

Package ‘STBEU’

November 27, 2018

Version 1.0.0

Date 2017-09-13

Title Space-Time Blockwise Euclidean Likelihood

Author Moreno Bevilacqua [aut, cre],
Victor Morales-Onate [aut]

Maintainer Victor Morales Onate <victor.morales@uv.cl>

Depends R (>= 2.12.0)

Description This package provides a set of procedures modeling Gaussian Random Fields using Blockwise Euclidean Likelihood using OpenCL

Imports methods, CompRandFld

Suggests spam, scatterplot3d, fields, mapproj, gsl, plot3D, shape,
sphereplot

License GPL (>= 2)

URL <https://github.com/vmoprojs/STBEU>

Repository Github

R topics documented:

DevOpenCL	1
STBEUFit	2
STBEUTimeEvalOcl	16
SubSampTimeEval	20
Index	25

DevOpenCL	<i>Prints Device Information</i>
-----------	----------------------------------

Description

Prints the device details available in your computer. Device name, Max compute units, whether it supports double precision, among others.

Usage

DevOpenCL()

Details

The user can take this information into account so that the local parameter is set up in GeoFit when GPU computation is chosen.

Author(s)

Moreno Bevilacqua, <moreno.bevilacqua@uv.cl>, <https://sites.google.com/a/uv.cl/moreno-bevilacqua/home>, Víctor Morales Oñate, <victor.morales@uv.cl>, <https://sites.google.com/site/moralesonatevictor/>

Examples

```
library(STBEU)
DevOpenCL()
```

STBEUFit

Blowise Eucliden Likelihood Fitting of Gaussian Random Fields.

Description

Blowise Eucliden Likelihood Fitting of Gaussian Random Fields. The function returns the model parameters' estimates and the estimates' variances. Moreover the function allows to fix any of the parameters in the optimization.

Usage

```
STBEUFit(theta, fix=NULL, coords, times, cc, data, type_dist=1, maxdist=NULL, maxtime=NULL, winc_s=NULL,
          winc_t=NULL, winstp_t=NULL, subs=NULL, weighted=FALSE, local=c(1,1), GPU=NULL)
```

Arguments

theta	A vector of starting parametes for estimation
fixed	An optional named vector giving the values of the parameters that will be considered as known values. The default is NULL
coords	A numeric ($d \times 2$)-matrix (where d is the number of spatial sites).
times	A numeric vector assigning 1-dimension of temporal coordinates.
cc	Numerical, the name of a correlation model. Values (1, 2) are allowed for Double Exponential and Gneting respectively. See the Section Details .
data	A t -matrix (a single spatial realisation) where d is the number of spatial sites and t is the number of temporal coordinates. For the description see the Section Details .
type_dist	Numerical; 1 is Euclidean and 2 is Geodesic. The default is 1, the euclidean distance. See the Section Details .
maxdist	Numeric; an optional positive value indicating the maximum spatial distance considered in the composite likelihood computation. See the Section Details for more information.
maxtime	Numeric; an optional positive value indicating the maximum temporal separation considered in the euclidean likelihood computation (see Details).

winc_s	Numeric; a positive value for computing the spatial sub-window in the sub-sampling procedure. See Details for more information.
winstp_s	Numeric; a value in $(0, 1]$ for defining the the proportion of overlapping in the spatial sub-sampling procedure. The case 1 correspond to no overlapping. See Details for more information.
winc_t	Numeric; a positive value for computing the temporal sub-window in the sub-sampling procedure. See Details for more information.
winstp_t	Numeric; a value in $(0, 1]$ for defining the the proportion of overlapping in the temporal sub-sampling procedure. The case 1 correspond to no overlapping. See Details for more information.
subs	Numeric; a value in $(1, 2, 3)$ for defining the type of sub-sampling procedure. Cases are space, time and spacetime respectively. See Details for more information.
weighted	Logical; if TRUE the likelihood objects are weighted, see the Section Details . If FALSE (the default) the euclidean likelihood is not weighted.
local	Numeric; number of local work-items of the OpenCL setup. Default is $c(1, 1)$
GPU	Numeric; if NULL (the default) no GPU computation is performed.

Details

The optimization method is Nelder-mead.

The cc corrmmodel parameter allows to select a specific correlation function for the RF. Options are: 1 for Double Exponential and 2 for Gneting.

The distance parameter allows to consider differents kinds of spatial distances. The settings alternatives are:

1. Eucl, the euclidean distance (default value);
2. Geod, the geodesic distance;

The subs parameter represents the subsampling configurations. The settings alternatives are:

1. Spatial, preferred option when the number of space sites is greater than temporal coordinates;
2. Time, preferred option when the number of temporal coordinates is greater than space sites;
3. Spacetime, preferred option when the number of space and temporal are balaced.

All the nuisance and covariance parameters must be specified by the user using the start and the fixed parameter. Specifically:

The start parameter allows to specify (as starting values for the optimization) the parameters to be estimated. The fixed parameter allows to fix some of the parameters.

