Package ‘sdfEXTREME’

October 17, 2021

Type Package

Title Spatial Deformation for Non-Stationary Extremal Dependence

Version 0.1.1

Author Jordan Richards

Maintainer The package maintainer <jordan.richards@kaust.edu.sa>

Description Contains functions for creating spatial deformations to account for non-stationarity in spatial data. Both correlation based methods and methods adapted for extremal dependence are included. Accompanies the paper of the same title.

License GPL

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Depends R (>= 2.10)

Imports fields,mvtnorm,evd,rmutil,MASS,maps,mapproj,mapdata,numDeriv

NeedsCompilation no

R topics documented:

Aus_Heat 2
Aus_Heat_Output 2
brnsims 3
chi3 4
chi3.emp 5
FindSplineParFNCOR 6
FindSplineParFNEXTR 8
gamsv 9
High_Precip 11
High_Precip_Output 11
m.reject 12
makepairs 13
nllHT 14
nllIMSPexp 15
nllIMSPexpSmith 16
nllMSPexp 17
returnDcoord 19
Score_Est 20

sdf.heur.Cor . . . 22
sdf.heur.EXTR . . . 24
Snow_Precip . . . 26
Snow_Precip_Output . . . 26
stat.boot . . . 27
Vterm . . . 28

Index **29**

`Aus_Heat` *Australia Summer Temperatures*

Description

Data consist of daily summer (DJF) maximum near-surface air temperatures taken from the HadGHCND global gridded dataset and interpolated to 72 grid point locations covering Australia, for the period 1957-2014.

Usage

```
data(Aus_Heat)
```

Format

A list with 2 elements:

Temp. A matrix with 5234 rows and 72 columns of temperature data.

coords A 72 by 2 matrix of lon-lat coordinates.

References

Caesar et al. (2006) J. Geophys. Res. 111, D05101, (doi)

Examples

```
data(Aus_Heat)
```

`Aus_Heat_Output` *Australia Summer Temperatures outputs*

Description

Outputs from deformations and model fits for `data(Aus_Heat)`.

Usage

```
data(Aus_Heat_Output)
```

Format

A list with 5 elements:

likG.IMSP Optim output from fitting Inverted Smith model to G-plane sampling locations. See `nllIMSPexpSmith`.

likG.MSP Optim output from fitting Brown-Resnick model to G-plane sampling locations. See `nllMSPexpSmith`.

likD.IMSP Optim output from fitting Inverted Smith model to D-plane sampling locations. See `nllIMSPexp`.

likD.MSP Optim output from fitting Brown-Resnick model to D-plane sampling locations. See `nllMSPexp`.

sdf D-plane transformation spline parameters. See `sdf.heur.EXTR`.

References

Caesar et al. (2006) J. Geophys. Res. 111, D05101, (doi)

Examples

```
data(Aus_Heat_Output)
```

`brnsims` *Simulate non-stationary Brown-Resnick process*

Description

Simulate a non-stationary (or stationary) Brown-Resnick process using the 'stopping rule' methodology of Dieker and Mikosch (2015). Non-stationary Brown-Resnick processes are simulated using the non-stationary semivariogram. detailed in the help file for `gamsv`.

Usage

```
brnsims(reps, locs, kappa, lambda, tau, centre = NULL)
```

Arguments

`reps`	Number of realisations, n.
`locs`	A p by 2 matrix of coordinates.
`kappa`	Value of κ. See `help(gamsv)`.
`lambda`	Value of λ. See `help(gamsv)`.
`tau`	A p by p matrix of correlation values.
`centre`	Vector of length 2 giving coordinates of centre of non-stationarity for semivariogram. If `NULL`, stationary semivariogram used. See `help(gamsv)`.

Value

n by p matrix

References

Dieker and Mikosch (2015) Extremes, 18(2):301-314, (doi)

Examples

```
##Creating correlation values to simulate non-stationary Brown-Resnick process.

lambda<-2
centre<-c(0,0)
kappa<-0.8

n.grid<-8
sim.coords<-as.matrix(expand.grid(seq(-1,1,length=n.grid),seq(-1,1,length=n.grid)))

p<-dim(sim.coords)[1]
tau<-matrix(NA,nrow=p,ncol=p)

for(i in 1:p){
for(j in 1:p){
tau[i,j]<-gamsv(s1=sim.coords[i,],s2=c(0,0),lam=lambda,kap=kappa,centre=NULL)+
gamsv(s1=sim.coords[j,],s2=c(0,0),lam=lambda,kap=kappa,centre=NULL)-
gamsv(s1=sim.coords[i,],s2=sim.coords[j,],lam=lambda,kap=kappa,centre=centre)
}
}

##Simulates 10 realisations of non-stationary BR process.

Sim<-brnsims(reps=10,locs=sim.coords,kappa=kappa,lambda=lambda,centre=centre,tau=tau)
```

---

chi3 *Theoretical triple-wise χ (χ_q) for a (Inverted) Brown-Resnick process*

Description

Calculates the theoretical triple-wise $\chi(s_i, s_j, s_k)$ or $\chi_q(s_i, s_j, s_k)$ for a given 3 by 3 matrix of semivariogram. values.

Usage

```
chi3(v_H)

chi3q(v_H, q)
```

Arguments

v_H	3 by 3 matrix of semivariogram. values.
q	Exceedance threshold for $\chi_q(s_i, s_j, s_k)$.

Value

Theoretical $\chi(s_i, s_j, s_k)$ or $\chi_q(s_i, s_j, s_k)$ measure for the corresponding matrix of semivariogram values.

Examples

```
data(Aus_Heat)
data(Aus_Heat_Output)
Z<-Aus_Heat$Temp.
Gcoords<-Aus_Heat$coords
sdf<-Aus_Heat_Output$sdf
likD.MSP<-Aus_Heat_Output$likD.MSP
likD.IMSP<-Aus_Heat_Output$likD.IMSP

Dcoords<-returnDcoord(sdf$par,Gcoords,sdf$m.ind,sphere.dis=TRUE)

 ind.triple<-c(1,2,3) ##Denotes indices of triple for which triple-wise chi calculated.

 #MSP
 v_H<-(rdist.earth(Dcoords[ind.triple,],miles=F)/likD.MSP$par[2])^likD.MSP$par[1]
 print(chi3(v_H))

  #IMSP - likD.IMSP only contains range parameter as smoothing parameter is 2.
 v_H<-(rdist.earth(Dcoords[ind.triple,],miles=F)/likD.IMSP$par[1])^2
 print(chi3q(v_H,q=0.95))
```

`chi3.emp` *Empirical estimate of triple-wise chi*

Description

Calculates empirical estimate of triple-wise chi for `(U,V,W)`.

