Package ‘StempCens’

October 12, 2022

Type Package

Title Spatio-Temporal Estimation and Prediction for Censored/Missing Responses

Version 1.1.0

Description It estimates the parameters of a censored or missing data in spatio-temporal models using the SAEM algorithm (Delyon et al., 1999). This algorithm is a stochastic approximation of the widely used EM algorithm and an important tool for models in which the E-step does not have an analytic form. Besides the expressions obtained to estimate the parameters to the proposed model, we include the calculations for the observed information matrix using the method developed by Louis (1982). To examine the performance of the fitted model, case-deletion measure are provided.

License GPL (>= 2)

Imports Rcpp, stats, utils, mvtnorm, tmvtnorm, MCMCglmm, ggplot2, grid, distances, Rdpack

RdMacros Rdpack

Encoding UTF-8

RoxygenNote 7.1.1

LinkingTo Rcpp, RcppArmadillo

Suggests testthat

NeedsCompilation yes

Author Larissa A. Matos [aut, cre] (<https://orcid.org/0000-0002-2635-0901>), Katherine L. Valeriano [aut] (<https://orcid.org/0000-0001-6388-4753>), Victor H. Lachos [ctb] (<https://orcid.org/0000-0002-7239-2459>)

Maintainer Larissa A. Matos <larissa.amatos@gmail.com>

Repository CRAN

Date/Publication 2020-10-21 20:20:21 UTC

R topics documented:

`StempCens-package` *Spatio-Temporal Estimation and Prediction for Censored/Missing Responses*

Description

It estimates the parameters of a censored or missing data in spatio-temporal models using the SAEM algorithm (Delyon et al., 1999). This algorithm is a stochastic approximation of the widely used EM algorithm and an important tool for models in which the E-step does not have an analytic form. Besides the expressions obtained to estimate the parameters to the proposed model, we include the calculations for the observed information matrix using the method developed by Louis (1982). To examine the performance of the fitted model, case-deletion measure are provided.

Details

The functions provided are:

- `CovarianceM`: computes the spatio-temporal covariance matrix for balanced data.

- `EffectiveRange`: computes the effective range for an isotropic spatial correlation function.

- `EstStempCens`: returns the maximum likelihood estimates of the unknown parameters.

- `PredStempCens`: performs spatio-temporal prediction in a set of new `S` spatial locations for fixed time points.

- `CrossStempCens`: performs cross-validation, which measure the performance of the predictive model on new test dataset.

- `DiagStempCens`: returns measures and graphics for diagnostic analysis.

Author(s)

Larissa A. Matos (ORCID), Katherine L. Valeriano (ORCID) and Victor H. Lachos (ORCID)

Maintainer: Larissa A. Matos (<`larissa.amatos@gmail.com`>).

References

Cook R (1977). “Detection of influential observation in linear regression.” *Technometrics*, **19**(1), 15–18. doi: 10.1080/00401706.1977.10489493.

Delyon B, Lavielle M, Moulines E (1999). “Convergence of a stochastic approximation version of the EM algorithm.” *Annals of Statistics*, **27**(1), 94–128. doi: 10.1214/aos/1018031103.

Louis T (1982). "Finding the observed information matrix when using the EM algorithm." *Journal of the Royal Statistical Society: Series B (Methodological)*, **44**(2), 226–233. doi: 10.1111/j.2517-6161.1982.tb01203.x.

Zhu H, Lee S, Wei B, Zhou J (2001). "Case-deletion measures for models with incomplete data." *Biometrika*, **88**(3), 727–737. doi: 10.1093/biomet/88.3.727.

`CovarianceM` *Covariance matrix for spatio-temporal model*

Description

It computes the spatio-temporal covariance matrix for balanced data, i.e., when we have the same temporal indexes per location. To compute the spatial correlation it provides 5 functions: exponential, gaussian, matern, spherical and power exponential. To compute the temporal correlation is used an autocorrelation function of an AR(1) process.

