Package ‘spate’

October 3, 2023

Title Spatio-Temporal Modeling of Large Data Using a Spectral SPDE Approach

Version 1.7.5

Date 2023-10-03

Author Fabio Sigrist, Hans R. Kuensch, Werner A. Stahel

Maintainer Fabio Sigrist <fabiosigrist@gmail.com>

Depends R (>= 2.10), mvtnorm, truncnorm

SystemRequirements fftw3 (>= 3.1.2)

Description Functionality for spatio-temporal modeling of large data sets is provided. A Gaussian process in space and time is defined through a stochastic partial differential equation (SPDE). The SPDE is solved in the spectral space, and after discretizing in time and space, a linear Gaussian state space model is obtained. When doing inference, the main computational difficulty consists in evaluating the likelihood and in sampling from the full conditional of the spectral coefficients, or equivalently, the latent space-time process. In comparison to the traditional approach of using a spatio-temporal covariance function, the spectral SPDE approach is computationally advantageous. See Sigrist, Kuensch, and Stahel (2015) <doi:10.1111/rssb.12061> for more information on the methodology. This package aims at providing tools for two different modeling approaches. First, the SPDE based spatio-temporal model can be used as a component in a customized hierarchical Bayesian model (HBM). The functions of the package then provide parameterizations of the process part of the model as well as computationally efficient algorithms needed for doing inference with the HBM. Alternatively, the adaptive MCMC algorithm implemented in the package can be used as an algorithm for doing inference without any additional modeling. The MCMC algorithm supports data that follow a Gaussian or a censored distribution with point mass at zero. Covariates can be included in the model through a regression term.

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-10-03 12:00:02 UTC

R topics documented:

`spate-package` *Spatio-temporal modeling of large data with the spectral SPDE approach*

Description

This is an R package for spatio-temporal modeling of large data sets. It provides tools for modeling of Gaussian processes in space and time defined through a stochastic partial differential equation (SPDE). The SPDE is solved in the spectral space, and after discretizing in time and space, a linear Gaussian state space model is obtained. When doing inference, the main computational difficulty consists in evaluating the likelihood and in sampling from the full conditional of the spectral coefficients, or equivalently, the latent space-time process. In comparison to the traditional approach of using a spatio-temporal covariance function, the spectral SPDE approach is computationally advantageous. This package aims at providing tools for two different modeling approaches. First, the SPDE based spatio-temporal model can be used as a component in a customized hierarchical Bayesian model (HBM). The functions of the package then provide parametrizations of the process part of the model as well as computationally efficient algorithms needed for doing inference with the HBM. Alternatively, the adaptive MCMC algorithm implemented in the package can be used as an algorithm for doing inference without any additional modeling. The MCMC algorithm supports data that follow a Gaussian or a censored distribution with point mass at zero. Covariates can be included in the model through a regression term.

Author(s)

Fabio Sigrist, Hans R. Kuensch, Werner A. Stahel

Maintainer: Fabio Sigrist <sigrist@stat.math.ethz.ch>

References

Fabio Sigrist, Hans R. K\"unsch, and Werner A. Stahel, "Stochastic Partial Differential Equation Based Modeling of Large Space-Time Data Sets", Journal of the Royal Statistical Society: Series B, Volume 77, Issue 1, 2015, pages 3-33

Fabio Sigrist, Hans R. Kuensch, Werner A. Stahel, "spate: An R Package for Spatio-Temporal Modeling with a Stochastic Advection-Diffusion Process.", Journal of Statistical Software, Volume 63, Number 14, 2015, pages 1-23, URL http://www.jstatsoft.org/v63/i14/

`cols` *Function that returns the color scale for 'image()'.*

Description

Function that returns the color scale for 'image()'. This function is a simplification of the function 'tim.colors()' from the 'fields' package.

Usage

```
cols()
```

Value

A vector with colors.

Author(s)

Fabio Sigrist

References

Fields Development Team (2006). fields: Tools for Spatial Data. National Center for Atmospheric Research, Boulder, CO. URL http://www.cgd.ucar.edu/Software/Fields.

`ffbs` *Forward Filtering Backward Sampling algorithm.*

Description

Forward Filtering Backward Sampling algorithm for sampling from the joint full conditional of the hidden state of a linear, Gaussian state space model. To be more specific, one samples from $P[\alpha|.]$ where α is specified through

$$y_t = lp_t + Hxi_t + nu_t, \nu_t\ N(0, \Omega)$$

and

$$\alpha_t = G\alpha_{t-1} + \epsilon_t, \epsilon_t\ N(0, \Sigma).$$

Usage

```
ffbs(y, lp, G, Sigma, H, Omega, N = dim(y)[2],T = dim(y)[1],
      NF = dim(G)[1], lglk = FALSE, BwSp = TRUE, filt = FALSE)
```

Arguments

`y`	Observed data in an T x N matrix with columns and rows corresponding to time and space, respectively.
`lp`	Mean (linear predictor) in an T x N matrix with columns and rows corresponding to time and space, respectively.
`G`	Propagator matrix of the latent process α.
`Sigma`	Innovation covariance matrix of the latent process α.
`H`	Observation matrix relating y to α.
`Omega`	Covariance matrix of the observation error ν.
`N`	Number of points in space.
`T`	Number of points in time.
`NF`	Dimension of the latent process α.
`lglk`	Logical; if 'TRUE' the value of the log-likelihood is returned as well.
`BwSp`	Logical; if 'TRUE' a sample from the full conditional of α is returned.
`filt`	Logical; if 'TRUE' the filtered values for α are returned.

Details

In the context of the SPDE, α are the Fourier coefficients.

Value

A list with entries (depending on whether 'lglk', 'BwSp', 'filt' are 'TRUE' or 'FALSE'):

`simAlpha`	A T x N matrix with a sample from the full conditional of latent process α,
`ll`	The evaluated log-likelihood,
`mtt`	A T x N matrix with the mean of the full conditional of latent process α.

Author(s)

Fabio Sigrist

`ffbs.spectral` *Forward Filtering Backward Sampling algorithm in the spectral space of the SPDE.*

Description

Forward Filtering Backward Sampling algorithm for sampling from the joint full conditional of the coefficients α and for evaluation of the log-likelihood.

Usage

```
ffbs.spectral(w=NULL,wFT=NULL,spec=NULL,Gvec=NULL,tau2=NULL,par=NULL,n,T,lglk=FALSE,
              BwSp=TRUE,NF=n*n,indCos=(1:((n*n-4)/2)*2+3),ns=4,nu=1,dt=1)
```

Arguments

`w`	Observed data or latent process w (depending on which data model is used) in an T x n*n matrix with columns and rows (points on a grid stacked into a vector) corresponding to time and space, respectively.
`wFT`	Vector of length T*n*n containing the real Fourier transform of 'w'.
`spec`	Spectrum of the innovations $\hat{\epsilon}$ in a vector of length n*n. If 'spec' is not given, it is constructed based on 'par'.
`Gvec`	The propagator matrix G in vector format obtained from 'get.G.vec'. If 'Gvec' is not given, it is constructed based on 'par'.
`tau2`	Measurement error variance tau2. If 'NULL'; tau2=par[9].
`par`	Vector of parameters for the SPDE in the following order: rho_0, sigma^2, zeta, rho_1, gamma, alpha, mu_x, mu_y, tau^2. If 'spec' and 'Gvec' are given, 'par' will not be used.
`n`	Number of grid points on each axis. n*n is the total number of spatial points.
`T`	Number of points in time.
`lglk`	Logical; if 'TRUE' the value of the log-likelihood is returned as well.
`BwSp`	Logical; if 'TRUE' a sample from the full conditional of α is returned.
`NF`	Number of Fourier functions used.
`indCos`	Vector of integers indicating the position cosine terms in the 1:NF real Fourier functions. The first 'ns' cosine wavenumbers in 'wave' are not included in 'indCos'.
`ns`	Number of real Fourier functions that have only a cosine and no sine term. 'ns' is maximal 4.
`nu`	Smoothness parameter of the Matern covariance function for the innovations. By default this equals 1 corresponding to the Whittle covariance function.
`dt`	Temporal lag between two time points. By default, this equals 1.

Value

A list with entries (depending on whether 'lglk' are 'BwSp' are 'TRUE' or 'FALSE'):

`simAlpha`	A T x n*n matrix with a sample from the full conditional of latent process α,
`ll`	The evaluated log-likelihood,

Author(s)

Fabio Sigrist

get.propagator *Propagator matrix G.*

Description

Function for obtaining the spectral propagator matrix G of the vector autoregressive model for the Fourier coefficients.