Usage

```
chi3.emp(U, V, W, q)
```

Arguments

U	Vector of length n of standard uniform variables.
V	Vector of length n of standard uniform variables.
W	Vector of length n of standard uniform variables.
q	Value of exceedance probability used in estimation.

Value

Estimate of triple-wise chi for (U, V, W).

Examples

```
data(Aus_Heat)
Z<-Aus_Heat$Temp.

unif<-function(x) rank(x)/(length(x)+1)
Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins

q<-0.95

ind.triple<-c(1,2,3) ##Denotes indices of triple for which triple-wise chi calculated.

chi3.emp(Z_U[,ind.triple[1]],Z_U[,ind.triple[2]],Z_U[,ind.triple[3]],q)
```

`FindSplineParFNCOR` *Find spline parameters (Correlation)*

Description

Find spline parameters for deformation based on correlation methods. Either use the Frobenuis norm via `type=="F-norm"` or the original Smith (1996) method via `type=="Smith"`. Output to be minimised to estimate spline parameters.

Usage

```
FindSplineParFNCOR(
  par,
  Gcoords,
  m.ind,
  n = 0,
  emp.cor,
  type = c("F-norm", "Smith"),
  sphere.dis = FALSE
)
```

Arguments

`par`	Parameter values $\phi = (b_1, b_2, \rho, \kappa, \delta_4^{(1)} ... \delta_m^{(1)}, \delta_4^{(2)} ... \delta_m^{(2)})$. If $m < 4$, δ parameters are not needed.
`Gcoords`	A `d` by 2 matrix of G-plane coordinates.
`m.ind`	A vector of length `m<d` giving the indices of the anchor points in `Gcoords`.
`n`	Number of data points used to estimate `emp.cor`. Only neccesary for `type=="Smith"`.
`emp.cor`	A `d` by `d` matrix of pairwise empirical correlation values.
`type`	`"F-norm"` for Frobenius norm method using theoretical $\rho(h_{ij}^*)$ from a stationary Matérn correlation model. `"Smith"` for original Smith (1996) method; also uses stationary Matérn correlation model.
`sphere.dis`	Is Spherical distance or Euclidean distance used?

Value

Frobenius norm of difference between theoretical $\rho(h^*_{ij})$ matrix and `emp.cor`, or negative log-likelihood for a stationary Gaussian process fit using D-plane coordinates.

Examples

```
data("Aus_Heat")

Z<-Aus_Heat$Temp.
Gcoords<-Aus_Heat$coords

Z_U<-Z

unif<-function(x){rank(x)/(length(x)+1)}

#Transform to uniform margins
for(i in 1:dim(Z_U)[2]){

  Z_U[,i]<-unif(Z[,i])
}
#Transform to Gaussian margins

Z_N<-qnorm(Z_U)

#Calculate pairwise empirical correlation
emp.cor<-matrix(rep(0,dim(Z_N)[2]^2),nrow=dim(Z_N)[2],ncol=dim(Z_N)[2])
for(i in 1:dim(Z_N)[2]){
  for(j in 1:i){

    emp.cor[i,j]<-cor(Z_N[,i],Z_N[,j])
  }

}
emp.cor<-emp.cor+t(emp.cor)
diag(emp.cor)<-diag(emp.cor)/2

# Set number of anchor points
m<-10
# Sample anchor points
m.ind<-sample(1:dim(Gcoords)[1],m,replace=FALSE)

#Transform to D-plane using Frobenius norm method
sdf<-optim(fn=FindSplineParFNCOR,par=c(0.05,0.05,0,1,rep(0,2*m-6)),
           type="F-norm",sphere.dis=TRUE,Gcoords=Gcoords,m.ind=m.ind,
           control=list(maxit=2000), emp.cor=emp.cor,method = "Nelder-Mead")

sdf<-try(optim(fn=FindSplineParFNCOR,par=sdf$par,sphere.dis=TRUE,type="F-norm",
         Gcoords=Gcoords,m.ind=m.ind, control=list(maxit=2000), emp.cor=emp.cor,
    method = "BFGS"))
sdf$m.ind<-m.ind
#Plot Dcoords

Dcoords<-returnDcoord(sdf$par,Gcoords,sdf$m.ind,sphere.dis=TRUE)
plot(Dcoords)
```

`FindSplineParFNEXTR` *Find spline parameters (Extremal)*

Description

Find spline parameters for deformation using extremal dependence methods. Provides Frobenuis norm of difference between pairwise empircal and theoretical χ measures. Output to be minimised to estimate spline parameters.

Usage

```
FindSplineParFNEXTR(
  par,
  Gcoords,
  m.ind,
  emp.dep,
  type = c("CHI_BR", "CHI_q_IBR"),
  q = 0,
  sphere.dis = FALSE
)
```

Arguments

`par`	Parameter values $\phi = (b_1, b_2, \rho, \kappa, \delta_4^{(1)} ... \delta_m^{(1)}, \delta_4^{(2)} ... \delta_m^{(2)})$. If m < 4, δ parameters are not needed.
`Gcoords`	A `d` by 2 matrix of G-plane coordinates.
`m.ind`	A vector of length `m` < `d` giving the indices of the anchor points in `Gcoords`.
`emp.dep`	A `d` by `d` matrix of pairwise empirical chi values.
`type`	`"CHI_BR"` for theoretical $\chi(h^*_{ij})$ from a Brown-Resnick model. `"CHI_q_IBR"` for theoretical $\chi_q(h^*_{ij})$ from an inverted Brown-Resnick model.
`q`	Threhold for $\chi_q(h^*_{ij})$. Only needed if `type=="CHI_q_IBR"`.
`sphere.dis`	Is Spherical distance or Euclidean distance used?