Usage

```
CovarianceM(phi, rho, tau2, sigma2, distSpa, disTemp, kappa, type.S)
```

Arguments

`phi`	value of the spatial scaling parameter.
`rho`	value of the time scaling parameter.
`tau2`	value of the the nugget effect parameter.
`sigma2`	value of the partial sill.
`distSpa`	nxn spatial distance matrix without considering repetitions.
`disTemp`	TxT temporal distance matrix without considering repetitions.
`kappa`	parameter for all spatial covariance functions. In the case of exponential, gaussian and spherical function κ is equal to zero. For the power exponential function κ is a number between 0 and 2. For the matern correlation function is upper than 0.
`type.S`	type of spatial correlation function: '`exponential`' for exponential, '`gaussian`' for gaussian, '`matern`' for matern, '`pow.exp`' for power exponential and '`spherical`' for spherical function, respectively. See the analytical form of these functions in `EffectiveRange`.

Value

The function returns the $nTxnT$ spatio-temporal covariance matrix for balanced data.

Author(s)

Katherine L. Valeriano, Victor H. Lachos and Larissa A. Matos

Examples

```
# Initial parameter values
phi <- 5;     rho <- 0.45
tau2 <- 0.80; sigma2 <- 2
# Simulating data
n1 <- 10   # Number of spatial locations
n2 <- 5    # Number of temporal index
set.seed(1000)
x.co <- round(runif(n1,0,10),5)  # X coordinate
y.co <- round(runif(n1,0,10),5)  # Y coordinate
coord <- cbind(x.co,y.co)        # Cartesian coordinates without repetitions
time <- as.matrix(seq(1,n2))     # Time index without repetitions
# Covariance matrix
Ms <- as.matrix(dist(coord))     # Spatial distances
Mt <- as.matrix(dist(time))      # Temporal distances
Cov <- CovarianceM(phi,rho,tau2,sigma2,distSpa=Ms,disTemp=Mt,kappa=0,type.S="exponential")
```

CrossStempCens *Cross-Validation in spatio-temporal model with censored/missing responses*

Description

This function performs cross-validation in spatio-temporal model with censored/missing responses, which measure the performance of the predictive model on new test dataset. The cross-validation method for assessing the model performance is validation set approach (or data split).

Usage

```
CrossStempCens(Pred.StempCens, yObs.pre)
```

Arguments

`Pred.StempCens`	an object of class `Pred.StempCens` given as output by the `PredStempCens` function.
`yObs.pre`	a vector of the observed responses, the test data.

Value

`Bias`	bias prediction error.
`Mspe`	mean squared prediction error.
`Rmspe`	root mean squared prediction error.
`Mae`	mean absolute error.

Author(s)

Katherine L. Valeriano, Victor H. Lachos and Larissa A. Matos

See Also

EstStempCens, PredStempCens

Examples

```
## Not run:
# Initial parameter values
beta <- c(-1,1.50)
phi <- 5;     rho <- 0.6
tau2 <- 0.80; sigma2 <- 2
# Simulating data
n1 <- 7    # Number of spatial locations
n2 <- 5    # Number of temporal index
set.seed(400)
x.co <- round(runif(n1,0,10),9)    # X coordinate
y.co <- round(runif(n1,0,10),9)    # Y coordinate
coord <- cbind(x.co,y.co)          # Cartesian coordinates without repetitions
coord2 <- cbind(rep(x.co,each=n2),rep(y.co,each=n2)) # Cartesian coordinates with repetitions
time <- as.matrix(seq(1,n2))       # Time index without repetitions
time2 <- as.matrix(rep(time,n1))   # Time index with repetitions
x1 <- rexp(n1*n2,2)
x2 <- rnorm(n1*n2,2,1)
x  <- cbind(x1,x2)
media <- x%*%beta
# Covariance matrix
Ms <- as.matrix(dist(coord))   # Spatial distances
Mt <- as.matrix(dist(time))    # Temporal distances
Cov <- CovarianceM(phi,rho,tau2,sigma2,Ms,Mt,0,"gaussian")
# Data
require(mvtnorm)
y <- as.vector(rmvnorm(1,mean=as.vector(media),sigma=Cov))
data <- data.frame(coord2,time2,y,x)
names(data) <- c("x.coord","y.coord","time","yObs","x1","x2")
# Splitting the dataset
local.est  <- coord[c(1,2,4,5,6),]
data.est   <- data[data$x.coord%in%local.est[,1] & data$y.coord%in%local.est[,2],]
data.valid <- data[data$x.coord%in%coord[c(3,7),1] & data$y.coord%in%coord[c(3,7),2],]
# Censored
perc <- 0.20
y  <- data.est$yObs
aa <- sort(y);  bb <- aa[1:(perc*nrow(data.est))]
cutof <- bb[perc*nrow(data.est)]
cc <- matrix(1,nrow(data.est),1)*(y<=cutof)
y[cc==1] <- cutof
data.est <- data.frame(data.est[,-c(4,5,6)],y,cc,data.est[,c(5,6)])
names(data.est) <- c("x.coord","y.coord","time","yObs","censored","x1","x2")

# Estimation
y  <- data.est$yObs
x  <- cbind(data.est$x1,data.est$x2)
cc <- data.est$censored
time2  <- as.data.frame(data.est$time)
```

```
coord2 <- data.est[,1:2]
LI <- y; LI[cc==1] <- -Inf     # Left-censored
LS <- y
est_teste <- EstStempCens(y, x, cc, time2, coord2, LI, LS, init.phi=3.5, init.rho=0.5,
                 init.tau2=1,type.Data="balanced", method="nlminb", kappa=0,
                 type.S="gaussian", IMatrix=FALSE, M=20, perc=0.25, MaxIter=300,
                 pc=0.20)
# Prediction
locPre <- data.valid[,1:2]
timePre <- as.data.frame(data.valid$time)
xPre <- cbind(data.valid$x1,data.valid$x2)
pre_teste <- PredStempCens(est_teste, locPre, timePre, xPre)
class(pre_teste)

# Cross-validation
cross_teste <- CrossStempCens(pre_teste,data.valid$yObs)
cross_teste$Mspe # MSPE
## End(Not run)
```