Usage

```
get.propagator(wave, indCos, zeta, rho1, gamma, alpha, muX, muY, dt = 1, ns=4)
```

Arguments

wave	Spatial wavenumbers.
indCos	Vector of integers indicating the position of columns in 'wave' of wavenumbers of cosine terms.
zeta	Damping parameter
rho1	Range parameter of the diffusion term
gamma	Parameter that determines the amount of anisotropy in the diffusion term
alpha	Parameter that determines the direction of anisotropy in the diffusion term
muX	X component of the drift vector.
muY	Y component of the drift vector.
dt	Temporal lag between two time points. By default, this equals 1.
ns	Number of real Fourier functions that have only a cosine and no sine term. 'ns' is maximal 4.

Value

Propagator matrix G.

Author(s)

Fabio Sigrist

Examples

```
##For illustration, four grid points on each axis
n <- 4
wave <- wave.numbers(n)
G <- get.propagator(wave=wave$wave,indCos=wave$indCos,zeta=0.5, rho1=0.1, gamma=2,
           alpha=pi/4, muX=0.2, muY=-0.15,dt=1,ns=4)
round(G,digits=2)
## View(round(G,digits=2))
```

```
##An example
n <- 50
spec <- matern.spec(wave=spate.init(n=n,T=1)$wave,n=n,rho0=0.05,sigma2=1,norm=TRUE)
alphat <- sqrt(spec)*rnorm(n*n)
##Propagate initial state
wave <- wave.numbers(n)
G <- get.propagator(wave=wave$wave,indCos=wave$indCos,zeta=0.5, rho1=0.02, gamma=2,
           alpha=pi/4, muX=0.2, muY=0.2,dt=1,ns=4)
alphat1 <- G%*%alphat

opar <- par(no.readonly = TRUE)
par(mfrow=c(1,2))
image(1:n,1:n,matrix(real.fft(alphat,n=n,inv=FALSE),nrow=n),main="Whittle
field",xlab="",ylab="",col=cols())
image(1:n,1:n,matrix(real.fft(alphat1,n=n,inv=FALSE),nrow=n),main="Propagated
field",xlab="",ylab="",col=cols())
par(opar) # Reset par() settings
```

`get.propagator.vec` *Propagator matrix G in vector form.*

Description

Function for obtaining the spectral propagator matrix G of the vector autoregressive model for the Fourier coefficients in vector form.

Usage

```
get.propagator.vec(wave, indCos, zeta, rho1, gamma, alpha, muX, muY, dt = 1,ns=4)
```

Arguments

`wave`	Spatial wavenumbers.
`indCos`	Vector of integers indicating the position of columns in 'wave' of wavenumbers of cosine terms.
`zeta`	Damping parameter
`rho1`	Range parameter of the diffusion term
`gamma`	Parameter that determines the amount of anisotropy in the diffusion term
`alpha`	Parameter that determines the direction of anisotropy in the diffusion term
`muX`	X component of the drift vector.
`muY`	Y component of the drift vector.
`dt`	Temporal lag between two time points. By default, this equals 1.
`ns`	Number of real Fourier functions that have only a cosine and no sine term. 'ns' is maximal 4.

Value

A list with three elements 'G11C', 'G11', and 'G12'. The first element contains a vector of length 'ns' which corresponds to the diagonal propagator of the cosin-only terms. The second element contains the remaining diagonal entries of G, i.e., the diagonal entries of the propagator for the cosine / sine pairs. Note that for each pair, only one value is taken since the diagonal elements for both the cosin and sine terms are equal. The third element is a vector with the off-diagonal terms of the propagator for the cosine / sine pairs.

Author(s)

Fabio Sigrist

Examples

```
##For illustration, four grid points on each axis
n <- 4
wave <- wave.numbers(n)
G <- get.propagator(wave=wave$wave,indCos=wave$indCos,zeta=0.5, rho1=0.1,
           gamma=2,alpha=pi/4, muX=0.2, muY=-0.15,dt=1,ns=4)
diag(G)[1:4]
diag(G[wave$indCos,wave$indCos])
diag(G[wave$indCos,wave$indCos+1])
get.propagator.vec(wave=wave$wave,indCos=wave$indCos,zeta=0.5, rho1=0.1,
gamma=2,alpha=pi/4, muX=0.2, muY=-0.15,dt=1,ns=4)
```

---

get.real.dft.mat *Matrix applying the two-dimensional real Fourier transform.*

Description

Returns the matrix that applies the two-dimensional real Fourier transform.

Usage

```
get.real.dft.mat(wave, indCos, ns = 4, n)
```

Arguments

wave	Matrix of size 2 x NF with spatial wavenumbers. NF is the number of Fourier functions.
indCos	Vector of integers indicating the position of columns in 'wave' of wavenumbers of cosine terms.
ns	Number of real Fourier functions that have only a cosine and no sine term. 'ns' is maximal 4.
n	Number of grid points on each axis. n x n is the total number of spatial points.

Value

A matrix that applies the two-dimensional real Fourier transform.

Author(s)

Fabio Sigrist

Examples

```
##Example nr. 1: sampling from a Matern field
n <- 50
spateFT <- spate.init(n=n,T=1)
spec <- matern.spec(wave=spateFT$wave,n=n,rho0=0.05,sigma2=1,norm=TRUE)
Phi <- get.real.dft.mat(wave=spateFT$wave, indCos=spateFT$indCos, n=n)
sim <- Phi %*% (sqrt(spec)*rnorm(n*n))
image(1:n,1:n,matrix(sim,nrow=n),main="Sample from Matern field",xlab="",ylab="")

##Example nr. 2: image reconstruction
n <- 50##Number of points on each axis
##Low-dimensional: only 41 Fourier functions
spateFT <- spate.init(n=n,T=17,NF=45)
Phi.LD <- get.real.dft.mat(wave=spateFT$wave, indCos=spateFT$indCos, ns=spateFT$ns, n=n)
##Mid-dimensional: 545 (of potentially 2500) Fourier functions
spateFT <- spate.init(n=n,T=17,NF=101)
Phi.MD <- get.real.dft.mat(wave=spateFT$wave, indCos=spateFT$indCos, ns=spateFT$ns, n=n)
##High-dimensional: all 2500 Fourier functions
spateFT <- spate.init(n=n,T=17,NF=2500)
Phi.HD <- get.real.dft.mat(wave=spateFT$wave, indCos=spateFT$indCos, ns=spateFT$ns, n=n)
##Define image
image <- rep(0,n*n)
for(i in 1:n){
  for(j in 1:n){
   image[(i-1)*n+j] <- cos(5*(i-n/2)/n*pi)*sin(5*(j)/n*pi)*(1-abs(i/n-1/2)-abs(j/n-1/2))
  }
}
opar <- par(no.readonly = TRUE)
par(mfrow=c(2,2),mar=c(2,3,2,1))
image(1:n, 1:n, matrix(image, nrow = n),col = cols(),xlab="",ylab="",main="Original image")
##Aply inverse Fourier transform, dimension reduction, and Fourier transform
spec.LD <- t(Phi.LD) %*% image
image.LD <- Phi.LD %*% spec.LD
spec.MD <- t(Phi.MD) %*% image
image.MD <- Phi.MD %*% spec.MD
spec.HD <- t(Phi.HD) %*% image
image.HD <- Phi.HD %*% spec.HD
image(1:n, 1:n, matrix(image.LD, nrow = n),col = cols(),
      xlab="",ylab="",main="45 of 2500 Fourier terms")
image(1:n, 1:n, matrix(image.MD, nrow = n),col = cols(),
      xlab="",ylab="",main="101 of 2500 Fourier terms")
image(1:n, 1:n, matrix(image.HD, nrow = n),col = cols(),
      xlab="",ylab="",main="All 2500 Fourier terms")
par(opar) # Reset par() settings
```

`index.complex.to.real.dft` *Auxilary function for the real Fourier transform.*

Description

Auxilary function for the conversion between the complex FFT and the real Fourier transform.

Usage

```
index.complex.to.real.dft(n)
```

Arguments

`n`	Number of points on each axis. n x n is the total number of spatial points.

Value

A a list of indices used for the conversion between the complex FFT and the real Fourier transform.

Author(s)

Fabio Sigrist

`innov.spec` *Spectrum of the innovation term epsilon.*

Description

Spectrum of the innovation term epsilon.

Usage

```
innov.spec(wave,n,ns=4,rho0,sigma2,zeta,rho1,alpha,gamma,nu=1,dt=1,norm=TRUE)
```

Arguments

`wave`	Spatial wavenumbers.
`n`	Number of grid points on each axis. n x n is the total number of spatial points.
`ns`	Number of real Fourier functions that have only a cosine and no sine term. 'ns' is maximal 4.
`rho0`	Range of the Matern covariance funtion for the innovation term epsilon
`sigma2`	Marginal variance of the Matern covariance funtion for the innovation term epsilon

zeta	Damping parameter
rho1	Range parameter of the diffusion term
alpha	Parameter that determines the direction of anisotropy in the diffusion term
gamma	Parameter that determines the amount of anisotropy in the diffusion term
nu	Smoothness parameter of the Matern covariance function for the innovations. By default, this equals 1 corresponding to the Whittle covariance function.
dt	Temporal lag between two time points. By default, this equals 1.
norm	logical; if 'TRUE' the spectrum is multiplied by n*n so that after applying the real Fourier transform 'real.FFT' one has the correct normalization.

