguments

ode.parameter	Structural parameters of the ODE model.
basis.initial	Initial values of the basis coefficients for nonlinear least square optimization.
derivative.model	The function defines the ODE model and is the same as the ode.model in 'pcode'
inner.input	Input that will be passed to the inner objective function. Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..
NLS	Default is TRUE so the function returns vector of residuals, and otherwise returns sum of squared errors.

Value

residual	Vector of residuals and evaluation of penalty function on quadrature points for approximating the integral.

outterobj_multi_nls *Outter objective function (multiple dimension version)*

Description

An objective function of the structural parameter computes the measure of fit for the basis expansion.

Usage

```
outterobj_multi_nls(ode.parameter, basis.initial, derivative.model, inner.input, NLS)
```

Arguments

ode.parameter	Structural parameters of the ODE model.
basis.initial	Initial values of the basis coefficients for nonlinear least square optimization.
derivative.model	The function defines the ODE model and is the same as the ode.model in pcode.
inner.input	Input that will be passed to the inner objective function. Contains dependencies for the optimization, including observations, penalty parameter lambda, and etc..
NLS	Default is TRUE so the function returns vector of residuals, and otherwise returns sum of squared errors.

Value

residual	Vector of residuals and evaluation of penalty function on quadrature points for approximating the integral.

pcode *Parameter Cascade Method for Ordinary Differential Equation Models*

Description

Obtain estimates of both structural and nuisance parameters of an ODE model by parameter cascade method.

Usage

```
pcode(data, time, ode.model, par.names, state.names,
             likelihood.fun, par.initial, basis.list,lambda,controls)
```

Arguments

data	A data frame or a matrix contain observations from each dimension of the ODE model.
time	A vector contain observation times or a matrix if time points are different between dimensions.
ode.model	An R function that computes the time derivative of the ODE model given observations of states variable and structural parameters.
par.names	The names of structural parameters defined in the 'ode.model'.
state.names	The names of state variables defined in the 'ode.model'.
likelihood.fun	A likelihood function passed to PCODE in case of that the error terms do not have a Normal distribution.
par.initial	Initial value of structural parameters to be optimized.
basis.list	A list of basis objects for smoothing each dimension's observations. Can be the same or different across dimensions.
lambda	Penalty parameter for controling the fidelity of interpolation.
controls	A list of control parameters. See Details.

Details

The controls argument is a list providing addition inputs for the nonlinear least square optimizer or general optimizer optim:

nquadpts Determine the number of quadrature points for approximating an integral. Default is 101.

smooth.lambda Determine the smoothness penalty for obtaining initial value of nuisance parameters.

tau Initial value of Marquardt parameter. Small values indicate good initial values for structural parameters.

tolx Tolerance for parameters of objective functions. Default is set at 1e-6.

tolg Tolerance for the gradient of parameters of objective functions. Default is set at 1e-6.

maxeval The maximum number of evaluation of the outter optimizer. Default is set at 20.

Value

structural.par	The structural parameters of the ODE model.
nuisance.par	The nuisance parameters or the basis coefficients for interpolating observations.

Examples

```
library(fda)
library(deSolve)
library(MASS)
library(pracma)
#Simple ode model example
#define model parameters
```

```
model.par   <- c(theta = c(0.1))
#define state initial value
state       <- c(X     = 0.1)
#Define model for function 'ode' to numerically solve the system
ode.model <- function(t, state,parameters){
 with(as.list(c(state,parameters)),
      {
        dX <- theta*X*(1-X/10)
        return(list(dX))
      })
}
#Observation time points
times <- seq(0,100,length.out=101)
#Solve the ode model
desolve.mod <- ode(y=state,times=times,func=ode.model,parms = model.par)
#Prepare for doing parameter cascading method
#Generate basis object for interpolation and as argument of pcode
#21 konts equally spaced within [0,100]
knots <- seq(0,100,length.out=21)
#order of basis functions
norder <- 4
#number of basis funtions
nbasis <- length(knots) + norder - 2
#creating Bspline basis
basis  <- create.bspline.basis(c(0,100),nbasis,norder,breaks = knots)
#Add random noise to ode solution for simulating data
nobs  <- length(times)
scale <- 0.1
noise <- scale*rnorm(n = nobs, mean = 0 , sd = 1)
observation <- desolve.mod[,2] + noise
#parameter estimation
pcode(data = observation, time = times, ode.model = ode.model,
                     par.initial = 0.1, par.names = 'theta',state.names = 'X',
                     basis.list = basis, lambda = 1e2)
```

---

pcode_1d *Parameter Cascade Method for Ordinary Differential Equation Models (Single dimension version)*

Description

Obtain estiamtes of structural parameters of an ODE model by parameter cascade method.