DiagStempCens — *Diagnostic in spatio-temporal model with censored/missing responses*

Description

Return measures and graphics for diagnostic analysis in spatio-temporal model with censored/missing responses.

Usage

```
DiagStempCens(Est.StempCens, type.diag = "individual", diag.plot = TRUE, ck)
```

Arguments

Est.StempCens	an object of class Est.StempCens given as output by the EstStempCens function. In the EstStempCensfunction, IMatrix must be TRUE.
type.diag	type of diagnostic: 'individual' is related when one observation is deleted, 'time' is related when an entire time is deleted, 'location' is related when an entire location is deleted and 'all' the three cases ('individual', 'time' and 'location'). By default type.diag is individual.
diag.plot	TRUE or FALSE. It indicates if diagnostic plots must be showed. By default = TRUE.
ck	the value for ck considered in the benchmark value for the index plot: $mean(GD) + ck * sd(GD)$, where GD is the vector with all values of the diagnostic measures.

Details

This function uses the case deletion approach to study the impact of deleting one or more observations from the dataset on the parameters estimates, using the ideas of Cook (1977) and Zhu et al. (2001). The measure is defined by

$GD_i(\theta*) = (\theta * - \theta * [i])'[-Q * *(\theta|\theta*)](\theta * - \theta * [i]), i = 1, ...m,$

where $\theta*$ is the estimate of θ using the complete data, $\theta * [i]$ are the estimates obtained after deletion of the i-th observation (or group of observations) and $Q * *(\theta|\theta*)$ is the Hessian matrix.

We can eliminate an observation, an entire location or an entire time index.

Value

The function returns a list with the diagnostic measures.

If `type.diag` == `individual | time | location`**:** `GD` is a data.frame with the index value of the observation and the GD measure.

If `type.diag` == `all`**:** `GDind` is a data.frame with the index value of the observation and the GD measure for individual.

`GDtime` is a data.frame with the time index value and the GD measure for time.

`GDloc` is a data.frame with the side index value and the GD measure for location.

Author(s)