Value

Vector with the spectrum of the integrated innovation term epsilon hat.

Author(s)

Fabio Sigrist

Examples

```
n <- 100
spec <- innov.spec(wave=spate.init(n=n,T=1)$wave,n=n,rho0=0.05,sigma2=0.5,zeta=0.5,
                   rho1=0.05,alpha=pi/4,gamma=2,norm=TRUE)
sim <- real.fft(sqrt(spec)*rnorm(n*n),n=n,inv=FALSE)
image(1:n,1:n,matrix(sim,nrow=n),main="Sample from the integrated
stochastic innovation",xlab="",ylab="",col=cols())
```

`lin.pred` *Linear predictor.*

Description

Calculates the linear predictor.

Usage

```
lin.pred(x, beta)
```

Arguments

x	Covariates in an array of dimensions p x T X N, where p denotes the number of covariates, T the number of time points, and N the number of spatial points.
beta	Coefficients of covariates in a vector of length p.

Value

Matrix of dimension T x N with linear predictors.

Author(s)

Fabio Sigrist

---

loglike *Log-likelihood of the hyperparameters.*

Description

Evaluates the log-likelihood of the hyperparameters given the data (Gaussian case) or given the latent variable w (in the Tobit case).

Usage

```
loglike(par=NULL,w=NULL,wFT=NULL,x=NULL,spec=NULL,Gvec=NULL,tau2=NULL,n,T,
        NF=n*n,indCos=(1:((n*n-4)/2)*2+3),ns=4,nu=1,dt=1,logScale=FALSE,
        logInd=c(1,2,3,4,5,9),negative=FALSE)
```

Arguments

par	Vector of parameters for the SPDE in the following order: rho_0, sigma^2, zeta, rho_1, gamma, alpha, mu_x, mu_y, tau^2, regression coefficients beta. rho_0 and sigma^2 are the range and marginal variance of the Whittle covariance function for the innovation term epsilon. zeta is the damping parameter. rho_1, gamma, and alpha parametrize the diffusion matrix with rho_1 being a range parameter, gamma and alpha determining the amount and the direction, respectively, of anisotropy. mu_x and mu_y are the two components of the drift vector. tau^2 denotes the variance of nugget effect or measurment error. Subsequently in par are the regression coefficients beta, if there are covariates.
w	Matrix of size T x N, where T and N denote the number of points in time and space. In the case of a Gaussian data model, w contains the observed values, with the Tobit model, w denotes the latent normal variable.
wFT	A vector with the (discrete) Fourier transform of the observed or latent w, depending on which data model is used. Note that, in contrast to w, this needs to be in stacked vector format. Use 'TSmat.to.vect'.
x	Covariates in an array of dimensions p x T X N, where p denotes the number of covariates, T the number of time points, and n the number of spatial points.
spec	A vector containing the spectrum of the innovation term epsilon. If 'spec' is not given, it is constructed based on 'par'.
Gvec	The propagator matrix G in vector format obtained from 'get.G.vec'. If 'Gvec' is not given, it is constructed based on 'par'.
tau2	Measurement error variance tau2. If 'NULL'; tau2=par[9].
n	Number of grid points on each axis. n x n is the total number of spatial points.
T	Number of points in time.

NF	Number of Fourier functions.
indCos	Vector of integers indicating the position of wavenumbers of cosine-only terms.
ns	Number of real Fourier functions that have only a cosine and no sine term. 'ns' is maximal 4.
nu	Smoothness parameter of the Matern covariance function for the innovations. By default this equals 1 corresponding to the Whittle covariance function.
dt	Temporal lag between two time points. By default, this equals 1.
logScale	logical; if 'TRUE' the parameters specified in 'logInd' are on the logarithmic scale. This is used for constraining parameters to be positive.
logInd	Vector of integers indicating which parameters are on the log-scale.
negative	logical; if 'TRUE' the negative log-likelihood is returned otherwise the positive log-likelihood is returned.

Value

Value of the log-likelihood evaluated at 'par'.

Author(s)

Fabio Sigrist

Examples

```
n <- 20
T <- 20
##Specify hyper-parameters
par <- c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01)
##Simulate data
spateSim <- spate.sim(par=par,n=n,T=T,seed=4)
w <- spateSim$w

##Initial values for optim. This takes a couple of seconds.
parI <- c(rho0=0.2,sigma2=0.1,zeta=0.25,rho1=0.01,gamma=1,alpha=0.3,muX=0,muY=0,tau2=0.005)
logInd=c(1,2,3,4,5,9)
##Transform to log-scale
parI[logInd] <- log(parI[logInd])

##Fourier transform needs to be done only once
wFT <- real.fft.TS(w,n=n,T=T)
##ML estimation using optim, takes a couple of seconds
##Load the precomputed object a line below to save time
##spateMLE <- optim(par=parI,loglike,control=list(trace=TRUE,maxit=1000),wFT=wFT,method="L-BFGS-B",
##     lower=c(-10,-10,-10,-10,-10,0,-0.5,-0.5,-10),
##     upper=c(10,10,10,10,10,pi/2,0.5,0.5,10),negative=TRUE,
##     logScale=TRUE,hessian=TRUE,n=n,T=T)
data("spateMLE")

mle <- spateMLE$par
mle[logInd] <- exp(mle[logInd])
```

```
sd=sqrt(diag(solve(spateMLE$hessian)))

MleConfInt <- data.frame(array(0,c(4,9)))
colnames(MleConfInt) <- names(par)
rownames(MleConfInt) <- c("True","Estimate","Lower","Upper")
MleConfInt[1,] <- par
MleConfInt[2,] <- mle
MleConfInt[3,] <- spateMLE$par-2*sd
MleConfInt[4,] <- spateMLE$par+2*sd
MleConfInt[c(3,4),logInd] <- exp(MleConfInt[c(3,4),logInd])
cat("\n")
round(MleConfInt,digits=4)
```

`map.obs.to.grid` *Maps non-gridded data to a grid.*

Description

Maps non-gridded data to a grid based on the coordinates supplied. Cells with no data are NA. For cells with more than one data point, the average is taken.

Usage

```
map.obs.to.grid(n,y.non.grid,coord,lengthx=NULL,lengthy=NULL)
```

Arguments

`y.non.grid`	Observed data in an T x N matrix with columns and rows corresponding to time and space, respectively. The coordinates of each observation point need to be specified in 'coord'.
`coord`	Matrix of dimension N x 2 with coordinates of the N observation points. Based on to these coordinates, each observation location is then mapped to a grid cell.
`lengthx`	Use together with 'coord' to specify the length of the x-axis. This is usefull if the observations lie in a rectangular area instead of a square. The length needs to be at least as large as the largest x-distance in 'coord.
`lengthy`	Use together with 'coord' to specify the length of the y-axis. This is usefull if the observations lie in a rectangular area instead of a square. The length needs to be at least as large as the largest y-distance in 'coord.
`n`	Number of point per axis of the square into which the points are mapped. In total, the process is modeled on a grid of size n*n.

Value

The function returns data in an T x n^2 matrix with columns and rows corresponding to time and space, respectively. Cells with no data are NA. For cells with more than one data point, the average is taken.

Author(s)

Fabio Sigrist

Examples

```
## See code of 'spate.mcmc'.
```

`matern.spec` *Spectrum of the Matern covariance function.*

Description

Spectrum of the Matern covariance function. Note that the spectrum is renormalized, by dividing with the sum over all frequencies so that they sum to one, so that σ^2 is the marginal variance no matter how many wavenumbers are included.

Usage

```
matern.spec(wave, n, ns=4, rho0, sigma2, nu = 1, norm = TRUE)
```

Arguments

`wave`	Spatial wavenumbers.
`n`	Number of grid points on each axis. n x n is the total number of spatial points.
`ns`	Integer indicating the number of cosine-only terms. Maximally this is 4.
`rho0`	Range parameter.
`sigma2`	Marginal variance parameter.
`nu`	Smoothness parameter of the Matern covariance function. By default this equals 1 corresponding to the Whittle covariance function.
`norm`	logical; if 'TRUE' the spectrum is multiplied by n*n so that after applying the real Fourier transform 'real.FFT' one has the correct normalization.

Details

The Matern covariance function is of the form

$$\sigma^2 2^(1-\nu)\Gamma(\nu)^{-1}(d/\rho_0)^\nu K_\nu(d/\rho_0)$$

with 'd' being the Euclidean distance between two points and K_nu(.) a modified Bessel function. Its spectrum is given by

$$2^{\nu-1}\nu((1/\rho_0)^(2\nu))(\pi*((1/\rho_0)^2+w)^(\nu+1))^{-1}$$

where 'w' is a spatial wavenumber.

Value

Vector with the spectrum of the Matern covariance function.

