Rdocumentation

April 29, 2021

`Group_specific_Var_AUC_estimation`
Variance of the Area Under The Curve of Group-Specific Polynomial Marginal Dynamics

---

Description

This function calculates the variance of the area under the curve of marginal dynamics modeled by group-structured polynomials or B-spline curves in Mixed-Effects models

Usage

```
Group_specific_Var_AUC_estimation(
  MEM_Pol_group,time,Groups = NULL,
  method = "trapezoid",Averaged = FALSE
)
```

Arguments

`MEM_Pol_group` A list with similar structure than the output provided by the function MEM_Polynomial_Group_struct

A list containing:

- `Model_estimation`: a list containing at least 2 elements:
  1. the vector of the marginal (fixed) parameters estimates (at least for the groups whose AUC is to estimate), labeled *'beta'*.
  2. the variance-covariance matrix of these parameters, labeled *'varFix'* (see MEM_Polynomial_Group_structure for details about the parameter order).
- `Model_features`: a list of at least 2 elements:
  1. `Groups`: a vector indicating the names of the groups whose fixed parameters are given.
  2. `Marginal.dyn.feature`: a list summarizing the features of the marginal dynamics defined in the model:
     - `dynamic.type`: a character scalar indicating the chosen type of marginal dynamics. Options are 'polynomial' or 'spline'.

- `intercept`: a logical vector summarizing choices about global and group-specific intercepts (Number of groups + 1) elements whose elements are named as ('global.intercept','group.intercept1', ..., 'group.interceptG') if G Groups are defined in `MEM_Pol_group`. For each element of the vector, if TRUE, the considered intercept is considered as included in the model (see *Examples*).

If `dynamic.type` is defined as 'polynomial':

- `polynomial.degree`: an integer vector indicating the degree of polynomial functions, one value for each group.

If `dynamic.type` is defined as 'spline':

- `spline.degree`: an integer vector indicating the degree of B-spline curves, one for each group.
- `knots`: a list of group-specific internal knots used to build B-spline basis (one numerical vector for each group) (see bs for more details).
- `df`: a numerical vector of group-specific degrees of freedom used to build B-spline basis, (one for each group).
- `boundary.knots`: a list of group-specific boundary knots used to build B-spline basis (one vector for each group) (see bs for more details).

`time` a numerical vector of time points (x-axis coordinates) or a list of numerical vectors (with as much elements than the number of groups in `Groups`).

`Groups` a vector indicating the names of the groups belonging to the set of groups involved in `MEM_Pol_group` for which we want to estimate the AUC (a subset or the entire set of groups involved in the model can be considered). If NULL (default), the AUC for all the groups involved the MEM is calculated.

`method` a character scalar indicating the interpolation method to use to estimate the AUC. Options are 'trapezoid' (default), 'lagrange' and 'spline'. In this version, the 'spline' interpolation is implemented with the "not-a-knot" spline boundary conditions.

`Averaged` a logical scalar. If TRUE, the function return the normalized AUC (nAUC) computed as the AUC divided by the range of the time calculation. If FALSE (default), the classic AUC is calculated. (See Group_specific_AUC_estimation for more details about calculation)

Value

A numerical vector containing the estimation of the variance of the AUC (or nAUC) for each group defined in the `Groups` vector.

See Also

bs, `MEM_Polynomial_Group_structure`

Examples

```
# Download of data
data("HIV_Simu_Dataset_Delta01_cens")
data <- HIV_Simu_Dataset_Delta01_cens

# Change factors in character vectors
data$id <- as.character(data$id) ; data$Group <- as.character(data$Group)
```

```
# Example 1: We consider the variable \code{MEM_Pol_Group} as the output of our function
# \link[AUCcomparison]{MEM_Polynomial_Group_structure}
MEM_estimation <- MEM_Polynomial_Group_structure(y=data$VL,x=data$time,Group=data$Group,
                                                 Id=data$id,Cens=data$cens)

time_group1 <- unique(data$time[which(data$Group == "Group1")])
time_group2 <- unique(data$time[which(data$Group == "Group2")])

Var_AUC_estimation <- Group_specific_Var_AUC_estimation(MEM_Pol_group=MEM_estimation,
                                                  time=list(time_group1,time_group2))

# Example 2: We consider results of MEM estimation from another source.
# We have to give build the variable 'MEM_Pol_group' with the good structure
# We build the variable 'MEM_Pol_group.1' with the results of MEM estimation obtained for 2 groups
# Generation of random matrix
 Covariance_Matrix_1 <- matrix(rnorm(7*7,mean=0,sd=0.01),ncol=7,nrow=7)
# Transform the matrix into symmetric one
Covariance_Matrix_1 <- Covariance_Matrix_1 %*% t(Covariance_Matrix_1)
MEM_Pol_group.1 <- list(Model_estimation=Covariance_Matrix_1,
                        Model_features=list(Groups=c("Group1","Group2"),
                               Marginal.dyn.feature=list(dynamic.type="polynomial",
                                            intercept=c(global.intercept=TRUE,
                                      group.intercept1=FALSE,group.intercept2=FALSE),
                                            polynomial.degree=c(3,3))))

Var_AUC_estimation_G1.1 <- Group_specific_Var_AUC_estimation(MEM_Pol_group.1,
                                                 time=time_group1,Groups=c("Group1"))

# We build the variable 'MEM_Pol_group.2' with the results of MEM estimation obtained only for the
# group of interest (extraction)
# Generation of random matrix
Covariance_Matrix_2 <-  matrix(rnorm(4*4,mean=0,sd=0.01),ncol=4,nrow=4)
# Transform the matrix into a symmetric one
Covariance_Matrix_2 <- Covariance_Matrix_2 %*% t(Covariance_Matrix_2)
MEM_Pol_group.2 <- list(Model_estimation=Covariance_Matrix_2,
                        Model_features=list(Groups=c("Group1"),
                               Marginal.dyn.feature=list(dynamic.type="polynomial",
                                             intercept=c(global.intercept=TRUE,
                                             group.intercept1=FALSE),
                                             polynomial.degree=c(3))))

Var_AUC_estimation_G1.2 <- Group_specific_Var_AUC_estimation(MEM_Pol_group=MEM_Pol_group.2,
                                                       time=time_group1)
```

Index