Katherine L. Valeriano, Victor H. Lachos and Larissa A. Matos

See Also

`EstStempCens`

Examples

```
## Not run:
# Initial parameter values
beta <- c(-1,1.5)
phi <- 3;   rho <- 0.40
tau2 <- 1;  sigma2 <- 2
# Simulating data
n1 <- 5    # Number of spatial locations
n2 <- 5    # Number of temporal index
set.seed(98765)
x.co <- round(runif(n1,0,10),9)   # X coordinate
y.co <- round(runif(n1,0,10),9)   # Y coordinate
coord <- cbind(x.co,y.co)         # Cartesian coordinates without repetitions
coord2 <- cbind(rep(x.co,each=n2),rep(y.co,each=n2)) # Cartesian coordinates with repetitions
time <- as.matrix(seq(1,n2))      # Time index without repetitions
time2 <- as.matrix(rep(time,n1))  # Time index with repetitions
x1 <- rexp(n1*n2,2)
x2 <- rnorm(n1*n2,2,1)
x  <- cbind(x1,x2)
media <- x%*%beta
```

```
# Covariance matrix
Ms  <- as.matrix(dist(coord))  # Spatial distances
Mt  <- as.matrix(dist(time))   # Temporal distances
Cov <- CovarianceM(phi,rho,tau2,sigma2,Ms,Mt,0,"exponential")
# Data
require(mvtnorm)
y <- as.vector(rmvnorm(1,mean=as.vector(media),sigma=Cov))
perc <- 0.20
aa <- sort(y); bb <- aa[((1-perc)*n1*n2+1):(n1*n2)]; cutof <- bb[1]
cc <- matrix(1,(n1*n2),1)*(y>=cutof)
y[cc==1] <- cutof
y[17] <- abs(y[17])+2*sd(y)
LI <- y
LS <- y; LS[cc==1] <- Inf    # Right-censored

# Estimation
set.seed(74689)
est <- EstStempCens(y, x, cc, time2, coord2, LI, LS, init.phi=2.5, init.rho=0.5, init.tau2=0.8,
          type.Data="balanced", method="nlminb", kappa=0, type.S="exponential",
          IMatrix=TRUE, lower.lim=c(0.01,-0.99,0.01), upper.lim=c(30,0.99,20), M=20,
          perc=0.25, MaxIter=300, pc=0.20)

# Diagnostic
set.seed(12345)
diag <- DiagStempCens(est, type.diag="time", diag.plot = TRUE, ck=1)
## End(Not run)
```

`EffectiveRange` *Effective range for some spatial correlation functions*

Description

It computes the effective range for an isotropic spatial correlation function, which is commonly defined to be the distance from which the correlation becomes small, typically below 0.05.

Usage

```
EffectiveRange(cor = 0.05, phi, kappa = 0, Sp.model = "exponential")
```

Arguments

`cor`	effective correlation to check for. By default = 0.05.
`phi`	spatial scaling parameter.
`kappa`	smoothness parameter, required by the matern and power exponential functions. By default = 0.
`Sp.model`	type of spatial correlation function: '`exponential`' for exponential, '`gaussian`' for gaussian, '`matern`' for matern, '`pow.exp`' for power exponential and '`spherical`' for spherical function, respectively. By default = `exponential`.

Details

The available isotropic spatial correlation functions are:

Exponential: $Corr(d) = exp{-d/\phi}$,

Gaussian: $Corr(d) = exp{-(d/\phi)^2}$,

Matern: $Corr(d) = 1/(2^(\kappa - 1)\Gamma(\kappa))(d/\phi)^\kappa K_\kappa(d/\phi)$,

Power exponential: $Corr(d) = exp{-(d/\phi)^\kappa}$,

Spherical: $Corr(d) = 1 - 1.5d/\phi + 0.5(d/\phi)^3$,

where d is the Euclidean distance between two observations, ϕ is the spatial scaling parameter, $\Gamma(.)$ is the gamma function, κ is the smoothness parameter and $K_\kappa(.)$ is the modified Bessel function of the second kind of order κ.

Value

The function returns the effective range, i.e., the approximate distance from which the spatial correlation is lower than `cor`.

Author(s)

Katherine L. Valeriano, Victor H. Lachos and Larissa A. Matos

Examples

```
phi <- 164.60
range1 <- EffectiveRange(0.05, phi, kappa=0, Sp.model="exponential")
range2 <- EffectiveRange(0.05, phi, kappa=1, Sp.model="pow.exp")
# Note that these functions are equivalent.
```

EstStempCens *ML estimation in spatio-temporal model with censored/missing responses*

Description

Return the maximum likelihood estimates of the unknown parameters of spatio-temporal model with censored/missing responses. The estimates are obtained using SAEM algorithm. The function also computes the observed information matrix using the method developed by Louis (1982). The types of censoring considered are left, right, interval or missing values.