The maxdist parameter set the maximum spatial distance below which pairs of sites with inferior distances are considered in the euclidean-likelihood. This can be lower of the maximum spatial distance. **Note** that this corresponds to use a weighted composite-likelihood with binary weights. Pairs with distance less than maxdist have weight 1 and are included in the likelihood computation, instead those with greater distance have weight 0 and then excluded. The default is NULL, in this case the effective maximum spatial distance between sites is considered.

The same arguments of maxdist are valid for maxtime but here the weigthted composite-likelihood regards the case of spatial-temporal field. At the moment is implemented only for Gaussian RFs. The default is NULL, in this case the effective maximum temporal lag between pairs of observations is considered.

The weighted parameter specifies if the likelihoods forming the composite-likelihood must be weighted. If TRUE the weights are selected by opportune procedures that improve the efficient of the maximum composite-likelihood estimator (not implemented yet). If FALSE the efficient improvement procedure is not used.

For computing the standard errors by the sub-sampling procedure, winconst and winstp parameters represent respectively a positive constant used to determine the sub-window size and the step with which the sub-window moves.

In the spatio-temporal case the subsampling is meant only in time as described by Li et al. (2007). Thus, winconst represents the length of the temporal sub-window. By default the size of the sub-window is computed following the rule established in Li et al. (2007). By default winstp is the time step.

Value

Returns an object of class STBEUFit. An object of class STBEUFit is a list containing at most the following components:

param The vector of parameters' estimates;

Author(s)

Moreno Bevilacqua, <moreno.bevilacqua@uv.cl>, <https://sites.google.com/a/uv.cl/moreno-bevilacqua/home>, Víctor Morales Oñate, <victor.morales@uv.cl>, <https://sites.google.com/site/moralesonatevictor/>

References

Composite-likelihood:

Varin, C., Reid, N. and Firth, D. (2011). An Overview of Composite Likelihood Methods. *Statistica Sinica*, **21**, 5–42.

Varin, C. and Vidoni, P. (2005) A Note on Composite Likelihood Inference and Model Selection. *Biometrika*, **92**, 519–528.

Weighted Composite-likelihood for binary RFs:

Patrick, J. H. and Subhash, R. L. (1998) A Composite Likelihood Approach to Binary Spatial Data. *Journal of the American Statistical Association, Theory & Methods*, **93**, 1099–1111.

Weighted Composite-likelihood for Gaussian RFs:

Bevilacqua, M. Gaetan, C., Mateu, J. and Porcu, E. (2012) Estimating space and space-time covariance functions for large data sets: a weighted composite likelihood approach. *Journal of the American Statistical Association, Theory & Methods*, **107**, 268–280.

Bevilacqua, M. Gaetan, C. (2013) On composite likelihood inference based on pairs for spatial Gaussian RFs *Technical Report, Department of Statistics, de Valparaiso University*.

Sub-sampling estimation:

Carlstein, E. (1986) The Use of Subseries Values for Estimating the Variance. *The Annals of Statistics*, **14**, 1171–1179.

Heagerty, P. J. and Lumley T. (2000) Window Subsampling of Estimating Functions with Application to Regression Models. *Journal of the American Statistical Association, Theory & Methods*, **95**, 197–211.

Lee, Y. D. and Lahiri S. N. (2002) Variogram Fitting by Spatial Subsampling. *Journal of the Royal Statistical Society. Series B*, **64**, 837–854.

Li, B., Genton, M. G. and Sherman, M. (2007). A nonparametric assessment of properties of space-time covariance functions. *Journal of the American Statistical Association*, **102**, 736–744