Author(s)

Fabio Sigrist

Examples

```
n <- 100
spec <- matern.spec(wave=spate.init(n=n,T=1)$wave,n=n,rho0=0.05,sigma2=1,norm=TRUE)
sim <- real.fft(sqrt(spec)*rnorm(n*n),n=n,inv=FALSE)
image(1:n,1:n,matrix(sim,nrow=n),main="Sample from a Gaussian process
with Matern covariance function",xlab="",ylab="",col=cols())
```

`mcmc.summary` *Summary function for MCMC output.*

Description

Auxilary function for summarizing MCMC output and illustrating the posterior distributions.

Usage

```
mcmc.summary(data, probs = c(0.025, 0.5, 0.975), mean = FALSE)
```

Arguments

`data`	Matrix of size p x Nmc where p denotes the number of parameters and Nmc the number of MCMC samples.
`probs`	Vector of quantiles that should be computed for each parameter.
`mean`	logical; if 'TRUE' the mean of the posterior distributions is computed as well.

Value

Matrix with quantiles and the mean of the posterior distributions.

Author(s)

Fabio Sigrist

Examples

```
data("spateMCMC")
mcmc.summary(spateMCMC$Post,mean=TRUE)
```

Palpha — *Prior for direction of anisotropy in diffusion parameter alpha.*

Description

Default prior for direction of anisotropy in diffusion parameter alpha. A uniform prior on [0,pi/4] is used.

Usage

```
Palpha(alpha, log = FALSE)
```

Arguments

`alpha`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'alpha'.

Author(s)

Fabio Sigrist

Pgamma — *Prior for amount of anisotropy in diffusion parameter gamma.*

Description

Default prior for amount of anisotropy in diffusion parameter gamma. A uniform prior on log(gamma) over the interval [1/100,100] is used.

Usage

```
Pgamma(gamma, log = FALSE)
```

Arguments

`gamma`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'gamma'.

Author(s)

Fabio Sigrist

`Plambda` *Prior for transformation parameter of the Tobit model.*

Description

Default prior for transformation parameter of the Tobit model. A locally constant, improper prior on the positive real line is used.

Usage

```
Plambda(lambda, log = FALSE)
```

Arguments

`lambda`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'lambda'.

Author(s)

Fabio Sigrist

`plot.spateMCMC` *Plot fitted spateMCMC objects.*

Description

Plots trace plots, pair plots, the posterior of the hyperparameters and the posterior of the latent spatio-temporal process.

Usage

```
## S3 method for class 'spateMCMC'
plot(x,..., trace = TRUE, hist = TRUE,
              medianHist=TRUE, pairs = FALSE,ask = TRUE, ToFile = FALSE,
              path = NULL,file = NULL,true=NULL,BurnInAdaptive=NULL,
              postProcess = FALSE)
```

Arguments

x	A 'spateMCMC' object obtained from 'spate.mcmc'.
...	Arguments to be passed to 'spate.plot' in case 'postProcess=TRUE' is selected.
trace	logical; if 'TRUE' trace plots are made
hist	logical; if 'TRUE' histograms of the posterior distributions for the hyper-parameters are plotted
medianHist	logical; if 'TRUE' medians are added to the histograms.
pairs	logical; if 'TRUE' trace plots are made
ask	logical; if 'TRUE' (and the R session is interactive) the user is asked for input, before a new figure is drawn.
ToFile	logical; if 'TRUE' the plots are save to a file.
path	The path.
file	The file name.
true	The true value of the parameters (for simulation only).
BurnInAdaptive	The number of samples used as burn-in before starting the adaptive estimation of Metropolis-Hastings proposal covariance for the hyper-parameters.
postProcess	logical; if 'TRUE' the posterior of the spatio-temporal process xi is plotted as well.

Value

Plots illustrating a fitted model saved in a 'spateMCMC' object.

Author(s)

Fabio Sigrist

Examples

```
data("spateMCMC")
plot(spateMCMC,medianHist=TRUE,pairs=TRUE)
```

---

plot.spateSim *Plotting function for 'spateSim' objects.*

Description

This is the plotting function for 'spateSim' objects. It calles the function 'spate.plot()'.

Usage

```
## S3 method for class 'spateSim'
plot(x,..., plotXi =TRUE,plotW = FALSE)
```

Arguments

`x`	'spateSim' object obtained from 'spate.sim'.
`...`	Arguments to be passed to 'spate.plot'
`plotXi`	Logical; if 'TRUE' the latent process 'xi' is plotted.
`plotW`	Logical; if 'TRUE' the observed process 'w' is plotted.

Value

Plots illustrating the simulated space-time field.

Author(s)

Fabio Sigrist

Examples

```
spateSim <-spate.sim(par=c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,
                     alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01),n=50,T=9)
plot(spateSim)
```

`Pmux` *Prior for y-component of drift.*

Description

Default prior for x-component of drift vector mu. A uniform prior on the interval [-0.5,0.5] is used.

Usage

```
Pmux(mux, log = FALSE)
```

Arguments

`mux`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'mux'.

Author(s)

Fabio Sigrist

`Pmuy` *Prior for y-component of drift.*

Description

Default prior for y-component of drift vector mu. A uniform prior on the interval [-0.5,0.5] is used.

Usage

```
Pmuy(muy, log = FALSE)
```

Arguments

`muy`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'muy'.

Author(s)

Fabio Sigrist

`post.dist.hist` *Histogram of posterior distributions.*

Description

Histogram of posterior distributions.

Usage

```
post.dist.hist(data, true=NULL, breaks = 20, mean = FALSE, median = TRUE)
```

Arguments

`data`	Matrix of size p x Nmc where p denotes the number of parameters and Nmc the number of MCMC samples.
`true`	The true value of the parameters (for simulation only).
`breaks`	Parameter for 'hist()' function.
`mean`	logical; if 'TRUE' the mean is added to the histogram.
`median`	logical; if 'TRUE' the median is added to the histogram.

Value

Histograms illustrating posterior distributions.

Author(s)

Fabio Sigrist

`Prho0` *Prior for range parameter rho0 of innovation epsilon.*

Description

Default prior for range parameter rho0 of stochastic source-sink term epsilon. A uniform prior on [0,100] is used.

Usage

```
Prho0(rho0, log = FALSE)
```

Arguments

`rho0`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'rho0'.

Author(s)

Fabio Sigrist

`Prho1` *Prior for range parameter rho1 of diffusion.*

Description

Default prior for range parameter rho1 of diffusive term. A uniform prior on [0,100] is used.

Usage

```
Prho1(rho1, log = FALSE)
```

Arguments

`rho1`	A quantile.
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at ’rho1’.

Author(s)

Fabio Sigrist

`print.spateMCMC` *Print function for spateMCMC objects.*

Description

Print function for spateMCMC objects.

Usage

```
## S3 method for class 'spateMCMC'
print(x,...)
```

Arguments

`x`	A ’spateMCMC’ object obtained from ’spate.mcmc’.
`...`	not used.

Author(s)

Fabio Sigrist

`print.spateSim` *Print function for ’spateSim’ objects.*

Description

Print function for ’spateSim’ objects.

Usage

```
## S3 method for class 'spateSim'
print(x,...)
```

Arguments

x	'spateSim' object obtained from 'spate.sim'.
...	Arguments to be passed to 'spate.plot'

Author(s)

Fabio Sigrist

Examples

```
spateSim <-spate.sim(par=c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,
                     alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01),n=50,T=9)
spateSim
```

---

propagate.spectral *Function that propagates a state (spectral coefficients).*

Description

Function that propagates the vector 'alphat'. This is equivalent to multiplying 'alphat' with the propagator matrix G. It is a lot faster though, due to the block-diagonal structure of G. This is a wrapper function of a C function.

Usage

```
propagate.spectral(alphat,spateFT=NULL,n=NULL,Gvec=NULL,par=NULL)
```

Arguments

alphat	A vector of spectral coefficients.
spateFT	A 'spateFT' obtained from 'spate.init'. Either this or 'n' needs to be given.
n	Number of points on each axis. n x n is the total number of spatial points. Either this or 'spateFT' needs to be given.
Gvec	The propagator matrix G in vector format obtained from 'get.propagator.vec'. If 'Gvec' is not given, it is constructed based on 'par'.
par	Parameters for the SPDE in the following order: rho_0, sigma^2, zeta, rho_1, gamma, alpha, mu_x, mu_y, tau^2. If 'Gvec' is not given, 'par' needs to be given.

Value

A vector of propagated coefficients G*alphat.