Value

Frobenius norm of difference between theoretical $\chi(h^*_{ij})$ or $\chi_q(h^*_{ij})$ matrix and `emp.chi`.

Examples

```
data("Aus_Heat")

Z<-Aus_Heat$Temp.
Gcoords<-Aus_Heat$coords

Z_U<-Z

unif<-function(x){rank(x)/(length(x)+1)}

#Transform to uniform margins
for(i in 1:dim(Z_U)[2]){
```

```
  Z_U[,i]<-unif(Z[,i])
}

chi.emp<-function(U,V,z) sum((U>=z)&(V>=z))/sum(U>=z)
q<-0.98

#Calculate pairwise empirical chi
emp.chi<-matrix(rep(0,dim(Z_U)[2]^2),nrow=dim(Z_U)[2],ncol=dim(Z_U)[2])
for(i in 1:dim(Z_U)[2]){
  for(j in 1:i){

    emp.chi[i,j]<-chi.emp(Z_U[,i],Z_U[,j],q)
  }

}
emp.chi<-emp.chi+t(emp.chi)
diag(emp.chi)<-diag(emp.chi)/2

# Set number of anchor points
m<-10
# Sample anchor points
m.ind<-sample(1:dim(Gcoords)[1],m,replace=FALSE)

#Transform to D-plane using Brown-Resnick theoretical chi function
sdf<-optim(fn=FindSplineParFNEXTR,par=c(0.05,0.05,0,1,rep(0,2*m-6)),type="CHI_BR",sphere.dis=TRUE,
           Gcoords=Gcoords,m.ind=m.ind,control=list(maxit=2000), emp.dep=emp.chi,
      method = "Nelder-Mead")

sdf<-optim(fn=FindSplineParFNEXTR,par=sdf$par,sphere.dis=TRUE,type="CHI_BR",
           Gcoords=Gcoords,m.ind=m.ind,
           control=list(maxit=2000), emp.dep=emp.chi,method = "BFGS")
sdf$m.ind<-m.ind

#Plot Dcoords

Dcoords<-returnDcoord(sdf$par,Gcoords,sdf$m.ind,sphere.dis=TRUE)
plot(Dcoords)
```

---

gamsv *Non-stationary 'centered' semivariogram.*

Description

Calculates the Fouedijo et al. (2015) theoretical semivariogram. This is a function of some central location $o =$ `centre` and distance between locations `s1` and `s2`. The semivariogram. has the form

$$\gamma^*(s_2, s_1) = \gamma(||\psi(s_2) - \psi(s_1)||),$$

where

$$\psi(s) = o + (s - o)||s - o||$$

and

$$\gamma(x) = (x/\lambda)^\kappa$$

for $\lambda > 0$ and $\kappa \in (0, 2]$.

Usage

```
gamsv(s1, s2, centre = NULL, lam, kap)
```

Arguments

- `s1` Vector of length 2 giving coordinates of first location.
- `s2` Vector of length 2 giving coordinates of second location.
- `centre` Vector of length 2 giving coordinates of centre of non-stationarity. If `NULL`, returns stationary semivariogram.
- `lam` Value of λ.
- `kap` Value of κ.

Value

Non-stationary semivariogram. between locations `s1` and `s2`

References

Fouedijo et al. (2015) Spatial Statistics, 13:45-61, (doi)

Examples

```
##Creating correlation values to simulate non-stationary Brown-Resnick process. See help(brnsims).

lambda<-2
centre<-c(0,0)
kappa<-0.8

n.grid<-8
sim.coords<-as.matrix(expand.grid(seq(-1,1,length=n.grid),seq(-1,1,length=n.grid)))

p<-dim(sim.coords)[1]
tau<-matrix(NA,nrow=p,ncol=p)

for(i in 1:p){
for(j in 1:p){
tau[i,j]<-gamsv(s1=sim.coords[i,],s2=c(0,0),lam=lambda,kap=kappa,centre=NULL)+
gamsv(s1=sim.coords[j,],s2=c(0,0),lam=lambda,kap=kappa,centre=NULL)-
gamsv(s1=sim.coords[i,],s2=sim.coords[j,],lam=lambda,kap=kappa,centre=centre)
}
}

##Simulates 10 realisations of non-stationary BR process.

Sim<-brnsims(reps=10,locs=sim.coords,kappa=kappa,lambda=lambda,centre=centre,tau=tau)
```

`High_Precip` *Highlands Precipitation Data*

Description

Data consist of winter (DJF) 12-hour average precipitation rate (mm/day) taken from the UKCP18 climate projection. Data is produced on $2.2 \times 2.2 km^2$ grid-boxes, between the years 1980 and 2000.

Usage

```
data(High_Precip)
```

Format

A list with 2 elements:

Pr. A matrix with 3600 rows and 100 columns of precipitation data.

coords A 100 by 2 matrix of lon-lat coordinates, corresponding to the centroid of each grid-box.

References

Lowe et al. (2018), Met Office, Exter, UK, (pdf)

Examples

```
data(High_Precip)
```

`High_Precip_Output` *Highlands Precipitation outputs*

Description

Outputs from deformations and model fitting for `data(High_Precip)`.

Usage

```
data(High_Precip_Output)
```

Format

A list with 5 elements:

likG.IMSP Optim output from fitting Inverted Brown-Resnick model to G-plane sampling locations. See `nllIMSPexp`.

likG.MSP Optim output from fitting Brown-Resnick model to G-plane sampling locations. See `nllMSPexp`.

likD.IMSP Optim output from fitting Inverted Brown-Resnick model to D-plane sampling locations. See `nllIMSPexp`.

likD.MSP Optim output from fitting Brown-Resnick model to D-plane sampling locations. See `nllMSPexp`.

sdf D-plane transformation spline parameters. See `sdf.heur.EXTR`.

References

Lowe et al. (2018) Met Office, Exter, UK, (pdf)

Examples

```
data(High_Precip_Output)
```

`m.reject` *A heuristic for deciding if initial anchor points have enough spread.*

Description

This heuristic ensures that chosen anchor points are not along a straight line and, if `length(m.ind)<5`, that the maximum distance between the initial anchor points in the G-plane is sufficiently large i.e. over 60% of the maximum pairwise distance of all sampling locations in the G-plane.