Examples

```
#####***** SPACE ***#####

#=====Double Exponential=====#

rm(list=ls())
#####R package#####
require(GeoModels)
require(MCMCpack)
require(STBEU)
####location sites #####
lambda=8
xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))   ###regular
# plot(coords)
#set.seed(15)                                     ### not regular
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))   ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
####temporal instants #####
times=seq(1,5,1)
#####

type_dist=1          ### type of distance      1=euclidean 2=chordal 3=geodesic
maxdist=2            ## compact support in weights function for pairwise liklihood
maxtime=2

model=1 # 1=double exponential      2 =gneiting

if (model == 1) {
  # exponential model
  cov.model <- "exp_exp"
  cc=1
  #####
  mean=0
  nugget=0
  scale_s<-1.5/3
  scale_t<-1.5/3
  sill=1
  param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill)
  fixed=list(nugget=0)
  fix=c(nugget=nugget)
}
#####
set.seed(276)
#####
# Simulation of the spatial Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
mm=mean(c(data))
vv=var(c(data))
##### Composite likelihood based on pairs estimation #####

start=list(mean=mm,scale_s=scale_s,scale_t=scale_t,sill=vv)
# Maximum composite-likelihood fitting of the random field:
fit <- GeoFit(data=data,coordx=coords,coordt=times,
```

```

corrmmodel=cov.model,maxtime=maxtime,maxdist=maxdist,
likelihood="Marginal",type="Pairwise",
start=start,fixed=fixed,weighted=TRUE)

# Results:
print(fit$param)
unlist(start)
#####
# parameters for the subsampling ####
#####
coordx=coords[,1]
coordy=coords[,2]
LX=abs(range(coordx)[1]-range(coordx)[2])
LY=abs(range(coordy)[1]-range(coordy)[2])

lato_fin=3 #changing window size
lx=lato_fin      #lunghezza lato x quadrato subfinestra
ly=lato_fin      #lunghezza lato y quadrato subfinestra
winc=c(lx/sqrt(LX),ly/sqrt(LY))
winstp= 1 ### 1/lato_fin complete overlapping 1 "no" overlapping
#####

#####
theta=start      #starting value

weighted=0
### eucliden likelihood #####
type_subs=1 ### type of subsampling 1=in space 2= in time
tCPU = proc.time()
# names(fix) = c("nugget");names(theta) = c("mean","sill","scale_s","scale_t")
res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,0,0,type_subs,weighted,v)
tCPU = proc.time()-tCPU;tCPU
eu_par=c(res$par[1],res$par[3],res$par[4],res$par[2])
names(eu_par)=names(fit$param)
print(eu_par)

res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,0,0,type_subs,weighted,v)

### OpenCL eucliden likelihood #####
local <- c(1,1)
GPU <- 0
tGPU = proc.time()
res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,0,0,type_subs,weighted,v)
tGPU = proc.time()-tGPU;tGPU
eu_par1=c(res1$par[1],res1$par[3],res1$par[4],res1$par[2])
names(eu_par1)=names(fit$param)
print(eu_par1)
tCPU;tGPU

#=====Gneiting=====#

```

```

rm(list=ls())

#####R package#####
require(GeoModels)
require(MCMCpack)
require(STBEU)
####location sites #####
lambda=8
xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))   ###regular
nrow(coords)
# plot(coords)
#set.seed(15)                                     ### not regular
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))   ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
####temporal instants #####
times=seq(1,5,1)
#####

type_dist=1          ### type of distance      1=euclidean 2=chordal 3=geodesic

maxdist=2            ## compact support in weights function for pairwise likelihood
maxtime=2

model=2 # 1=double exponential      2 =gneiting

if (model == 2) {
  # gneiting model
  cov.model <-"gneiting"
  cc=2
  #####
  mean=0
  nugget=0
  scale_s<-1.5/3
  scale_t<-1.5/20
  sill=4
  power_s=1;power_t=1;sep=0.5
  param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill,
             power_s=power_s,power_t=power_t,sep=sep)
  fixed=list(nugget=0,power_s=power_s,power_t=power_t,sep=sep)
  fix=c(nugget = nugget, power_s = power_s, power_t = power_t,sep = sep)
}
#####

set.seed(276)
#####
# Simulation of the spatial Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
mm=mean(c(data))
vv=var(c(data))
#####
#####