Author(s)

Fabio Sigrist

Examples

```
n <- 50
spec <- matern.spec(wave=spate.init(n=n,T=1)$wave,n=n,rho0=0.05,sigma2=1,norm=TRUE)
alphat <- sqrt(spec)*rnorm(n*n)
##Propagate initial state
wave <- wave.numbers(n)
Gvec <- get.propagator.vec(wave=wave$wave,indCos=wave$indCos,zeta=0.1,rho1=0.02,gamma=2,
                  alpha=pi/4,muX=0.2,muY=0.2,dt=1,ns=4)
alphat1 <- propagate.spectral(alphat,n=n,Gvec=Gvec)

opar <- par(no.readonly = TRUE)
par(mfrow=c(1,2))
image(1:n,1:n,matrix(real.fft(alphat,n=n,inv=FALSE),nrow=n),main="Whittle
field",xlab="",ylab="",col=cols())
image(1:n,1:n,matrix(real.fft(alphat1,n=n,inv=FALSE),nrow=n),main="Propagated
field",xlab="",ylab="",col=cols())
par(opar) # Reset par() settings
```

`Psigma2` *Prior for for variance parameter sigma2 of innovation epsilon. hyper-parameter.*

Description

Default prior for marginal variance parameter sigma2 (=sigma^2) of the stochastic source-sink term epsilon. A uniform, improper prior on sigma (P[sigma] propto 1 or P[sigma2] propto 1/tau) is used.

Usage

```
Psigma2(sigma2, log = FALSE)
```

Arguments

`sigma2`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'sigma2'.

Author(s)

Fabio Sigrist

`Ptau2` *Prior for nugget effect parameter tau2.*

Description

Default prior for measurment error or small scale variation tau2 (nugget effect). A uniform, improper prior on tau (P[tau] propto 1 or P[tau2] propto 1/tau) is used.

Usage

```
Ptau2(tau2, log = FALSE)
```

Arguments

`tau2`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at tau2.

Author(s)

Fabio Sigrist

`Pzeta` *Prior for damping parameter zeta.*

Description

Default prior for damping parameter zeta. A uniform, improper prior on the positive real line (P[zeta] propto 1) is used.

Usage

```
Pzeta(zeta, log = FALSE)
```

Arguments

`zeta`	A quantile
`log`	Indicates whether the logarithm should be calculated or not

Value

Value of (log) prior at 'zeta'.

Author(s)

Fabio Sigrist

---

real.fft *Fast calculation of the two-dimensional real Fourier transform.*

Description

Fast calculation of the real Fourier transform. This is a wrapper function for a C function which uses the complex FFT function from the 'fftw3' library.

Usage

```
real.fft(w,n,inv=TRUE,indFFT=NULL)
```

Arguments

w	A spatial field in a stacked vector of length N=n^2.
n	Number of grid points on each axis. n x n is the total number of spatial points.
inv	Indicates whether the inverse Fourier transform should be calculated or not.
indFFT	A list of containing vectors of natural numbers representing indices used to transform between the real and the complex Fourier transform.

Value

A vector of length n*n containing the real (inverse) Fourier transformation of 'w'.

Author(s)

Fabio Sigrist

Examples

```
n <- 100
spec <- matern.spec(wave=spate.init(n=n,T=1)$wave,n=n,rho0=0.05,sigma2=1,norm=TRUE)
sim <- real.fft(sqrt(spec)*rnorm(n*n),n=n,inv=FALSE)
image(1:n,1:n,matrix(sim,nrow=n),main="Sample from Matern field",xlab="",ylab="")
```

real.fft.TS — *Fast calculation of the two-dimensional real Fourier transform of a space-time field. For each time point, the spatial field is transformed.*

Description

This function calculates the two-dimensional real Fourier transform of a space-time field. This is a wrapper function for a C function which uses the complex FFT function from the 'fftw3' library. In contrast to using T times the function 'real.FFT', R needs to communicate with C only once and not T times which saves computational time.

Usage

```
real.fft.TS(w,n,T,inv=TRUE,indFFT=NULL)
```

Arguments

`w`	Spatio-temporal field in a stacked vector of length T x N. Stacking is done first over space and then time. E.g., the first N=n^2 entries contain the spatial field at time t=1. Note that the spatial field itself is stacked as well, i.e., each spatial field is in a vector of length N=n^2.
`n`	Number of grid points on each axis. n x n is the total number of spatial points.
`T`	Number of time points.
`inv`	Indicates whether the inverse Fourier transform should be calculated or not.
`indFFT`	A list of containing vectors of natural numbers representing indices used to transform between the real and the complex Fourier transform.

Value

A vector of length T x N containing the (inverse) Fourier transform of 'w'.

Author(s)

Fabio Sigrist

Examples

```
n <- 100
T <- 4
spec <- matern.spec(wave=spate.init(n=n,T=1)$wave,n=n,rho0=0.05,sigma2=1,norm=TRUE)
specsim <- matrix(0,nrow=T,ncol=n*n)
for(t in 1:T) specsim[t,] <- rnorm(n*n)*sqrt(spec)
maternsim <- vect.to.TSmat(real.fft.TS(TSmat.to.vect(specsim),n=n,T=T,inv=FALSE),T=T)
opar <- par(no.readonly = TRUE)
par(mfrow=c(2,2))
for(t in 1:T) image(1:n,1:n,matrix(maternsim[t,],nrow=n),
                    main="Sample from Matern field",xlab="",ylab="")
par(opar) # Reset par() settings
```

`sample.four.coef` *Sample from the full conditional of the Fourier coefficients.*

Description

Sample from the full conditional of the Fourier coefficients.

Usage

```
sample.four.coef(w=NULL,wFT=NULL,spec=NULL,Gvec=NULL,tau2=NULL,par=NULL,n,T,
                 NF=n*n,indCos=(1:((n*n-4)/2)*2+3),ns=4,nu=1,dt=1)
```

Arguments

`w`	Observed data or latent process w (depending on which data model is used) in an T x n*n matrix with columns and rows (points on a grid stacked into a vector) corresponding to time and space, respectively.
`wFT`	Vector of length T*n*n containing the real Fourier transform of 'w'.
`spec`	Spectrum of the innovations $\hat{\epsilon}$ in a vector of length n*n. If 'spec' is not given, it is constructed based on 'par'.
`Gvec`	The propagator matrix G in vector format obtained from 'get.G.vec'. If 'Gvec' is not given, it is constructed based on 'par'.
`tau2`	Measurement error variance tau2. If 'NULL'; tau2=par[9].
`par`	Vector of parameters for the SPDE in the following order: rho_0, sigma^2, zeta, rho_1, gamma, alpha, mu_x, mu_y, tau^2. If 'spec' and 'Gvec' are given, 'par' will not be used.
`n`	Number of grid points on each axis. n*n is the total number of spatial points.
`T`	Number of points in time.
`NF`	Number of Fourier functions used.
`indCos`	Vector of integers indicating the position cosine terms in the 1:NF real Fourier functions. The first 'ns' cosine wavenumbers in 'wave' are not included in 'indCos'.
`ns`	Number of real Fourier functions that have only a cosine and no sine term. 'ns' is maximal 4.
`nu`	Smoothness parameter of the Matern covariance function for the innovations. By default this equals 1 corresponding to the Whittle covariance function.
`dt`	Temporal lag between two time points. By default, this equals 1.

Value

A T x n*n matrix with a sample from the full conditional of latent process α.

Author(s)

Fabio Sigrist

Examples

```
##Specifications for simulated example
n <- 50
T <- 4
par <- c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01)
spateSim <- spate.sim(par=par,n=n,T=T,seed=4)
w <- spateSim$w
##Sample from full conditional
Nmc <- 50
alphaS <- array(0,c(T,n*n,Nmc))
wFT <- real.fft.TS(w,n=n,T=T)
for(i in 1:Nmc){
  alphaS[,,i] <- sample.four.coef(wFT=wFT,par=par,n=n,T=T,NF=n*n)
}
##Mean from full conditional
alphaMean <- apply(alphaS,c(1,2),mean)
xiMean <- real.fft.TS(alphaMean,n=n,T=T,inv=FALSE)

opar <- par(no.readonly = TRUE)
par(mfrow=c(2,4),mar=c(1,1,1,1))
for(t in 1:4) image(1:n,1:n,matrix(w[t,],nrow=n),xlab="",ylab="",col=cols(),
                    main=paste("w(",t,")",sep=""),xaxt='n',yaxt='n')
for(t in 1:4) image(1:n,1:n,matrix(xiMean[t,],nrow=n),xlab="",ylab="",col=cols(),
                    main=paste("xiPost(",t,")",sep=""),xaxt='n',yaxt='n')
par(opar) # Reset par() settings
```

---

spate.init — *Constructor for 'spateFT' object which are used for the two-dimensional Fourier transform.*

Description

Auxilary function for constructing 'spateFT' objects which are used for the two-dimensional Fourier transform.

Usage

```
spate.init(n,T,NF=n*n)
```

Arguments

n	Number of points on each axis. n x n is the total number of spatial points.
T	Number of temporal points.
NF	This integer specifies the number of Fourier functions. If NF<n*n, dimension reduction is obtained. In this case, Fourier functions with wavenumbers closest to the origin (0,0) are first included. If a given 'NF' implies a basis with anisotropy, i.e., higher frequencies in one direction than in another, this is automatically corrected by using the next higher integer NF' such that the basis has the same resolution in all directions.