Usage

```
EstStempCens(
  y,
  x,
  cc,
  time,
  coord,
  LI,
  LS,
  init.phi,
  init.rho,
  init.tau2,
  tau2.fixo = FALSE,
  type.Data = "balanced",
  method = "nlminb",
  kappa = 0,
  type.S = "exponential",
  IMatrix = TRUE,
  lower.lim = c(0.01, -0.99, 0.01),
  upper.lim = c(30, 0.99, 20),
  M = 20,
  perc = 0.25,
  MaxIter = 300,
  pc = 0.2,
  error = 1e-06
)
```

Arguments

`y`	a vector of responses.
`x`	a matrix or vector of covariates.
`cc`	a vector of censoring indicators. For each observation: `1` if censored/missing and `0` if non-censored/non-missing.
`time`	a vector of time.
`coord`	a matrix of coordinates of the spatial locations.
`LI`	lower limit of detection. For each observation: if non-censored/non-missing =`y`, if left-censored/missing =`-Inf` or =`LOD` if right/interval-censored.
`LS`	upper limit of detection. For each observation: if non-censored/non-missing =`y`, if right-censored/missing =`Inf` or =`LOD` if left/interval-censored.
`init.phi`	initial value of the spatial scaling parameter.
`init.rho`	initial value of the time scaling parameter.
`init.tau2`	initial value of the the nugget effect parameter.
`tau2.fixo`	`TRUE` or `FALSE`. Indicate if the nugget effect (τ^2) parameter must be fixed. By default = `FALSE`.
`type.Data`	type of the data: '`balanced`' for balanced data and '`unbalanced`' for unbalanced data. By default = `balanced`.
`method`	optimization method used to estimate (ϕ, ρ and τ^2): '`optim`' for the function `optim` and '`nlminb`' for the function `nlminb`. By default = `nlminb`.
`kappa`	parameter for all spatial covariance functions. In the case of exponential, gaussian and spherical function κ is equal to zero. For the power exponential function κ is a number between 0 and 2. For the matern correlation function is upper than 0.

`type.S`	type of spatial correlation function: '`exponential`' for exponential, '`gaussian`' for gaussian, '`matern`' for matern, '`pow.exp`' for power exponential and '`spherical`' for spherical function, respectively. Default is `exponential` function.
`IMatrix`	`TRUE` or `FALSE`. Indicate if the observed information matrix will be computed. By default = `TRUE`.
`lower.lim, upper.lim`	vectors of lower and upper bounds for the optimization method. If unspecified, the default is `c(0.01,-0.99,0.01)` for the lower bound and `c(30,0.99,20)` for the upper bound if tau2.fixo=`FALSE`.
`M`	number of Monte Carlo samples for stochastic approximation. By default = `20`.
`perc`	percentage of burn-in on the Monte Carlo sample. By default = `0.25`.
`MaxIter`	the maximum number of iterations of the SAEM algorithm. By default = `300`.
`pc`	percentage of iterations of the SAEM algorithm with no memory. By default = `0.20`.
`error`	the convergence maximum error. By default = `1e-6`.

Details

The spatio-temporal Gaussian model is giving by:

$Y(s_i,t_j) = \mu(s_i,t_j) + Z(s_i,t_j) + \epsilon(s_i,t_j),$

where the deterministic term $\mu(s_i,t_j)$ and the stochastic terms $Z(s_i,t_j)$, $\epsilon(s_i,t_j)$ can depend on the observed spatio-temporal indexes for $Y(s_i,t_j)$. We assume Z is normally distributed with zero-mean and covariance matrix $\Sigma_z = \sigma^2\Omega_{\phi\rho}$, where σ^2 is the partial sill, $\Omega_{\phi\rho}$ is the spatio-temporal correlation matrix,ϕ and ρ are the spatial and time scaling parameters; $\epsilon(s_i,t_j)$ is an independent and identically distributed measurement error with $E[\epsilon(s_i,t_j)] = 0$, variance $Var[\epsilon(s_i,t_j)] = \tau^2$ (the nugget effect) and $Cov[\epsilon(s_i,t_j),\epsilon(s_k,t_l)] = 0$ for all $s_i =!s_k$ or $t_j =!t_l$.