```

```
##### Composite likelihood based on pairs estimation #####
```

```
start=list(mean=mm,scale_s=scale_s,scale_t=scale_t,sill=vv)
# Maximum composite-likelihood fitting of the random field:
fit <- GeoFit(data=data,coordx=coords,coordt=times,
              corrmodel=cov.model,maxtime=maxtime,maxdist=maxdist,
              likelihood="Marginal",type="Pairwise",
              start=start,fixed=fixed,weighted=TRUE)
```

```
# Results:
```

```
print(fit$param)
```

```
unlist(start)
```

```
#####
```

```
# parameters for the subsampling ####
```

```
#####
```

```
coordx=coords[,1]
```

```
coordy=coords[,2]
```

```
LX=abs(range(coordx)[1]-range(coordx)[2])
```

```
LY=abs(range(coordy)[1]-range(coordy)[2])
```

```
lato_fin=3 #changing window size
```

```
lx=lato_fin #lunghezza lato x quadrato subfinestra
```

```
ly=lato_fin #lunghezza lato y quadrato subfinestra
```

```
winc=c(lx/sqrt(LX),ly/sqrt(LY))
```

```
winstp= 1 ### 1/lato_fin complete overlapping 1 "no" overlapping
```

```
#####
```

```
#####
```

```
theta=start #starting value
```

```
weighted=0
```

```
### eucliden likelihood #####
```

```
type_subs=1 ### type of subsampling 1=in space 2= in time
```

```
tCPU = proc.time()
```

```
res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,0,0,type_subs,weighted,v)
```

```
tCPU = proc.time()-tCPU;tCPU
```

```
eu_par=c(res$par[1],res$par[3],res$par[4],res$par[2])
```

```
names(eu_par)=names(fit$param)
```

```
print(eu_par)
```

```
res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,0,0,type_subs,weighted,v)
```

```
### OpenCL eucliden likelihood #####
```

```
local <- c(1,1)
```

```
GPU <- 0
```

```
tGPU = proc.time()
```

```
res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,0,0,type_subs,weighted,v)
```

```
tGPU = proc.time()-tGPU;tGPU
```

```
eu_par1=c(res1$par[1],res1$par[3],res1$par[4],res1$par[2])
```

```
names(eu_par1)=names(fit$param)
```

```
print(eu_par1)
```

```
tCPU;tGPU
```

```
res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,0,0,type_subs,weighted,v)
```



```
#####***** TIME ***#####
#=====Double Exponential=====#
rm(list=ls())
#####R package#####
require(GeoModels)
require(MCMCpack)
require(STBEU)
####location sites #####
lambda=2
xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))   ###regular
nrow(coords)

#set.seed(15)                                     ### not regular
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))   ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
####temporal instants #####
times=seq(1,100,1)

type_dist=1          ### type of distance      1=euclidean 2=chordal 3=geodesic
maxdist=2
maxtime=2
#####
#####
model=1 # 1=double exponential      2 =gneiting

if (model == 1) {
  # exponential model
  cov.model <- "exp_exp"
  cc=1
  #####
  mean=0
  nugget=0
  scale_s<-1.5/3
  scale_t<-1.5/3
  sill=4
  param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill)
  fixed=list(nugget=0)
  fix=c(nugget = nugget)
}
#####
#####
# Simulation of the spatial Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
mm=mean(c(data))
vv=var(c(data))
#####

##### Composite likelihood based on pairs estimation #####
start=list(mean=mm,scale_s=scale_s,scale_t=scale_t,sill=vv)
# Maximum composite-likelihood fitting of the random field:
```

```

fit <- GeoFit(data=data,coordx=coords,coordt=times,
              corrmodel=cov.model,maxtime=maxtime,maxdist=maxdist,
              likelihood="Marginal",type="Pairwise",
              start=start,fixed=fixed,weighted=TRUE)

# Results:
print(fit$param)
unlist(start)
#####
#   parameters for the temporal subsampling ####
#####

winc=5      ### length of temporal window
winstp=1    ### 0.5 half overlapping 1 "no" overlapping
#####

#####
theta=start          #starting value
weighted=0
### eucliden likelihood #####
type_subs=2      ### type of subsampling 1=in space    2= in time
tCPU = proc.time()
res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)
tCPU = proc.time()-tCPU;tCPU
eu_par=c(res$par[1],res$par[3],res$par[4],res$par[2])
names(eu_par)=names(fit$param)
print(eu_par)

res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)

### OpenCL eucliden likelihood #####
local <- c(1,1)
GPU <- 0
tGPU = proc.time()
res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)
tGPU = proc.time()-tGPU;tGPU
eu_par1=c(res1$par[1],res1$par[3],res1$par[4],res1$par[2])
names(eu_par1)=names(fit$param)
print(eu_par1)
tCPU;tGPU

res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)

#=====Gneiting=====#

rm(list=ls())
#####R package#####
require(GeoModels)
require(MCMCpack)
require(STBEU)
###location sites #####
lambda=2

```

```

xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))   ###regular
nrow(coords)

#set.seed(15)                               ### not regular
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))   ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
####temporal instants #####
times=seq(1,100,1)

type_dist=1                               ### type of distance    1=euclidean 2=chordal 3=geodesic
maxdist=2
maxtime=2
#####
model=2 # 1=double exponential      2 =gneiting

if (model == 2) {
  # gneiting model
  cov.model <- "gneiting"
  cc=2
  #####
  mean=0
  nugget=0
  scale_s<-1.5/3
  scale_t<-1.5/20
  sill=4
  power_s=1;power_t=1;sep=0.5
  param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill,
             power_s=power_s,power_t=power_t,sep=sep)
  fixed=list(nugget=0,power_s=power_s,power_t=power_t,sep=sep)
  fix=c(nugget = nugget, power_s = power_s, power_t = power_t,sep = sep)
}
#####
#####
# Simulation of the spatial Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
mm=mean(c(data))
vv=var(c(data))
#####
#####

##### Composite likelihood based on pairs estimation #####
start=list(mean=mm,scale_s=scale_s,scale_t=scale_t,sill=vv)
# Maximum composite-likelihood fitting of the random field:
fit <- GeoFit(data=data,coordx=coords,coordt=times,
             corrmodel=cov.model,maxtime=maxtime,maxdist=maxdist,
             likelihood="Marginal",type="Pairwise",
             start=start,fixed=fixed,weighted=TRUE)

# Results:
print(fit$param)
unlist(start)
#####
# parameters for the temporal subsampling #####