Value

A 'spateFT' object. This is a list with

`wave`	a matrix containing the wavenumbers
`indCos`	a vector indicating the position of the cosine terms (excluding the frist 'ns')
`ns`	an integer indicating the number of cosine-only terms
`indFFT`	a list of indices used for the conversion between the complex FFT and the real Fourier transform.
`n`	number of points on each axis
`T`	number of points in time

Author(s)

Fabio Sigrist

`spate.mcmc` *MCMC algorithm for fitting the model.*

Description

MCMC algorithm for fitting the model.

Usage

```
spate.mcmc(y,coord=NULL,lengthx=NULL,lengthy=NULL,Sind=NULL,n=NULL,
          IncidenceMat=FALSE,x=NULL,SV=c(rho0=0.2,sigma2=0.1,
          zeta=0.25,rho1=0.2,gamma=1,alpha=0.3,muX=0,muY=0,tau2=0.005),
          betaSV=rep(0,dim(x)[1]),RWCov=NULL,parh=NULL,tPred=NULL,
          sPred=NULL,P.rho0=Prho0,P.sigma2=Psigma2,P.zeta=Pzeta,P.rho1=Prho1,
          P.gamma=Pgamma,P.alpha=Palpha,P.mux=Pmux,P.muy=Pmuy,P.tau2=Ptau2,
          lambdaSV=1,sdlambda=0.01,P.lambda=Plambda,DataModel="Normal",
          DimRed=FALSE,NFour=NULL,indEst=1:9,Nmc=10000,BurnIn =1000,
          path=NULL,file=NULL,SaveToFile=FALSE,PlotToFile=FALSE,
          FixEffMetrop=TRUE,saveProcess=FALSE,Nsave=200,seed=NULL,
          Padding=FALSE,adaptive=TRUE,NCovEst=500,BurnInCovEst=500,
          MultCov=0.5,printRWCov=FALSE,MultStdDevLambda=0.75,
          Separable=FALSE,Drift=!Separable,Diffusion=!Separable,
          logInd=c(1,2,3,4,5,9),nu=1,plotTrace=TRUE,
          plotHist=FALSE,plotPairs=FALSE,trueVal=NULL,
          plotObsLocations=FALSE,trace=TRUE,monitorProcess=FALSE,
          tProcess=NULL,sProcess=NULL)
```

Arguments

- `y` Observed data in an T x N matrix with columns and rows corresponding to time and space (observations on a grid stacked into a vector), respectively. By default, at each time point, the observations are assumed to lie on a square grid with each axis scaled so that it has unit length.
- `coord` If specified, this needs to be a matrix of dimension N x 2 with coordinates of the N observation points. Observations in 'y' can either be on a square grid or not. If not, the coordinates of each observation point need to be specified in 'coord'. According to these coordinates, each observation location is then mapped to a grid cell. If 'coord' is not specified, the observations in 'y' are assumed to lie on a square grid with each axis scaled so that it has unit length.
- `lengthx` Use together with 'coord' to specify the length of the x-axis. This is usefull if the observations lie in a rectangular area instead of a square. The length needs to be at least as large as the largest x-distance in 'coord.
- `lengthy` Use together with 'coord' to specify the length of the y-axis. This is usefull if the observations lie in a rectangular area instead of a square. The length needs to be at least as large as the largest y-distance in 'coord.
- `Sind` Vector of indices of grid cells where observations are made, in case, the observation are not made at every grid cell. Alternatively, the coordinates of the observation locations can be specfied in 'coord'.
- `n` Number of point per axis of the square into which the points are mapped. In total, the process is modeled on a grid of size n*n.
- `IncidenceMat` Logical; if 'TRUE' an incidence matrix relating the latent process to observation locations is used. This is only recommended to use when the observations are relatively low-dimensional and when the latent process is modeled in a reduced dimensional space as well.
- `x` Covariates in an array of dimensions p x T X N, where p denotes the number of covariates, T the number of time points, and N the number of spatial points.
- `SV` Starting values for parameters. Parameters for the SPDE in the following order: rho_0, sigma^2, zeta, rho_1, gamma, alpha, mu_x, mu_y, tau^2. rho_0 and sigma^2 are the range and marginal variance of the Matern covariance funtion for the innovation term epsilon. zeta is the damping parameter. rho_1, gamma, and alpha parametrize the diffusion matrix with rho_1 being a range parameter, gamma and alpha determining the amount and the direction, respectively, of anisotropy. mu_x and mu_y are the two components of the drift vector. tau^2 denotes the nugget effect or measurment error.
- `betaSV` Starting values for regression coefficients.
- `RWCov` Covariance matrix of the proposal distribution used in the random walk Metropolis-Hastings step for the hyper-parameters.
- `parh` Only used in prediction mode. If 'parh' is not 'NULL', this indicates that 'spate.mcmc' is used for making predictions at locations (tPred,sPred) instead of applying the traditional MCMC algorithm. In case 'parh' is not 'NULL', it is a Npar x Nsim matrix containing Nsim samples from the posterior of the Npar parameters. This argument is used by the wrapper function 'spate.predict'.

tPred	Time points where predictions are made.This needs to be a vector if predictions are made at multiple times. For instance, if T is the number of time points in the data 'y', then tPred=c(T+1, T+2) means that predictions are made at time 'T+1' and 'T+2'. This argument is used by the wrapper function 'spate.predict'.
sPred	Vector of indices of grid cells (positions of locations in the stacked spatial vector) where predictions are made. This argument is used by the wrapper function 'spate.predict'.
P.rho0	Function specifying the prior for rho0.
P.sigma2	Function specifying the prior for sigma2.
P.zeta	Function specifying the prior for zeta.
P.rho1	Function specifying the prior for rho1.
P.gamma	Function specifying the prior for gamma.
P.alpha	Function specifying the prior for alpha.
P.mux	Function specifying the prior for mux.
P.muy	Function specifying the prior for muy.
P.tau2	Function specifying the prior for tau2.
lambdaSV	Starting value for transformation parameter lambda in the Tobit model.
sdlambda	Standard deviation of the proposal distribution used in the random walk Metropolis-Hastings step for lambda.
P.lambda	Function specifying the prior for lambda.
DataModel	Specifies the data model. "Normal" or "SkewTobit" are available options.
DimRed	Logical; if 'TRUE' dimension reduction is applied. This means that not the full number (n*n) of Fourier functions is used but rather only a reduced dimensional basis of dimension 'NFour'.
NFour	If 'DimRed' is 'TRUE', this specifies the number of Fourier functions.
indEst	A vector of numbers specifying which for which parameters the posterior should be computed and which should be held fix (at their starting value). If the corresponding to the index of rho_0, sigma^2, zeta, rho_1, gamma, alpha, mu_x, mu_y, tau^2 is present in the vector, the parameter will be estimated otherwise not. Default is indEst=1:9 which means that one samples from the posterior for all parameters.
Nmc	Number of MCMC samples.
BurnIn	Length of the burn-in period.
path	Path, in case plots and / or the spateMCMC object should be save in a file.
file	File name, in case plots and / or the spateMCMC object should be save in a file.
SaveToFile	Indicates whether the spateMCMC object should be save in a file.
PlotToFile	Indicates whether the MCMC output analysis plots should be save in a file.
FixEffMetrop	The fixed effects, i.e., the regression coefficients, can either be sampled in a Gibbs step or updated together with the hyperparameters in the Metropolis-Hastings step. The latter is the default and recommended option since correlations between fixed effects and the random process can result in slow mixing.

saveProcess — Logical; if 'TRUE' samples from the posterior of the latent spatio-temporal process xi are saved.

Nsave — Number of samples from the posterior of the latent spatio-temporal process xi that should be save.

seed — Seed for random generator.

Padding — Indicates whether padding is applied or not. If the range parameters are large relative to the domain, this is recommended since otherwise spurious periodicity can occur.

adaptive — Indicates whether an adaptive Metropolis-Hastings algorithm is used or not. If yes, the proposal covariance matrix 'RWCov' is adaptively estimated during the algorithm and tuning does not need to be done by hand.

NCovEst — Minimal number of samples to be used for estimating the proposal matrix.

BurnInCovEst — Burn-in period for estimating the proposal matrix.

MultCov — Numeric used as multiplier for the adaptively estimated proposal cocariance matrix 'RWCov' of the hyper-parameters. I.e., the estimated covariance matrix is multiplied by 'MultCov'.

printRWCov — Logical, if 'TRUE' the estimated proposal cocariance matrix is printed each time.

MultStdDevLambda — Numeric used as multiplier for the adaptively estimated proposal standard deviation of the Tobit transformation parameter lambda. I.e., the estimated standard deviation is multiplied by 'MultStdDevLambda'.

Separable — Indicates whether a separable model, i.e., no transport / drift and no diffusion, should be estimated.