Usage

```
m.reject(m.ind, Gcoords, sphere.dis = FALSE)
```

Arguments

`m.ind`	A vector of indices denoting the anchor points in `Gcoords`.
`Gcoords`	A d by 2 matrix of G-plane sampling locations.
`sphere.dis`	Is Spherical distance or Euclidean distance used?

Value

`1` if anchor points are rejected, `0` otherwise.

Examples

```
data("Aus_Heat")

Gcoords<-Aus_Heat$coords
# Set number of anchor points
m<-10
# Sample anchor points
m.ind<-sample(1:dim(Gcoords)[1],m,replace=FALSE)

reject<-m.reject(m.ind,Gcoords,sphere.dis=TRUE)
print(reject)
```

makepairs *Make pairs*

Description

Produce a list of all configurations of pairwise exceedances above some threshold u. Used in censored likelihood estimation.

Usage

```
makepairs(Z, u)
```

Arguments

Z	A N by d matrix.
u	Censoring threshold.

Value

List of four data frames. Each data frame has 4 columns and the rows sum to N*d*(d-1)/2. For each data frame the first two columns are the values and the last two are the indices for the location of these values in the columns of Z.

Component 1 Pairs of observations where both components exceed u.

Component 2 Pairs of observations where the first component exceeds u and the second equals u.

Component 3 Pairs of observations where the second component exceeds u and the first equals u.

Component 4 Pairs of observations where both components equal u.

Examples

```
#For a N by d matrix of data "Z" and d by 2 matrix of coordinates "Gcoords".
# We use a very small subset of data(Aus_Heat) as an example.
##THIS WILL TAKE A LONG TIME TO RUN WITH THE FULL DATASET##

data(Aus_Heat)
Z<-Aus_Heat$Temp.[,1:3]
Gcoords<-Aus_Heat$coords[1:3,]

unif<-function(x) rank(x)/(length(x)+1)
Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins
Z_Exp<-qexp(Z_U) #Transform to exponential margins

q<-0.98
u<-quantile(Z_Exp,prob=q) # Censoring threshold
# Create a list of length 4 of pairwise exceedances and index in coordinate matrix
Zpair<-makepairs(Z_Exp,u=u)
```

`nllHT` *Negative log-likelihood for bivariate Heffernan and Tawn (2004) model*

Description

Calculates the negative log-likelihood for $Y|X = x$. This is a profile log-likelihood with free parameters α and β. We use the same normalising functions as in the original paper.

Usage

```
nllHT(X, Y, par)
```

Arguments

X	Vector of Laplace variables. This is the conditioning variable i.e. $X = x$.
Y	Vector of Laplace variables, Y.
par	Vector (α, β).

Value

Negative log-likelihood for Heffernan and Tawn model fit to $Y|X = x$.

References

Heffernan and Tawn (2004), Journal of the Royal Statistical Society: Series B, 66:497-546, (doi)

Examples

```
#For a N by d matrix of data "Z" and d by 2 matrix of coordinates "Gcoords".
# We use data(Aus_Heat) as an example.

library(rmutil)
library(fields)
data(Aus_Heat)
Z<-Aus_Heat$Temp.
Gcoords<-Aus_Heat$coords

unif<-function(x) rank(x)/(length(x)+1)
Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins
Z_LP<-qlaplace(Z_U) # Transform to Laplace margins


p<-dim(Gcoords)[1]
ConExp<-matrix(0,nrow=p,ncol=p)

u<-quantile(Z_LP,0.98) #Quantile for estimating conditional expectation

#Calculate conditional expectation for each pair
for(i in 1:p){
 for(j in 1:i){
   Exceedances<-cbind(Z_LP[,i],Z_LP[,j])[which(Z_LP[,i]>=u),]
```

```
    opt<-optim(nllHT,X=Exceedances[,1],Y=Exceedances[,2],par=c(0.3,0.8))
    #print(opt)
    alpha<-opt$par[1]
    beta<-opt$par[2]
   mu<-mean((Exceedances[,2]-alpha*Exceedances[,1])/Exceedances[,1]^beta)
   ConExp[i,j]<-alpha*u+u^beta*mu
 }

 }
 ConExp<-ConExp+t(ConExp)  ##Symmetry assumed
 diag(ConExp)<-diag(ConExp)/2

 plot(rdist.earth(Gcoords,miles=F),ConExp, ylab="Conditional Expectation",
     xlab="Distance (Km)", main="")
```

`nllIMSPexp` *Negative composite censored log-likelihood for the stationary inverted Brown-Resnick process*

Description

Calculate the negative composite censored log-likelihood for the stationary inverted Brown-Resnick process on standard exponential margins.

Usage

```
nllIMSPexp(par, ZBE, ZXE, ZYE, ZNE, coord, n.a, sphere.dis = F)
```

Arguments

par	Stationary semivariogram parameters (κ, λ).
ZBE	A list of length d with each element a matrix with 2 columns.
ZXE	A list of length d with each element a matrix with 2 columns.
ZYE	A list of length d with each element a matrix with 2 columns.
ZNE	A list of length d with each element a matrix with 2 columns.
coord	A d by 2 matrix of coordinates.
n.a	A vector with length determined by the number of unique pairs in ZNE.
sphere.dis	Is Spherical distance or Euclidean distance used?

Value

Negative composite censored log-likelihood for inverted Brown-Resnick process.