```

```
#####

winc=4    ### length of temporal window
winstp=1 ### 0.5 half overlapping 1 "no" overlapping
#####
theta=start          #starting value
weighted=0
### eucliden likelihood #####
type_subs=2    ### type of subsampling 1=in space    2= in time
tCPU = proc.time()
res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)
tCPU = proc.time()-tCPU;tCPU
eu_par=c(res$par[1],res$par[3],res$par[4],res$par[2])
names(eu_par)=names(fit$param)
print(eu_par)

res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)

### OpenCL eucliden likelihood #####
local <- c(1,1)
GPU <- 0
tGPU = proc.time()
res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)
tGPU = proc.time()-tGPU;tGPU
eu_par1=c(res1$par[1],res1$par[3],res1$par[4],res1$par[2])
names(eu_par1)=names(fit$param)
print(eu_par1)
tCPU;tGPU

res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,0,0,winc,winstp,type_subs,weighted,v)

#####***** SPACETIME ***#####
#####Double Exponential#####
rm(list=ls())
#####R package#####
require(GeoModels)
require(MCMCpack)
require(STBEU)
####location sites #####
lambda=8
xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))    ###regular
nrow(coords)
# plot(coords)
```

```

#set.seed(15)                                     ### not regular
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))    ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
#####temporal instants #####
times=seq(1,5,1)

type_dist=1          ### type of distance      1=euclidean 2=chordal 3=geodesic
maxdist=2
maxtime=2

#####
model=1 # 1=double exponential      2 =gneiting

if (model == 1) {
  # exponential model
  cov.model <- "exp_exp"
  cc=1
  #####
  mean=0
  nugget=0
  scale_s<-1.5/3
  scale_t<-1.5/3
  sill=4
  param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill)
  fixed=list(nugget=0)
  fix=c(nugget = nugget)
}

#####
set.seed(276)
#####
# Simulation of the Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
#####
mm=mean(c(data))
vv=var(c(data))
##### Composite likelihood based on pairs estimation #####
start=list(mean=mm,scale_s=scale_s,scale_t=scale_t,sill=vv)

# Maximum composite-likelihood fitting of the random field:
fit <- GeoFit(data=data,coordx=coords,coordt=times,
              corrmodel=cov.model,maxtime=maxtime,maxdist=maxdist,
              likelihood="Marginal",type="Pairwise",
              start=start,fixed=fixed,weighted=TRUE)

# Results:
print(fit$param)
unlist(start)
#####
# parameters for the subsampling ###
#####
coordx=coords[,1]
coordy=coords[,2]
LX=abs(range(coordx)[1]-range(coordx)[2])
LY=abs(range(coordy)[1]-range(coordy)[2])
lato_fin=3 #changing window size

```

```

lx=lato_fin          #lunghezza lato x quadrato subfinestra
ly=lato_fin          #lunghezza lato y quadrato subfinestra
winc=c(lx/sqrt(LX),ly/sqrt(LY))
winstp= 1### 1/lato_fin complete overlapping in space 1 "no" overlapping in space
#####
winc_t=4  ### length of temporal window
winstp_t=1 ### 0.5 half overlapping 1 "no" overlapping
#####

theta=start          #starting value
weighted=0           #weigthed version 1=yes 0=no
### eucliden likelihood #####
type_subs=3  ### type of subsampling 1=in space 2= in time 3 =spacetime
tCPU = proc.time()
res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)
tCPU = proc.time()-tCPU;tCPU
eu_par=c(res$par[1],res$par[3],res$par[4],res$par[2])
names(eu_par)=names(fit$param)
print(eu_par)

res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)

### OpenCL eucliden likelihood #####
local <- c(1,1)
GPU <- 0
tGPU = proc.time()
res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)
tGPU = proc.time()-tGPU;tGPU
eu_par1=c(res1$par[1],res1$par[3],res1$par[4],res1$par[2])
names(eu_par1)=names(fit$param)
print(eu_par1)
tCPU;tGPU

res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)

#####Gneting#####

rm(list=ls())
#####R package#####
require(GeoModels)
require(MCMCpack)
require(STBEU)
###location sites #####
lambda=8
xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))  ###regular
nrow(coords)
# plot(coords)
#set.seed(15)
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))  ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
###temporal instants #####

```

```

times=seq(1,5,1)

type_dist=1          ### type of distance      1=euclidean 2=chordal 3=geodesic
maxdist=2
maxtime=2
#####
model=2 # 1=double exponential      2 =gneiting

if (model == 2) {
  # gneiting model
  cov.model <- "gneiting"
  cc=2
  #####
  mean=0
  nugget=0
  scale_s<-1.5/3
  scale_t<-1.5/20
  sill=4
  power_s=1;power_t=1;sep=0.5
  param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill,
             power_s=power_s,power_t=power_t,sep=sep)
  fixed=list(nugget=0,power_s=power_s,power_t=power_t,sep=sep)
  fix=c(nugget = nugget, power_s = power_s, power_t = power_t,sep=sep)
}
#####
set.seed(276)
#####
# Simulation of the Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
#####
#####
mm=mean(c(data))
vv=var(c(data))
##### Composite likelihood based on pairs estimation #####
start=list(mean=mm,scale_s=scale_s,scale_t=scale_t,sill=vv)

# Maximum composite-likelihood fitting of the random field:
fit <- GeoFit(data=data,coordx=coords,coordt=times,
              corrmodel=cov.model,maxtime=maxtime,maxdist=maxdist,
              likelihood="Marginal",type="Pairwise",
              start=start,fixed=fixed,weighted=TRUE)

# Results:
print(fit$param)
unlist(start)
#####
# parameters for the subsampling ###
#####
coordx=coords[,1]
coordy=coords[,2]
LX=abs(range(coordx)[1]-range(coordx)[2])
LY=abs(range(coordy)[1]-range(coordy)[2])
lato_fin=3 #changing window size
lx=lato_fin      #lunghezza lato x quadrato subfinestra
ly=lato_fin      #lunghezza lato y quadrato subfinestra
winc=c(lx/sqrt(LX),ly/sqrt(LY))