Drift — Indicates whether a drift term should be included.

Diffusion — Indicates whether a diffusion term should be included.

logInd — Indicates which parameters are sampled on the log-scale. Default is logInd=c(1, 2, 3, 4, 5, 9) corresponding to rho_0, sigma2, zeta, rho_1, gamma, and tau^2.

nu — Smoothness parameter of the Matern covariance function for the innovations. By default this equals 1 corresponding to the Whittle covariance function.

plotTrace — Indicates whether trace plots are made.

plotHist — Indicates whether histograms of the posterior distributions are made.

plotPairs — Indicates whether scatter plots of the hyper-parameters and the regression coefficients are made.

trueVal — In simulations, true values can be supplied for comparison with the MCMC output.

plotObsLocations — Logical; if 'TRUE' the observations locations are ploted together with the grid cells.

trace — Logical; if 'TRUE' tracing information on the progress of the MCMC algorithm is produced.

monitorProcess Logical; if 'TRUE' in addition to the trace plots of the hyper-parameters, the mixing properties of the latent process xi=Phi*alpha is monitored. This is done by plotting the current sample of the process. More specifically, the time series at locations 'sProcess' and the spatial fieldd at time points 'tProcess'.

tProcess To be secified if 'monitorProcess=TRUE'. Time points at which spatial fields of the sampled process should be plotted.

sProcess To be secified if 'monitorProcess=TRUE'. Locations at which time series of the sampled process should be plotted.

Value

The function returns a 'spateMCMC' object with, amongst others, the following entries

Post Matrix containing samples from the posterior of the hyper-parameters and the regression coefficient

xiPost Array with samples from the posterior of the spatio-temporal process

RWCov (Estimated) proposal covariance matrix

Author(s)

Fabio Sigrist

Examples

```
##Specify hyper-parameters
par <- c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01)
##Simulate data
spateSim <- spate.sim(par=par,n=20,T=20,seed=4)
w <- spateSim$w

##Below is an example to illustrate the use of the MCMC algorithm.
##In practice, more samples are needed for a sufficiently large effective sample size.

##The following takes a couple of minutes.
##Load the precomputed object some lines below to save time.
##spateMCMC <- spate.mcmc(y=w,x=NULL,SV=c(rho0=0.2,sigma2=0.1,
##               zeta=0.25,rho1=0.2,gamma=1,alpha=0.3,muX=0,muY=0,tau2=0.005),
##               RWCov=diag(c(0.005,0.005,0.05,0.005,0.005,0.001,0.0002,0.0002,0.0002)),
##               Nmc=10000,BurnIn=2000,seed=4,Padding=FALSE,plotTrace=TRUE,NCovEst=500,
##               BurnInCovEst=500,trueVal=par,saveProcess=TRUE)
##spateMCMC
##plot(spateMCMC.fit,true=par,postProcess=TRUE)

##Instead of waiting, you can also use this precomputed object
data("spateMCMC")
spateMCMC
plot(spateMCMC,true=par,medianHist=FALSE)
```

spate.plot *Plot a spatio-temporal field.*

Description

Generates a figure or an animation of a spatio-temporal field.

Usage

```
spate.plot(xi,nx=NULL,whichT=NULL,format="ImgTogether",ToFile=FALSE,path=NULL,
           file=NULL,indScale=FALSE,main=NULL,mfrow=NULL,
           imagesize=c(1000,1000),zlim=NULL,breaks=NULL,...)
```

Arguments

`xi`	A spatio-temporal field stored in an T x N matrix with columns and rows corresponding to time and space, respectively.
`nx`	Integer specifying the number of points on the x-axis. If 'NULL', a quadratic grid is assumed.
`whichT`	Vector of integers specifying the time points that are plotted. If 'NULL', all time points are plotted.
`format`	A string specifying how the spatio-temporal field should be ploted. "ImgTogether" produces one single plot containing all spatial fields at all time points. With "ImgSeparate", the spatial fields at each time point are plotted in separate plots.
`ToFile`	Indicates whether the output should be saved to a file.
`path`	Path indicating where to save the file.
`file`	File name.
`indScale`	Indicates whether the color scale for the spatial plots is the same for all time points (indScale=FALSE) or separate for each time point (indScale=TRUE).
`main`	Titles for the plots. Can be either be NULL or a character vector of length equal to the number of time points or 1.
`mfrow`	See 'par'. Can be either NULL or an integer vector of length two. If it is NULL, the functions determines mfrow automatically.
`imagesize`	The size of the .jpeg image if ToFile=TRUE.
`zlim`	Graphical parameter to be passed to 'image'. Determines the scale on the z-axis of the plots. If 'indScale=FALSE' one can specify the common scale on the z-axis of the plots through this argument.
`breaks`	Graphical parameter to be passed to 'image'.
`...`	Other graphical parameters that are passed to 'image' and 'par'.

Value

Plots illustrating a space-time field.

Author(s)

Fabio Sigrist

Examples

```
spateSim <- spate.sim(par=c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,
                      alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01),n=50,T=9)
spate.plot(spateSim$xi)
```

---

spate.predict *Obtain samples from predictive distribution in space and time.*

Description

Obtain samples from predictive distribution in space and time given the posterior of the hyperparameters.

Usage

```
spate.predict(y,tPred,sPred=NULL,xPred=NULL,yPred=NULL,spateMCMC,Nsim=200,
              BurnIn=5,coord=NULL,lengthx=NULL,lengthy=NULL,Sind=NULL,
              n=NULL,IncidenceMat=FALSE,x=NULL,DataModel="Normal",
              DimRed=FALSE,NFour=NULL,seed=NULL,nu =1,trace=FALSE)
```

Arguments

`y`	Observed data in an T x N matrix with columns and rows corresponding to time and space, respectively.
`x`	Covariates in an array of dimensions p x T X N, where p denotes the number of covariates, T the number of time points, and N the number of spatial points.
`tPred`	Time points where predictions are made.This needs to be a vector if predictions are made at multiple times. For instance, if T is the number of time points in the data 'y', then tPred=c(T+1, T+2) means that predictions are made at time 'T+1' and 'T+2'. If 'xPred' and 'yPred' are empty, then predictions are made at all spatial points for each time point in 'tPred'. Otherwise 'xPred' and 'yPred', or 'sPred, need to have the same length as 'tPred', and predictions are made at the points (tPred,xPred,yPred), ore (tPred, sPred), respectively.
`sPred`	Vector of indices of grid cells (positions of locations in the stacked spatial vector) where predictions are made. This is an alternative to specifying the coordinates 'xPred' and 'yPred'.
`xPred`	Vector of x-coordinates of spatial points where predictions are made. This is an alternative to specifying the grid cell in 'sPred'.
`yPred`	Vector of y-coordinates of spatial points where predictions are made. This is an alternative to specifying the grid cell in 'sPred'.
`spateMCMC`	'spateMCMC' object obtained from 'spate.mcmc' containing the posterior of the hyper-parameters and information on the model used.

- `Nsim` — Number of samples used to characterize the predictive distribution.
- `BurnIn` — Length of burn-in period.
- `coord` — If specified, this needs to be a matrix of dimension N x 2 with coordinates of the N observation points. Observations in 'y' can either be on a square grid or not. If not, the coordinates of each observation point need to be specified in 'coord'. According to these coordinates, each observation location is then mapped to a grid cell. If 'coord' is not specified, the observations in 'y' are assumed to lie on a square grid with each axis scaled so that it has unit length.
- `lengthx` — Use together with 'coord' to specify the length of the x-axis. This is usefull if the observations lie in a rectangular area instead of a square. The length needs to be at least as large as the largest x-distance in 'coord.
- `lengthy` — Use together with 'coord' to specify the length of the y-axis. This is usefull if the observations lie in a rectangular area instead of a square. The length needs to be at least as large as the largest y-distance in 'coord.
- `Sind` — Vector of indices of grid cells where observations are made, in case, the observation are not made at every grid cell. Alternatively, the coordinates of the observation locations can be specfied in 'coord'.
- `n` — Number of point per axis of the square into which the points are mapped. In total, the process is modeled on a grid of size n*n.
- `IncidenceMat` — Logical; if 'TRUE' and incidence matrix relating the latent process to observation locations. This is only recommended to use when the observations are relatively low-dimensional and when the latent process is modeled in a reduced dimensional spaceas well.
- `DataModel` — Specifies the data model. "Normal" or "SkewTobit".
- `DimRed` — Logical; if 'TRUE' dimension reduction is applied. This means that not the full number (n*n) of Fourier functions is used but rather only a reduced dimensional basis of dimension 'NFour'.
- `NFour` — If 'DimRed' is 'TRUE', this specifies the number of Fourier functions.
- `seed` — Seed for random generator.
- `nu` — Smoothness parameter of the Matern covariance function for the innovations. By default this equals 1 corresponding to the Whittle covariance function.
- `trace` — Logical; if 'TRUE' tracing information on the progress of the MCMC algorithm is produced.