Examples

```
# For a N by d matrix of data "Z" and d by 2 matrix of coordinates "Gcoords".
# We use a very small subset of data(Aus_Heat) as an example.
##THIS WILL TAKE A LONG TIME TO RUN WITH THE FULL DATASET##

library(fields)
data(Aus_Heat)
```

```
Z<-Aus_Heat$Temp.[,1:3]
Gcoords<-Aus_Heat$coords[1:3,]

unif<-function(x) rank(x)/(length(x)+1)
Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins
Z_Exp<-qexp(Z_U) #Transform to exponential margins

q<-0.98
u<-quantile(Z_Exp,prob=q) # Censoring threshold
# Create a list of length 4 of pairwise exceedances and index in coordinate matrix
Zpair<-makepairs(Z_Exp,u=u)

#Identify number of non-exceedances per pair
#Speeds up likelihood computation
dist<-rdist(Gcoords+runif(dim(Gcoords)[1]*2,0,1))
dist2<-apply(Zpair[[4]][,3:4],1,function(x){dist[x[1],x[2]]})
unique.d<-as.matrix(unique(dist2))
n.a<-rep(0, dim(unique.d)[1])
for(i in 1:length(unique.d)){
  n.a[i]<-sum(dist2==unique.d[i,1])
}

#WARNING- pairwise likelihoods take some time to run
# Inverted Brown-Resnick process fit to G-plane coordinate system
likG.IMSP<-optim(fn=nllIMSPexp,par=c(1,500),ZBE=as.matrix(Zpair[[1]]),
           ZXE=as.matrix(Zpair[[2]]),
           ZYE=as.matrix(Zpair[[3]]),ZNE=as.matrix(Zpair[[4]]),
           n.a=n.a,sphere.dis=TRUE,coord=Gcoords,
           control=list(maxit=2000),
           method = "Nelder-Mead",hessian=TRUE)
```

`nllIMSPexpSmith` *Smith process likelihoods*

Description

Calculate the negative composite censored log-likelihood for the Smith and Inverted Smith models on standard exponential margins.

Usage

```
nllIMSPexpSmith(par, ZBE, ZXE, ZYE, ZNE, coord, n.a, sphere.dis = F)

nllMSPexpSmith(par, ZBE, ZXE, ZYE, ZNE, coord, n.a, sphere.dis = F)
```

Arguments

par	Stationary semivariogram parameter λ.
ZBE	A list of length d with each element a matrix with 2 columns.
ZXE	A list of length d with each element a matrix with 2 columns.
ZYE	A list of length d with each element a matrix with 2 columns.
ZNE	A list of length d with each element a matrix with 2 columns.

coord	A d by 2 matrix of coordinates.
n.a	A vector with length determined by the number of unique pairs in ZNE.
sphere.dis	Is Spherical distance or Euclidean distance used?

Value

Negative composite censored log-likelihood for Smith (or inverted Smith) model.

Examples

```
# For a N by d matrix of data "Z" and d by 2 matrix of coordinates "Gcoords". We use a very
# small subset of data(Aus_Heat) as an example.
##THIS WILL TAKE A LONG TIME TO RUN WITH THE FULL DATASET##

library(fields)
data(Aus_Heat)
Z<-Aus_Heat$Temp.[,1:3]
Gcoords<-Aus_Heat$coords[1:3,]

unif<-function(x) rank(x)/(length(x)+1)
Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins
Z_Exp<-qexp(Z_U) #Transform to exponential margins

q<-0.98
u<-quantile(Z_Exp,prob=q) # Censoring threshold
# Create a list of length 4 of pairwise exceedances and index in coordinate matrix
Zpair<-makepairs(Z_Exp,u=u)

#Identify number of non-exceedances per pair
#Speeds up likelihood computation
dist<-rdist(Gcoords+runif(dim(Gcoords)[1]*2,0,1))
dist2<-apply(Zpair[[4]][,3:4],1,function(x){dist[x[1],x[2]]})
unique.d<-as.matrix(unique(dist2))
n.a<-rep(0, dim(unique.d)[1])
for(i in 1:length(unique.d)){
  n.a[i]<-sum(dist2==unique.d[i,1])
}

#WARNING- pairwise likelihoods take some time to run
# Inverted Smith process fit to G-plane coordinate system
likG.INV.SMITH<-optim(fn=nllIMSPexpSmith,par=c(1000),lower=0,upper=2000,ZBE=as.matrix(Zpair[[1]]),
           ZXE=as.matrix(Zpair[[2]]),
           ZYE=as.matrix(Zpair[[3]]),ZNE=as.matrix(Zpair[[4]]),
           n.a=n.a,sphere.dis=TRUE,coord=Gcoords,
           control=list(maxit=2000),
           method = "Brent",hessian=TRUE)
```

---

nllMSPexp	*Negative composite censored log-likelihood for the stationary Brown-Resnick process*

Description

Calculate the negative composite censored log-likelihood for the stationary Brown-Resnick process on standard exponential margins.

Usage

```
nllMSPexp(par, ZBE, ZXE, ZYE, ZNE, coord, n.a, sphere.dis = F)
```

Arguments

par	Stationary semivariogram parameters (κ, λ).
ZBE	A list of length d with each element a matrix with 2 columns.
ZXE	A list of length d with each element a matrix with 2 columns.
ZYE	A list of length d with each element a matrix with 2 columns.
ZNE	A list of length d with each element a matrix with 2 columns.
coord	A d by 2 matrix of coordinates.
n.a	A vector with length determined by the number of unique pairs in ZNE.
sphere.dis	Is Spherical distance or Euclidean distance used?

Value

Negative composite censored log-likelihood for Brown-Resnick process.

Examples

```
# For a N by d matrix of data "Z" and d by 2 matrix of coordinates "Gcoords".
# We use a very small subset of data(Aus_Heat) as an example.
##THIS WILL TAKE A LONG TIME TO RUN WITH THE FULL DATASET##

library(fields)
data(Aus_Heat)
Z<-Aus_Heat$Temp.[,1:3]
Gcoords<-Aus_Heat$coords[1:3,]

unif<-function(x) rank(x)/(length(x)+1)


Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins
Z_Exp<-qexp(Z_U) #Transform to exponential margins

q<-0.98
u<-quantile(Z_Exp,prob=q) # Censoring threshold
# Create a list of length 4 of pairwise exceedances and index in coordinate matrix
Zpair<-makepairs(Z_Exp,u=u)

#Identify number of non-exceedances per pair
#Speeds up likelihood computation
dist<-rdist(Gcoords+runif(dim(Gcoords)[1]*2,0,1))
dist2<-apply(Zpair[[4]][,3:4],1,function(x){dist[x[1],x[2]]})
unique.d<-as.matrix(unique(dist2))
n.a<-rep(0, dim(unique.d)[1])
for(i in 1:length(unique.d)){
```

```
  n.a[i]<-sum(dist2==unique.d[i,1])
}

#WARNING- pairwise likelihoods take some time to run
# Brown-Resnick process fit to G-plane coordinate system
likG.MSP<-optim(fn=nllMSPexp,par=c(1.4,500),ZBE=as.matrix(Zpair[[1]]),
           ZXE=as.matrix(Zpair[[2]]),
           ZYE=as.matrix(Zpair[[3]]),ZNE=as.matrix(Zpair[[4]]),
           n.a=n.a,sphere.dis=TRUE,coord=Gcoords,
           control=list(maxit=2000),
           method = "Nelder-Mead",hessian=TRUE)
```

---

returnDcoord *Return D-plane coordinates*

Description

Returns the D-plane coordinates given spline parameter values calculated using `FindSplineParFNEXTR` or `FindSplineParFNCOR`.