Usage

```
pcode_1d(data, time, ode.model, par.initial,par.names, basis,lambda,controls = list())
```

Arguments

data	A data frame or a vector contains observations from the ODE model.
time	The vector contain observation times.
ode.model	Defined R function that computes the time derivative of the ODE model given observations of states variable.
par.initial	Initial value of structural parameters to be optimized.
par.names	The names of structural parameters defined in the 'ode.model'.
basis	A basis objects for smoothing observations.
lambda	Penalty parameter.
controls	A list of control parameters. See 'Details'.

Value

structural.par	The structural parameters of the ODE model.
nuisance.par	The nuisance parameters or the basis coefficients for interpolating observations.

---

pcode_lkh *pcode_lkh (likelihood and multiple dimension version)*

Description

Obtain estimates of both structural and nuisance parameters of an ODE model by parameter cascade method.

Usage

```
pcode_lkh(data, likelihood.fun, time, ode.model, par.names,
                 state.names, par.initial, basis.list, lambda, controls)
```

Arguments

data	A data frame or a matrix contain observations from each dimension of the ODE model.
likelihood.fun	A function computes the likelihood or the loglikelihood of the errors.
time	A vector contains observation ties or a matrix if time points are different between dimesion.
ode.model	An R function that computes the time derivative of the ODE model given observations of states variable and structural parameters.
par.names	The names of structural parameters defined in the 'ode.model'.
state.names	The names of state variables defined in the 'ode.model'.
par.initial	Initial value of structural parameters to be optimized.
basis.list	A list of basis objects for smoothing each dimension's observations. Can be the same or different across dimensions.
lambda	Penalty parameter.
controls	A list of control parameters. See 'Details'.

Details

The controls argument is a list providing addition inputs for the nonlinear least square optimizer:

- nquadpts Determine the number of quadrature points for approximating an integral. Default is 101.
- smooth.lambda Determine the smoothness penalty for obtaining initial value of nuisance parameters.
- tau Initial value of Marquardt parameter. Small values indicate good initial values for structural parameters.
- tolx Tolerance for parameters of objective functions. Default is set at 1e-6.
- tolg Tolerance for the gradient of parameters of objective functions. Default is set at 1e-6.
- maxeval The maximum number of evaluation of the optimizer. Default is set at 20.

Value

structural.par	The structural parameters of the ODE model.
nuisance.par	The nuisance parameters or the basis coefficients for interpolating observations.

---

pcode_lkh_1d *Parameter Cascade Method for Ordinary Differential Equation Models (likelihood and Single dimension version)*

Description

Obtain estimates of both structural and nuisance parameters of an ODE model by parameter cascade method.

Usage

```
pcode_lkh_1d(data, likelihood.fun, time, ode.model, par.names,
                    state.names, par.initial, basis.list, lambda, controls)
```

Arguments

data	A data frame or a matrix contain observations from each dimension of the ODE model.
likelihood.fun	A function computes the likelihood or the loglikelihood of the errors.
time	A vector contains observation ties or a matrix if time points are different between dimesion.
ode.model	An R function that computes the time derivative of the ODE model given observations of states variable and structural parameters.
par.names	The names of structural parameters defined in the 'ode.model'.
state.names	The names of state variables defined in the 'ode.model'.

par.initial	Initial value of structural parameters to be optimized.
basis.list	A list of basis objects for smoothing each dimension's observations. Can be the same or different across dimensions.
lambda	Penalty parameter.
controls	A list of control parameters. See 'Details'.

Details

The controls argument is a list providing addition inputs for the nonlinear least square optimizer:

- nquadpts Determine the number of quadrature points for approximating an integral. Default is 101.
- smooth.lambda Determine the smoothness penalty for obtaining initial value of nuisance parameters.
- tau Initial value of Marquardt parameter. Small values indicate good initial values for structural parameters.
- tolx Tolerance for parameters of objective functions. Default is set at 1e-6.
- tolg Tolerance for the gradient of parameters of objective functions. Default is set at 1e-6.
- maxeval The maximum number of evaluation of the optimizer. Default is set at 20.

Value

structural.par	The structural parameters of the ODE model.
nuisance.par	The nuisance parameters or the basis coefficients for interpolating observations.

---

pcode_missing *Parameter Cascade Method for Ordinary Differential Equation Models with missing state variable*

Description

Obtain estiamtes of both structural and nuisance parameters of an ODE model by parameter cascade method when the dynamics are partially observed.