Value

Depending on whether 'xPred' and 'yPred' are empty or not, either

- `(i)` — an array of size t x s x Nsim where the first index is for time, the second for space, and the third for the number of samples 'Nsim'

or

- `(ii)` — a matrix of size length(tPred) x Nsim

Author(s)

Fabio Sigrist

Examples

```
par <- c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01)
##Simulate data
spateSim <- spate.sim(par=par,n=20,T=20,seed=4)
w <- spateSim$w
data("spateMCMC")
##Make predictions. Takes a couple of seconds
predict <- spate.predict(y=w,tPred=(17:25),spateMCMC=spateMCMC,Nsim =200,
                      BurnIn=10,DataModel="Normal")

Pmean <- apply(predict,c(1,2),mean)
Psd <- apply(predict,c(1,2),sd)

opar <- par(no.readonly = TRUE)
par(mfrow=c(2,2))
image(1:20,1:20,matrix(w[19,],nrow=20),main="Observed field at t=19",xlab="x",ylab="y")
image(1:20,1:20,matrix(Pmean[3,],nrow=20),main="Fitted field at t=19",xlab="x",ylab="y")
image(1:20,1:20,matrix(w[20,],nrow=20),main="Observed field at t=20",xlab="x",ylab="y")
image(1:20,1:20,matrix(Pmean[4,],nrow=20),main="Fitted field at t=20",xlab="x",ylab="y")

par(mfrow=c(3,3))
zlim=c(min(Pmean),max(Pmean))
for(i in 1:9){
  image(1:20,1:20,matrix(Pmean[i,],nrow=20),zlim=zlim,
        main=paste("Mean t=",i+16,sep=""),xlab="x",ylab="y")
}

par(mfrow=c(3,3))
zlim=c(min(Psd),max(Psd))
for(i in 1:9){
  image(1:20,1:20,matrix(Psd[i,],nrow=20),zlim=zlim,
        main=paste("Std.dev. t=",i+16,sep=""),xlab="x",ylab="y")
par(opar) # Reset par() settings
}
```

---

spate.sim *Simulate from the SPDE.*

Description

Generates one sample from the Gaussian process specified through the SPDE.

Usage

```
spate.sim(par,n,T,seed=NULL,StartVal=NULL,nu=1)
```

Arguments

`par` Vector of parameters for the SPDE in the following order: rho_0, sigma^2, zeta, rho_1, gamma, alpha, mu_x, mu_y, tau^2. rho_0 and sigma^2 are the range and marginal variance of the Matern covariance funtion for the innovation term epsilon. zeta is the damping parameter. rho_1, gamma, and alpha parametrize the diffusion matrix with rho_1 being a range parameter, gamma and alpha determining the amount and the direction, respectively, of anisotropy. mu_x and mu_y are the two components of the drift vector. tau^2 denotes the variance of nugget effect or measurment error.

`n` Number of grid points on each axis. n x n is the total number of spatial points.

`T` Number of points in time.

`seed` Seed for random number generator.

`StartVal` A starting value (field) for the SPDE can be defined. This is the spatial field at the initial time that get propagated forward by the SPDE. The starting fields needs to be a stacked vector of lengths n x n (number of spatial points). Use 'as.vector()' to convert a spatial matrix to a vector.

`nu` Smoothness parameter of the Matern covariance function for the innovations. By default this equals 1 corresponding to the Whittle covariance function.

Value

A list containing a simulated spatio-temporal field xi with covariance structure as defined through the SPDE, a simulated observation field w obtained by adding a measurement error, and the simulated Fourier coefficients. The last two are returned only on demand.

Author(s)

Fabio Sigrist

Examples

```
StartVal <- rep(0,100^2)
StartVal[75*100+75] <- 1000
par <- c(rho0=0.05,sigma2=0.7^2,zeta=-log(0.99),rho1=0.06,
         gamma=3,alpha=pi/4,muX=-0.1,muY=-0.1,tau2=0.00001)
spateSim <- spate.sim(par=par,n=100,T=6,StartVal=StartVal,seed=1)
plot(spateSim,mfrow=c(2,3),mar=c(2,2,2,2),indScale=TRUE,
     cex.axis=1.5,cex.main=2)
```

`spateMCMC.RData` *'spateMCMC' object output obtained from 'spate.mcmc'.*

Description

Precalculated 'spateMCMC' object containing a fitted model (MCMC output) obtained from 'spate.mcmc'.

Usage

```
spateMCMC
```

`spateMLE.RData` *Maximum likelihood estimate for SPDE model with Gaussian observations.*

Description

Precalculated maximum likelihood estimate using 'optim' and the function 'loglike'.

Usage

```
spateMLE
```

`summary.spateSim` *Summary function for 'spateSim' objects.*

Description

Summary function for 'spateSim' objects.

Usage

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

Arguments

`object`	'spateSim' object obtained from 'spate.sim()'.
`...`	not used.

Author(s)

Fabio Sigrist

Examples

```
spateSim <- spate.sim(par=c(rho0=0.1,sigma2=0.2,zeta=0.5,rho1=0.1,gamma=2,
                      alpha=pi/4,muX=0.2,muY=-0.2,tau2=0.01),n=50,T=9)
summary(spateSim)
```

`tobit.lambda.log.full.cond` *Full conditional for transformation parameter lambda.*

Description

Full conditional for transformation parameter lambda of the Tobit model. This is used in the Metropolis-Hastings step of the MCMC algorithm.

Usage

```
tobit.lambda.log.full.cond(y, z, tau2, lambda)
```

Arguments

`y`	Observed data.
`z`	Latent Gaussian variable.
`tau2`	Value of variance (corresponds to nugget effect).
`lambda`	Value of transformation parameter lambda.

Value

Value of evaluated full conditional for transformation parameter lambda.

Author(s)

Fabio Sigrist

`trace.plot` *Trace plots for MCMC output analysis.*

Description

Trace plots for MCMC output analysis.

Usage

```
trace.plot(data, true = NULL, BurnIn = NULL,BurnInAdaptive=NULL)
```

Arguments

`data`	A p x Nmc data.frame of matrix where p denotes the number of parameters and Nmc the number of Monte Carlo samples.
`true`	The true value of the parameters (for simulation only).
`BurnIn`	The number of samples used as burn-in if the burn-in has not yet been removed from the sample.
`BurnInAdaptive`	The number of samples used as burn-in before starting the adaptive estimation of Metropolis-Hastings proposal covariance for the hyper-parameters.

Value

Trace plots.

Author(s)

Fabio Sigrist

Examples

```
data <- matrix(rnorm(1200),nrow=6)
opar <- par(no.readonly = TRUE)
par(mfrow=c(2,3))
trace.plot(data,true=rep(0,6))
par(opar) # Reset par() settings
```

`TSmat.to.vect` *Converts a matrix stacked vector.*

Description

Converts a time-space matrix with columns and rows corresponding to time and space into a stacked N*T vector.

Usage

```
TSmat.to.vect(mat)
```

Arguments

`mat`	A T x N matrix with columns and rows corresponding to time and space, respectively.

Value

A vector of stacked values. Stacking is done first over space and then time.

Author(s)

Fabio Sigrist

Examples

```
vect <- 1:12
mat <- vect.to.TSmat(vect,T=3)##Convert vector to matrix
TSmat.to.vect(mat)##Convert matrix to vector.
```

---

vect.to.TSmat *Converts a stacked vector into matrix.*

Description

Converts a stacked N*T vector into a time-space matrix with columns and rows corresponding to time and space, respectively.

Usage

```
vect.to.TSmat(vect, T = 1)
```

Arguments

vect	A vector of stacked values. Stacking is done first over space and then time.
T	Number of time points.

Value

A T x N matrix with columns and rows corresponding to time and space, respectively.

Author(s)

Fabio Sigrist

Examples

```
vect <- 1:12
vect
vect.to.TSmat(vect,T=3)
```

vnorm *Eucledian norm of a vector*

Description

Calculates the Eucledian norm of a vector

Usage

```
vnorm(v)
```

Arguments

v	Vector.

Value

The Eucledian norm of the vector 'v'.

Author(s)

Fabio Sigrist

Examples

```
v <- c(1,2)
vnorm(v)
```

wave.numbers *Wave numbers.*

Description

Returns wave numbers used in real Fourier transform.

Usage

```
wave.numbers(n)
```

Arguments

n	Number of grid points on each axis. n x n is the total number of spatial points.

Value

Returns a list with

- `wave` — A 2 x n^2 matrix with wavenumbers used in the real Fourier transform. The first four columns contain the wavenumbers that are only used by cosine terms and not by sine terms. Subsequent columns alternate between wavenumbers of cosine and sine terms.
- `indCos` — Vector of integers indicating the position of columns in 'wave' of wavenumbers of cosine terms. The first four cosine wavenumbers in 'wave' are not included in 'indCos'.

Author(s)

Fabio Sigrist

Index