Usage

```
returnDcoord(par, Gcoords, whichm, sphere.dis = F)

returnDGridcoord(par, gridcoord, Gcoords, whichm, sphere.dis = F)
```

Arguments

`par`	Spline parameter values calculated using either `FindSplineParFNEXTR` or `FindSplineParFNCOR` or `sdf.heur.Cor` or `sdf.heur.EXTR`.
`Gcoords`	A `d` by 2 matrix of G-plane sampling locations.
`whichm`	A vector of length `m < d` giving the indices of the anchor points in `Gcoords`. Must be the same as used to find `par`.
`sphere.dis`	Is Spherical distance or Euclidean distance used?
`gridcoord`	A `K` by 2 matrix of any coordinates in the G-plane.

Value

`returnDcoord` returns a `d` by 2 matrix of coordinates. `returnDGridcoord` returns a `K` by 2 matrix of coordinates.

Score_Est	*Score function estimation*

Description

For Score_Est, the score for the likelihood of a Brown-Resnick or inverted Brown-Resnick model is estimated at each time point. This can then be used to calculate the CLAIC, see examples. Score_Est_Smith estimates the score for the likelihood of a Smith or inverted Smith model.

Usage

```
Score_Est(Z_Exp, u, coord, lik, type = c("BR,IBR"), sphere.dis = F)

Score_Est_Smith(
  Z_Exp,
  u,
  coord,
  lik,
  type = c("Smith,InvSmith"),
  sphere.dis = F
)
```

Arguments

Z_Exp	A N by d matrix of exponential random variables.
u	Censoring threshold.
coord	A d by 2 matrix of coordinates.
lik	Optim output from model fitting. See help(nllMSPexp) or help(nllMSPexpSmith).
type	Either Brown-Resnick ("BR") or inverted Brown-Resnick ("IBR") for Score_Est. Smith or inverted Smith for Score_Est_Smith.
sphere.dis	Is Spherical distance or Euclidean distance used?

Value

An N by 2 matrix of score estimates for semivariogram parameters (κ, λ) (just λ for Score_Est_Smith).

Examples

```
# Z_Exp: N by d matrix of data on exponential margins
# Gcoords and Dcoords: d by 2 matrix of sampling locations in the G-plane
# and D-plane respectively. See help(returnDcoord) for obtaining Dcoords.
# likG.MSP and likD.MSP: optim outputs from model fitting. See help(nllMSPexp).

#We use data(Aus_Heat) as an example.
##THIS WILL TAKE A LONG TIME TO RUN WITH THE FULL DATASET##

data(Aus_Heat)
Z<-Aus_Heat$Temp.
Gcoords<-Aus_Heat$coords

data(Aus_Heat_Output)
```

```
sdf<-Aus_Heat_Output$sdf
likG.MSP<-Aus_Heat_Output$likG.MSP
likD.MSP<-Aus_Heat_Output$likD.MSP

Dcoords<-returnDcoord(sdf$par,Gcoords,sdf$m.ind,sphere.dis=TRUE)

 unif<-function(x) rank(x)/(length(x)+1)
Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins
Z_Exp<-qexp(Z_U) #Transform to exponential margins

q <- 0.98 # 98% quantile used as threshold in composite likelihood
u <- quantile(Z_Exp,prob=q)


s_G_MSP <- Score_Est(Z_Exp,u,coord=Gcoords,lik=likG.MSP,type="BR",sphere.dis=T)
s_D_MSP <- Score_Est(Z_Exp,u,coord=Dcoords,lik=likD.MSP,type="BR",sphere.dis=T)

#Estimate variance of score - This is specfic to the Australian summer temperatures data.
# Set block.size. Here we take 90 and 91, corresponding to a regular season
block.sizes <- c(91,90)
#and a season with a leap year
years <- 1957:2014
k <- length(years)

temp <- matrix(0,nrow=k,ncol=2)
temp2 <- matrix(0,nrow=k,ncol=2)
int <- 0
for(l in 1:k){
 if(years[l]%%4==0) block.size=block.sizes[1] else block.size=block.sizes[2]

 int <- int + block.size
 temp[l,] <- colSums(s_G_MSP[(int-block.size+1):(int),])
 temp2[l,] <- colSums(s_D_MSP[(int-block.size+1):(int),])

}

#Estimate variance of score

varS_G_MSP <- var(temp)
varS_D_MSP <- var(temp2)


#Estimate CLAIC
CLAIC_G_MSP <-2*likG.MSP$value+2*sum(diag(varS_G_MSP%*%solve(likG.MSP$hessian)))
CLAIC_D_MSP <- 2*likD.MSP$value+2*sum(diag(varS_D_MSP%*%solve(likD.MSP$hessian)))

#Estimate CLAIC for Inverted Smith model
#'# likG.IMSP and likD.IMSP: optim outputs from model fitting. See help(nllMSPexp).


likG.IMSP<-Aus_Heat_Output$likG.IMSP
likD.IMSP<-Aus_Heat_Output$likD.IMSP

q <- 0.98 # 98% quantile used as threshold in composite likelihood
u <- quantile(Z_Exp,prob=q)
s_G_IMSP <- Score_Est_Smith(Z_Exp,u,coord=Gcoords,lik=likG.IMSP,type="InvSmith",sphere.dis=T)
```

```
s_D_IMSP <- Score_Est_Smith(Z_Exp,u,coord=Dcoords,lik=likD.IMSP,type="InvSmith",sphere.dis=T)

#Estimate variance of score - This is specfic to the Australian summer temperatures data.
block.sizes <- c(91,90) #Set block.size # Here we take 90 and 91, corresponding to a regular season
#and a season with a leap year
years <- 1957:2014
k <- length(years)

temp <- matrix(0,nrow=k,ncol=1)
temp2 <- matrix(0,nrow=k,ncol=1)
int <- 0
for(l in 1:k){
 if(years[l]%%4==0) block.size=block.sizes[1] else block.size=block.sizes[2]

 int <- int + block.size
 temp[l] <- sum(s_G_IMSP[(int-block.size+1):(int)])
 temp2[l] <- sum(s_D_IMSP[(int-block.size+1):(int)])

}

#'#Estimate variance of score

varS_G_IMSP <- var(temp)
varS_D_IMSP <- var(temp2)


#Estimate CLAIC
CLAIC_G_IMSP <-2*likG.IMSP$value+2*sum(diag(varS_G_IMSP%*%solve(likG.IMSP$hessian)))
CLAIC_D_IMSP <- 2*likD.IMSP$value+2*sum(diag(varS_D_IMSP%*%solve(likD.IMSP$hessian)))
```