```

```

winstp= 1### 1/lato_fin complete overlapping in space 1 "no" overlapping in space
#####
winc_t=4 ### length of temporal window
winstp_t=1 ### 0.5 half overlapping 1 "no" overlapping
#####

theta=start #starting value
weighted=0 #weigthed version 1=yes 0=no
### eucliden likelihood #####
type_subs=3 ### type of subsampling 1=in space 2= in time 3 =spacetime
tCPU = proc.time()
res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)
tCPU = proc.time()-tCPU;tCPU
eu_par=c(res$par[1],res$par[3],res$par[4],res$par[2])
names(eu_par)=names(fit$param)
print(eu_par)

res=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)

### OpenCL eucliden likelihood #####
local <- c(1,1)
GPU <- 0
tGPU = proc.time()
res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)
tGPU = proc.time()-tGPU;tGPU
eu_par1=c(res1$par[1],res1$par[3],res1$par[4],res1$par[2])
names(eu_par1)=names(fit$param)
print(eu_par1)
tCPU;tGPU

res1=STBEUFit(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,winstp,winc_t,winstp_t,type_sub)

```

STBEUTimeEvalOcl

Time evaluation for main function in STBEU (CPU vs OpenCL).

Description

Time evaluation for main function in STBEU. Excecution times are evaluated for CPU and OpenCL depending on the OpenCL device selection. Its arguments are the same as STBEUFit The function returns CPU time evaluation, OpenCL time evaluation, its respective parameter values, the absolute time difference and which one is the fastest.

Usage

```

STBEUTimeEvalOcl(theta,fix=NULL,coords,times,cc,data,type_dist=1,maxdist=NULL,maxtime=NULL, winc=
winc_t=NULL,winstp_t=NULL,subs=NULL,weighted=FALSE,local=c(1,1),GPU=NULL)

```

Arguments

theta A vector of starting parametes for estimation

fixed	An optional named vector giving the values of the parameters that will be considered as known values. The default is NULL
coords	A numeric ($d \times 2$)-matrix (where d is the number of spatial sites).
times	A numeric vector assigning 1-dimension of temporal coordinates.
cc	String; the name of a correlation model. Values (1, 2) are allowed for Double Exponential and Gneting respectively. See the Section Details .
data	A t -matrix (a single spatial realisation) where d is the number of spatial sites and t is the number of temporal coordinates. For the description see the Section Details .
type_dist	Numeric; 1 is Euclidean and 2 is Geodesic. The default is 1, the euclidean distance. See the Section Details .
maxdist	Numeric; an optional positive value indicating the maximum spatial distance considered in the composite likelihood computation. See the Section Details for more information.
maxtime	Numeric; an optional positive value indicating the maximum temporal separation considered in the euclidean likelihood computation (see Details).
winc_s	Numeric; a positive value for computing the spatial sub-window in the sub-sampling procedure. See Details for more information.
winstp_s	Numeric; a value in $(0, 1]$ for defining the the proportion of overlapping in the spatial sub-sampling procedure. The case 1 correspond to no overlapping. See Details for more information.
winc_t	Numeric; a positive value for computing the temporal sub-window in the sub-sampling procedure. See Details for more information.
winstp_t	Numeric; a value in $(0, 1]$ for defining the the proportion of overlapping in the temporal sub-sampling procedure. The case 1 correspond to no overlapping. See Details for more information.
subs	Numeric; a value in $(1, 2, 3)$ for defining the type of sub-sampling procedure. Cases are space, time and spacetime respectively. See Details for more information.
weighted	Logical; if TRUE the likelihood objects are weighted, see the Section Details . If FALSE (the default) the euclidean likelihood is not weighted.
local	Numeric; number of local work-items of the OpenCL setup. Default is $c(1, 1)$
GPU	Numeric; if NULL (the default) no GPU computation is performed.

Details

The optimization method is Nelder-mead.

The cc corrmmodel parameter allows to select a specific correlation function for the RF. Options are: 1 for Double Exponential and 2 for Gneting.