In particular, we define $\mu(s_i,t_j)$, the mean of the stochastic process as

$\mu(s_i,t_j) = \sum_{k=1}^{p} x_k(s_i,t_j)\beta_k,$

where $x_1(s_i,t_j),...,x_p(s_i,t_j)$ are known functions of (s_i,t_j), and $\beta_1,...,\beta_p$ are unknown parameters to be estimated. Equivalently, in matrix notation, we have the spatio-temporal linear model as follows:

$Y = X\beta + Z + \epsilon,$

$Z\, N(0,\sigma^2\Omega_{\phi\rho}),$

$\epsilon\, N(0,\tau^2 I_m).$

Therefore the spatio-temporal process, Y, has normal distribution with mean $E[Y] = X\beta$ and variance $\Sigma = \sigma^2\Omega_{\phi\rho} + \tau^2 I_m$. We assume that Σ is non-singular and X has full rank.

The estimation process was computed via SAEM algorithm initially proposed by Delyon et al. (1999).

Value

The function returns an object of class `Est.StempCens` which is a list given by:

`m.data` Returns a list with all data components given in input.

`m.results` A list given by:

`theta`	final estimation of $\theta = (\beta, \sigma^2, \tau^2, \phi, \rho)$.
`Theta`	estimated parameters in all iterations, $\theta = (\beta, \sigma^2, \tau^2, \phi, \rho)$.
`beta`	estimated β.
`sigma2`	estimated σ^2.
`tau2`	estimated τ^2.
`phi`	estimated ϕ.
`rho`	estimated ρ.
`Eff.range`	estimated effective range.
`PsiInv`	estimated Ψ^-1, where $\Psi = \Sigma/\sigma^2$.
`Cov`	estimated Σ.
`SAEMy`	stochastic approximation of the first moment for the truncated normal distribution.
`SAEMyy`	stochastic approximation of the second moment for the truncated normal distribution.
`Hessian`	Hessian matrix, the negative of the conditional expected second derivative matrix given the observed values.
`Louis`	the observed information matrix using the Louis' method.
`loglik`	log likelihood for SAEM method.
`AIC`	Akaike information criteria.
`BIC`	Bayesian information criteria.
`AICcorr`	corrected AIC by the number of parameters.
`iteration`	number of iterations needed to convergence.

Author(s)

Katherine L. Valeriano, Victor H. Lachos and Larissa A. Matos

Examples

```
## Not run:
# Simulating data
# Initial parameter values
beta <- c(-1,1.50)
phi <- 5;     rho <- 0.45
tau2 <- 0.80; sigma2 <- 1.5
n1 <- 5    # Number of spatial locations
n2 <- 5    # Number of temporal index
set.seed(1000)
x.coord <- round(runif(n1,0,10),9)    # X coordinate
y.coord <- round(runif(n1,0,10),9)    # Y coordinate
coord  <- cbind(x.coord,y.coord)      # Cartesian coordinates without repetitions
coord2 <- cbind(rep(x.coord,each=n2),rep(y.coord,each=n2)) # Cartesian coordinates with repetitions
time <- as.matrix(seq(1,n2))          # Time index without repetitions
```

```
time2 <- as.matrix(rep(time,n1))      # Time index with repetitions
x1 <- rexp(n1*n2,2)
x2 <- rnorm(n1*n2,2,1)
x  <- cbind(x1,x2)
media <- x%*%beta
# Covariance matrix
Ms  <- as.matrix(dist(coord))   # Spatial distances
Mt  <- as.matrix(dist(time))    # Temporal distances
Cov <- CovarianceM(phi,rho,tau2,sigma2,Ms,Mt,1.5,"matern")
# Data
require(mvtnorm)
y <- as.vector(rmvnorm(1,mean=as.vector(media),sigma=Cov))
perc <- 0.20
aa <- sort(y); bb <- aa[1:(perc*n1*n2)]; cutof <- bb[perc*n1*n2]
cc <- matrix(1,(n1*n2),1)*(y<=cutof)
y[cc==1] <- cutof
LI <- y; LI[cc==1] <- -Inf    # Left-censored
LS <- y

# Estimation
est_teste <- EstStempCens(y, x, cc, time2, coord2, LI, LS, init.phi=3.5,
                 init.rho=0.5, init.tau2=0.7,tau2.fixo=FALSE, kappa=1.5,
                 type.S="matern", IMatrix=TRUE, M=20, perc=0.25,
                 MaxIter=300, pc=0.2)
## End(Not run)
```

`PredStempCens` *Prediction in spatio-temporal model with censored/missing responses*

Description

This function performs spatio-temporal prediction in a set of new `S` spatial locations for fixed time points.