---

sdf.heur.Cor *Heuristic for creating bijective deformations (Correlation)*

Description

This algorithm uses `FindSplineParFNCOR` to create a deformation with `m.init` inital anchor points. The initial anchor points are the last `m.init` entries of `Full.m.ind`. After creating this deformation, the spline values are used as initial parameters in creating a deformation with `m.init+1` anchor points. This iterative procedure repeats until a deformation with all `Full.m.ind` anchor points is created.

Usage

```
sdf.heur.Cor(
  m.init,
  Full.m.ind,
  Gcoords,
  emp.cor,
  type = c("F-norm", "Smith"),
  n = 0,
  par = NULL,
  sphere.dis = F
)
```

Arguments

`m.init`	Number of inital anchor points. Must have `m.init` < `length(Full.m.ind)`.
`Full.m.ind`	Full vector of indices for anchor points in `Gcoords`.
`Gcoords`	A `d` by 2 matrix of G-plane coordinates.
`emp.cor`	A `d` by `d` matrix of pairwise empirical correlation values.
`type`	`"F-norm"` for Frobenius norm method using theoretical $\rho(h^*_{ij})$ from a Matérn correlation model. `"Smith"` for original Smith (1996) method, also using Matérn correlation model.
`n`	Number of data points used to estimate `emp.cor`. Only neccesary for `type=="Smith"`.
`par`	Initial parameters for first deformation. If not stated, initial parameters are given.
`sphere.dis`	Is Spherical distance or Euclidean distance used?

Value

List with three elements:

par Spline parameter values.

value Objective value from final optimisation.

m.ind Vector of indices for full set of anchor points.

Examples

```
data("Aus_Heat")

Z<-Aus_Heat$Temp.
Gcoords<-Aus_Heat$coords

Z_U<-Z

unif<-function(x){rank(x)/(length(x)+1)}

#Transform to uniform margins
for(i in 1:dim(Z_U)[2]){

  Z_U[,i]<-unif(Z[,i])
}


#Transform to Gaussian margins

Z_N<-qnorm(Z_U)

#Calculate pairwise empirical correlation
emp.cor<-matrix(rep(0,dim(Z_N)[2]^2),nrow=dim(Z_N)[2],ncol=dim(Z_N)[2])
for(i in 1:dim(Z_N)[2]){
  for(j in 1:i){

    emp.cor[i,j]<-cor(Z_N[,i],Z_N[,j])
  }

}
emp.cor<-emp.cor+t(emp.cor)
```

```
diag(emp.cor)<-diag(emp.cor)/2


m.init<-6
Full.m.ind<-sample(1:dim(Gcoords)[1],m.init+2)

#Transform to D-plane using Smith (1996) method

##WARNING: This may take a while to run.
sdf<-sdf.heur.Cor(m.init,Full.m.ind,Gcoords,emp.cor,type="Smith",n=dim(Z)[1],sphere.dis=TRUE)

#Plot Dcoords

Dcoords<-returnDcoord(sdf$par,Gcoords,sdf$m.ind,sphere.dis=TRUE)
plot(Dcoords,main="D-plane",ylab="",xlab="")
```

`sdf.heur.EXTR` *Heuristic for creating bijective deformations (Extremal)*

Description

This algorithm uses `FindSplineParFNEXTR` to create a deformation with `m.init` inital anchor points. The initial anchor points are the last `m.init` entries of `Full.m.ind`. After creating this deformation, the spline values are used as initial parameters in creating a deformation with `m.init + 1` anchor points. This iterative procedure repeats until a deformation with all `Full.m.ind` anchor points is created.

Usage

```
sdf.heur.EXTR(
  m.init,
  Full.m.ind,
  Gcoords,
  emp.chi,
  type = c("CHI_BR", "CHI_q_IBR"),
  q = 0,
  par = NULL,
  sphere.dis = F
)
```

Arguments

`m.init`	Number of inital anchor points. Must have `m.init` < `length(Full.m.ind)`.
`Full.m.ind`	Full vector of indices for anchor points in `Gcoords`.
`Gcoords`	A `d` by 2 matrix of G-plane coordinates.
`emp.chi`	A `d` by `d` matrix of pairwise empirical chi values.
`type`	`"CHI_BR"` for theoretical $\chi(h^*_{ij})$ from a Brown-Resnick model. `"CHI_q_IBR"` for theoretical $\chi_q(h^*_{ij})$ from an inverted Brown-Resnick model.
`q`	Threhold for $\chi_q(h^*_i j)$. Only needed if `type=="CHI_q_IBR"`.
`par`	Initial parameters for first deformation. If not stated, initial parameters are given.
`sphere.dis`	Is Spherical distance or Euclidean distance used?

Value

List with three elements:

par Spline parameter values.

value Objective value from final optimisation.

m.ind Vector of indices for full set of anchor points.