The distance parameter allows to consider different kinds of spatial distances. The settings alternatives are:

1. Eucl, the euclidean distance (default value);
2. Geod, the geodesic distance;

The subs parameter represents the subsampling configurations. The settings alternatives are:

1. Spatial, preferred option when the number of space sites is greater than temporal coordinates;

2. Time, preferred option when the number of temporal coordinates is greater than space sites;
3. Spacetime, preferred option when the number of space and temporal are balanced.

All the nuisance and covariance parameters must be specified by the user using the `start` and the `fixed` parameter. Specifically:

The `start` parameter allows to specify (as starting values for the optimization) the parameters to be estimated. The `fixed` parameter allows to fix some of the parameters.

The `maxdist` parameter set the maximum spatial distance below which pairs of sites with inferior distances are considered in the euclidean-likelihood. This can be lower of the maximum spatial distance. **Note** that this corresponds to use a weighted composite-likelihood with binary weights. Pairs with distance less than `maxdist` have weight 1 and are included in the likelihood computation, instead those with greater distance have weight 0 and then excluded. The default is `NULL`, in this case the effective maximum spatial distance between sites is considered.

The same arguments of `maxdist` are valid for `maxtime` but here the weighed composite-likelihood regards the case of spatial-temporal field. At the moment is implemented only for Gaussian RFs. The default is `NULL`, in this case the effective maximum temporal lag between pairs of observations is considered.

The `weighted` parameter specifies if the likelihoods forming the composite-likelihood must be weighted. If `TRUE` the weights are selected by opportune procedures that improve the efficient of the maximum composite-likelihood estimator (not implemented yet). If `FALSE` the efficient improvement procedure is not used.

For computing the standard errors by the sub-sampling procedure, `winconst` and `winstp` parameters represent respectively a positive constant used to determine the sub-window size and the step with which the sub-window moves.

In the spatio-temporal case the subsampling is meant only in time as described by Li et al. (2007). Thus, `winconst` represents the length of the temporal sub-window. By default the size of the sub-window is computed following the rule established in Li et al. (2007). By default `winstp` is the time step.

Value

Returns an object of class `STBEUfit`. An object of class `STBEUfit` is a list containing at most the following components:

`param` The vector of parameters' estimates;

Author(s)

Moreno Bevilacqua, <moreno.bevilacqua@uv.cl>, <https://sites.google.com/a/uv.cl/moreno-bevilacqua/home>, Víctor Morales Oñate, <victor.morales@uv.cl>, <https://sites.google.com/site/moralesonatevictor/>

References

Composite-likelihood:

Varin, C., Reid, N. and Firth, D. (2011). An Overview of Composite Likelihood Methods. *Statistica Sinica*, **21**, 5–42.

Varin, C. and Vidoni, P. (2005) A Note on Composite Likelihood Inference and Model Selection. *Biometrika*, **92**, 519–528.

Weighted Composite-likelihood for binary RFs:

Patrick, J. H. and Subhash, R. L. (1998) A Composite Likelihood Approach to Binary Spatial Data. *Journal of the American Statistical Association, Theory & Methods*, **93**, 1099–1111.

Weighted Composite-likelihood for Gaussian RFs:

Bevilacqua, M. Gaetan, C., Mateu, J. and Porcu, E. (2012) Estimating space and space-time covariance functions for large data sets: a weighted composite likelihood approach. *Journal of the American Statistical Association, Theory & Methods*, **107**, 268–280.

Bevilacqua, M. Gaetan, C. (2013) On composite likelihood inference based on pairs for spatial Gaussian RFs *Technical Report, Department of Statistics, de Valparaiso University*.

Sub-sampling estimation:

Carlstein, E. (1986) The Use of Subseries Values for Estimating the Variance. *The Annals of Statistics*, **14**, 1171–1179.

Heagerty, P. J. and Lumley T. (2000) Window Subsampling of Estimating Functions with Application to Regression Models. *Journal of the American Statistical Association, Theory & Methods*, **95**, 197–211.

Lee, Y. D. and Lahiri S. N. (2002) Variogram Fitting by Spatial Subsampling. *Journal of the Royal Statistical Society. Series B*, **64**, 837–854.

Li, B., Genton, M. G. and Sherman, M. (2007). A nonparametric assessment of properties of space-time covariance functions. *Journal of the American Statistical Association*, **102**, 736–744