Usage

```
PredStempCens(Est.StempCens, locPre, timePre, xPre)
```

Arguments

`Est.StempCens`	an object of class `Est.StempCens` given as output by the `EstStempCens` function.
`locPre`	a matrix of coordinates for which prediction is performed.
`timePre`	the time point vector for which prediction is performed.
`xPre`	a matrix of covariates for which prediction is performed.

Value

The function returns an object of class `Pred.StempCens` which is a list given by:

`predValues`	predicted values.
`VarPred`	predicted covariance matrix.

Author(s)

Katherine L. Valeriano, Victor H. Lachos and Larissa A. Matos

See Also

`EstStempCens`

Examples

```
## Not run:
# Initial parameter values
beta <- c(-1,1.50)
phi  <- 5;    rho <- 0.60
tau2 <- 0.80; sigma2 <- 2
# Simulating data
n1 <- 17   # Number of spatial locations
n2 <- 5    # Number of temporal index
set.seed(12345)
x.co <- round(runif(n1,0,10),9)   # X coordinate
y.co <- round(runif(n1,0,10),9)   # Y coordinate
coord <- cbind(x.co,y.co)         # Cartesian coordinates without repetitions
coord2 <- cbind(rep(x.co,each=n2),rep(y.co,each=n2)) # Cartesian coordinates with repetitions
time <- as.matrix(seq(1,n2))      # Time index without repetitions
time2 <- as.matrix(rep(time,n1))  # Time index with repetitions
x1 <- rexp(n1*n2,2)
x2 <- rnorm(n1*n2,2,1)
x  <- cbind(x1,x2)
media <- x%*%beta
# Covariance matrix
Ms  <- as.matrix(dist(coord))   # Spatial distances
Mt  <- as.matrix(dist(time))    # Temporal distances
Cov <- CovarianceM(phi,rho,tau2,sigma2,Ms,Mt,0.50,"pow.exp")
# Data
require(mvtnorm)
y <- as.vector(rmvnorm(1,mean=as.vector(media),sigma=Cov))
data <- data.frame(coord2,time2,y,x)
names(data) <- c("x.coord","y.coord","time","yObs","x1","x2")
# Splitting the dataset
local.est  <- coord[-c(4,13),]
data.est   <- data[data$x.coord%in%local.est[,1]&data$y.coord%in%local.est[,2],]
data.valid <- data[data$x.coord%in%coord[c(4,13),1]&data$y.coord%in%coord[c(4,13),2],]
# Censored
perc <- 0.10
y <- data.est$yObs
```

```
aa <- sort(y);  bb <- aa[1:(perc*nrow(data.est))]
cutof <- bb[perc*nrow(data.est)]
cc <- matrix(1,nrow(data.est),1)*(y<=cutof)
y[cc==1] <- cutof
data.est <- cbind(data.est[,-c(4,5,6)],y,cc,data.est[,c(5,6)])
names(data.est) <- c("x.coord","y.coord","time","yObs","censored","x1","x2")

# Estimation
y  <- data.est$yObs
x  <- cbind(data.est$x1,data.est$x2)
cc <- data.est$censored
time2  <- matrix(data.est$time)
coord2 <- data.est[,1:2]
LI <- y; LI[cc==1] <- -Inf    # Left-censored
LS <- y
est_teste <- EstStempCens(y, x, cc, time2, coord2, LI, LS, init.phi=3.5,
                 init.rho=0.5, init.tau2=1, kappa=0.5, type.S="pow.exp",
                 IMatrix=FALSE, M=20, perc=0.25, MaxIter=300, pc=0.20)
class(est_teste)

# Prediction
locPre <- data.valid[,1:2]
timePre <- matrix(data.valid$time)
xPre <- cbind(data.valid$x1,data.valid$x2)
pre_teste <- PredStempCens(est_teste, locPre, timePre, xPre)
library(ggplot2)
Model <- rep(c("y Observed","y Predicted"),each=10)
station <- rep(rep(c("Station 1", "Station 2"),each=5),times=2)
xcoord1 <- rep(seq(1:5),4)
ycoord1 <- c(data.valid$yObs,pre_teste$predValues)
data2 <- data.frame(Model,station,xcoord1,ycoord1)
ggplot(data=data2,aes(x=xcoord1,y=ycoord1)) + geom_line(aes(color=Model)) +
facet_wrap(station~.,nrow=2) + labs(x="",y="") + theme(legend.position="bottom")
## End(Not run)
```

Index