Examples

```
# Note that the given code will not produce the Australian Summer Temperature deformation
# used in the paper. See Data_Example.R and help(Aus_Heat_Output) for the deformation
# used in the paper. This deformation will use much fewer anchor points as the full
# deformation takes a long time to run.

data("Aus_Heat")

Z<-Aus_Heat$Temp.
Gcoords<-Aus_Heat$coords

Z_U<-Z

unif<-function(x){rank(x)/(length(x)+1)}

#Transform to uniform margins
for(i in 1:dim(Z_U)[2]){

  Z_U[,i]<-unif(Z[,i])
}

chi.emp<-function(U,V,z) sum((U>=z)&(V>=z))/sum(U>=z)
q<-0.98

#Calculate pairwise empirical chi
emp.chi<-matrix(rep(0,dim(Z_U)[2]^2),nrow=dim(Z_U)[2],ncol=dim(Z_U)[2])
for(i in 1:dim(Z_U)[2]){
  for(j in 1:i){

    emp.chi[i,j]<-chi.emp(Z_U[,i],Z_U[,j],q)
  }

}
emp.chi<-emp.chi+t(emp.chi)
diag(emp.chi)<-diag(emp.chi)/2

m.init<-6
Full.m.ind<-sample(1:dim(Gcoords)[1],m.init+2)

#Transform to D-plane using Brown-Resnick theoretical chi function

##WARNING: This may take a while to run.
sdf<-sdf.heur.EXTR(m.init,Full.m.ind,Gcoords,emp.chi,type="CHI_BR",
                     par=c(.05,.05,0.2,1.6,rep(0,2*m.init-6)),sphere.dis=TRUE)

#Plot Dcoords

Dcoords<-returnDcoord(sdf$par,Gcoords,sdf$m.ind,sphere.dis=TRUE)
```

```
plot(Dcoords,main="D-plane",ylab="",xlab="")
```

`Snow_Precip` *Snowdonia Precipitation Data*

Description

Data consist of winter (DJF) 12-hour average precipitation rate (mm/day) taken from the UKCP18 climate projection. Data is produced on $2.2 \times 2.2km^2$ grid-boxes, between the years 1980 and 2000.

Usage

```
data(Snow_Precip)
```

Format

A list with 2 elements:

Pr. A matrix with 3600 rows and 100 columns of precipitation data.

coords A 100 by 2 matrix of lon-lat coordinates, corresponding to the centroid of each grid-box.

References

Lowe et al. (2018), Met Office, Exter, UK, (pdf)

Examples

```
data(Snow_Precip)
```

`Snow_Precip_Output` *Snowdonia Precipitation outputs*

Description

Outputs from deformations and model fits for `data(Snow_Precip)`.

Usage

```
data(Snow_Precip_Output)
```

Format

A list with 5 elements:

likG.IMSP Optim output from fitting Inverted Brown-Resnick model to G-plane sampling locations. See `nllIMSPexp`.

likG.MSP Optim output from fitting Brown-Resnick model to G-plane sampling locations. See `nllMSPexp`.

likD.IMSP Optim output from fitting Inverted Brown-Resnick model to D-plane sampling locations. See `nllIMSPexp`.

likD.MSP Optim output from fitting Brown-Resnick model to D-plane sampling locations. See `nllMSPexp`.

sdf D-plane transformation spline parameters. See `sdf.heur.EXTR`.

References

Lowe et al. (2018) Met Office, Exter, UK, (pdf)

Examples

```
data(Snow_Precip_Output)
```

---

stat.boot *Stationary Bootstrap*

Description

Creates one bootstrap sample of a N by d matrix, `X`. Here N denotes the number of time points and d the number of sampling locations. The stationary bootstrap (Politis and Romano, 1994) generates a bootstrap sample by repeated sampling of random blocks with expected value `block.size` until a sample of length N has been created.

Usage

```
stat.boot(X, mean.block.size)
```

Arguments

`X`	Matrix with N rows corresponding to time points and d columns corresponding to spatial locations.
`mean.block.size`	Expected value of temporal block size.

References

Politis and Romano (1994), JASA,5(4):303-336, (doi)

Examples

```
\eqn{N}{} by \eqn{d}{} matrix

data(Aus_Heat)
Z<-Aus_Heat$Temp.

unif<-function(x) rank(x)/(length(x)+1)
Z_U<-Z
for(i in 1:dim(Z_U)[2]) Z_U[,i]<-unif(Z[,i]) # Transform to uniform margins

q<-0.95

ind.triple<-c(1,2,3) ##Denotes indices of triple for which triple-wise chi calculated.

block.mean<-14 #Mean block size for random block choice - Here a fortnight

boot <- stat.boot(Z_U[,ind.triple],block.mean)

#Create one bootstrap estimate of triple-wise chi
chi3.emp(boot[,1],boot[,2],boot[,3],q)
```

`Vterm` *Exponent functions*

Description

Exponent function for the Brown-Resnick process and its first- and second-order partial derivatives. For use in likelihoods.

Usage

```
Vterm(x, y, a)

Vpart(x, y, a)

Vpart2(x, y, a)
```

Arguments

`x`	`x` component.
`y`	`y` component.
`a`	$\sqrt{2\gamma(s_x - s_y)}$, where s_x and s_y are the sampling locations for the x and y components.

Value

Value

Index

∗ **datasets**
Aus_Heat, 2
High_Precip, 11
Snow_Precip, 26
∗ **output**
Aus_Heat_Output, 2
High_Precip_Output, 11
Snow_Precip_Output, 26

Aus_Heat, 2
Aus_Heat_Output, 2

brnsims, 3

chi3, 4
chi3.emp, 5
chi3q *(*chi3*)*, 4

FindSplineParFNCOR, 6
FindSplineParFNEXTR, 8

gamsv, 9

High_Precip, 11
High_Precip_Output, 11

m.reject, 12
makepairs, 13

nllHT, 14
nllIMSPexp, 15
nllIMSPexpSmith, 16
nllMSPexp, 17
nllMSPexpSmith *(*nllIMSPexpSmith*)*, 16

returnDcoord, 19
returnDGridcoord *(*returnDcoord*)*, 19

Score_Est, 20
Score_Est_Smith *(*Score_Est*)*, 20
sdf.heur.Cor, 22
sdf.heur.EXTR, 24
Snow_Precip, 26
Snow_Precip_Output, 26
stat.boot, 27

Vpart *(*Vterm*)*, 28
Vpart2 *(*Vterm*)*, 28
Vterm, 28