Usage

```
pcode_missing(data, time, ode.model, par.names, state.names,
                    likelihood.fun,par.initial, basis.list,lambda,controls)
```

Arguments

data	A data frame or a matrix contain observations from each dimension of the ODE model.
time	A vector contain observation times or a matrix if time points are different between dimensions.
ode.model	An R function that computes the time derivative of the ODE model given observations of states variable and structural parameters.
par.names	The names of structural parameters defined in the 'ode.model'.
state.names	The names of state variables defined in the 'ode.model'.
likelihood.fun	A likelihood function passed to PCODE in case of that the error termsdevtools::document()do not have a Normal distribution.
par.initial	Initial value of structural parameters to be optimized.
basis.list	A list of basis objects for smoothing each dimension's observations. Can be the same or different across dimensions.
lambda	Penalty parameter.
controls	A list of control parameters. See Details.

Details

The controls argument is a list providing addition inputs for the nonlinear least square optimizer or general optimizer optim:

- nquadpts Determine the number of quadrature points for approximating an integral. Default is 101.
- smooth.lambda Determine the smoothness penalty for obtaining initial value of nuisance parameters.
- tau Initial value of Marquardt parameter. Small values indicate good initial values for structural parameters.
- tolx Tolerance for parameters of objective functions. Default is set at 1e-6.
- tolg Tolerance for the gradient of parameters of objective functions. Default is set at 1e-6.
- maxeval The maximum number of evaluation of the optimizer. Default is set at 20.

Value

structural.par	The structural parameters of the ODE model.
nuisance.par	The nuisance parameters or the basis coefficients for interpolating observations.

prepare_basis *Evaluate basis objects over observation times and quadrature points*

Description

Calculate all basis functions over observation time points and store them as columns in a single matrix for each dimension. Also include first and second order derivative. Repeat over quadrature points.

Usage

```
prepare_basis(basis, times, nquadpts)
```

Arguments

basis	A basis object.
times	The vector contain observation times for corresponding dimension.
nquadpts	Number of quadrature points will be used later for approximating integrals.

Value

Phi.mat	Evaluations of all basis functions stored as columns in the matrix.
Qmat	Evaluations of all basis functions over quadrature points stored as columns in the matrix.
Q.D1mat	Evaluations of first order derivative all basis functions over quadrature points stored as columns in the matrix.
Q.D2mat	Evaluations of second order derivative all basis functions over quadrature points stored as columns in the matrix.
quadts	Quadrature points.
quadwts	Quadrature weights.

tunelambda *Find optimial penalty parameter lambda by cross-validation.*

Description

Obtain the optimal sparsity parameter given a search grid based on cross validation score with replications.

Usage

```
tunelambda(data, time, ode.model, par.names, state.names,
           par.initial, basis.list,lambda_grid,cv_portion,kfolds, rep,controls)
```

Arguments

data	A data frame or matrix contrain observations from each dimension of the ODE model.
time	The vector contain observation times or a matrix if time points are different between dimensions.
ode.model	Defined R function that computes the time derivative of the ODE model given observations of states variable.
par.names	The names of structural parameters defined in the 'ode.model'.
state.names	The names of state variables defined in the 'ode.model'.
par.initial	Initial value of structural parameters to be optimized.
basis.list	A list of basis objects for smoothing each dimension's observations. Can be the same or different across dimensions.
lambda_grid	A search grid for finding the optimial sparsity parameter lambda.
cv_portion	A number indicating the proportion of data will be saved for doing cross validation. Default is set at 5 as minimum.
kfolds	A number indicating the number of folds the data should be seprated into.
rep	A integer controls the number of replication of doing cross-validation for each penalty parameter.
controls	A list of control parameters. See 'Details'.

Value

lambda_grid	The original input vector of a search grid for the optimal lambda.
cv.score	The matrix contains the cross validation score for each lambda of each replication

Index

bootsvar, 2

deltavar, 3

innerobj, 4
innerobj_lkh, 4
innerobj_lkh_1d, 5
innerobj_multi, 6
innerobj_multi_missing, 6

nls_optimize, 7
nls_optimize.inner, 8

outterobj, 8
outterobj_lkh, 9
outterobj_lkh_1d, 9
outterobj_multi_missing, 10
outterobj_multi_nls, 11

pcode, 11
pcode_1d, 13
pcode_lkh, 14
pcode_lkh_1d, 15
pcode_missing, 16
prepare_basis, 18

tunelambda, 18