Examples

```
# "model=1" is Double Exponential and "model=2" is Gneiting
rm(list=ls())
require(GeoModels)
require(MCMCpack)
require(STBEU)

#####location sites #####
lambda=8
xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))   ###regular
nrow(coords)
# plot(coords)
#set.seed(15)                                     ### not regular
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))   ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
#####temporal instants #####
times=seq(1,5,1)

type_dist=1          ### type of distance      1=euclidean 2=chordal 3=geodesic
maxdist=2
maxtime=2

#####
model=1 # 1=double exponential      2 =gneiting

if (model == 1) {
  # exponential model
  cov.model <-"exp_exp"
  cc=1
  #####
  mean=0
  nugget=0
  scale_s<-3/3
```

```

scale_t<-3/3
sill=1
param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill)
fixed=list(nugget=0)
fix=c(nugget=nugget)
}
#####
set.seed(276)
#####
# Simulation of the Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
#####
mm=mean(c(data))
vv=var(c(data))

#####
# parameters for the subsampling ###
#####
coordx=coords[,1]
coordy=coords[,2]
LX=abs(range(coordx)[1]-range(coordx)[2])
LY=abs(range(coordy)[1]-range(coordy)[2])
lato_fin=3 #changing window size
lx=lato_fin #lunghezza lato x quadrato subfinestra
ly=lato_fin #lunghezza lato y quadrato subfinestra
winc=c(lx/sqrt(LX),ly/sqrt(LY))
winstp= 1### 1/lato_fin complete overlapping in space 1 "no" overlapping in space
#####
winc_t=4 ### length of temporal window
winstp_t=1 ### 0.5 half overlapping 1 "no" overlapping
#####
theta=c(mean = mm,sill = vv,scale_s = scale_s,scale_t = scale_t) #starting value
weighted=0 #weigthed version 1=yes 0=no
### eucliden likelihood #####
type_subs=1 ### type of subsampling 1=in space 2= in time 3 =spacetime
tCPU = proc.time()
GPU = 0;local = c(1,1)
STBEUTimeEvalOcl(theta,fix,coords,times,cc,data,type_dist,maxdist ,
maxtime,winc,winstp,winc_t,winstp_t,type_subs,weighted,GPU = GPU, local = local)

```

SubSampTimeEval

Time evaluation for main function in STBEU (CPU vs OpenCL) looping over blocks

Description

Time evaluation for main function in STBEU looping over blocks. Excecution times are evaluated for CPU and OpenCL depending on the OpenCL device selection. Its arguments are the same as STBEUFit The function returns CPU time evaluation, OpenCL time evaluation, its respective parameter values, the absolute time difference and which one is the fastest.

Usage

```
STBEUTimeEvalOcl(theta,fix=NULL,coords,times,cc,data,type_dist=1,maxdist=NULL,maxtime=NULL, winc_s=
winc_t=NULL,winstp_t=NULL,subs=NULL,weighted=FALSE,local=c(1,1),GPU=NULL)
```

Arguments

theta	A vector of starting parametes for estimation
fixed	An optional named vector giving the values of the parameters that will be considered as known values. The default is NULL
coords	A numeric ($d \times 2$)-matrix (where d is the number of spatial sites).
times	A numeric vector assigning 1-dimension of temporal coordinates.
cc	String; the name of a correlation model. Values (1, 2) are allowed for Double Exponential and Gneting respectively. See the Section Details .
data	A t -matrix (a single spatial realisation) where d is the number of spatial sites and t is the number of temporal coordinates. For the description see the Section Details .
type_dist	Numerical; 1 is Euclidean and 2 is Geodesic. The default is 1, the euclidean distance. See the Section Details .
maxdist	Numeric; an optional positive value indicating the maximum spatial distance considered in the composite likelihood computation. See the Section Details for more information.
maxtime	Numeric; an optional positive value indicating the maximum temporal separation considered in the euclidean likelihood computation (see Details).
winc_s	Numeric; a positive value for computing the spatial sub-window in the sub-sampling procedure. See Details for more information.
winstp_s	Numeric; a value in $(0, 1]$ for defining the the proportion of overlapping in the spatial sub-sampling procedure. The case 1 correspond to no overlapping. See Details for more information.
winc_t	Numeric; a positive value for computing the temporal sub-window in the sub-sampling procedure. See Details for more information.
winstp_t	Numeric; a value in $(0, 1]$ for defining the the proportion of overlapping in the temporal sub-sampling procedure. The case 1 correspond to no overlapping. See Details for more information.
subs	Numeric; a value in $(1, 2, 3)$ for defining the type of sub-sampling procedure. Cases are space, time and spacetime respectively. See Details for more information.
weighted	Logical; if TRUE the likelihood objects are weighted, see the Section Details . If FALSE (the default) the euclidean likelihood is not weighted.
local	Numeric; number of local work-items of the OpenCL setup. Default is $c(1, 1)$
GPU	Numeric; if NULL (the default) no GPU computation is performed.

Details

The optimization method is Nelder-mead.

The cc corrmmodel parameter allows to select a specific correlation function for the RF. Options are: 1 for Double Exponential and 2 for Gneting.

The distance parameter allows to consider differents kinds of spatial distances. The settings alternatives are:

1. Eucl, the euclidean distance (default value);
2. Geod, the geodesic distance;

The subs parameter represents the subsampling configurations. The settings alternatives are:

1. Spatial, preferred option when the number of space sites is greater than temporal coordinates;
2. Time, preferred option when the number of temporal coordinates is greater than space sites;
3. Spacetime, preferred option when the number of space and temporal are balanced.

All the nuisance and covariance parameters must be specified by the user using the start and the fixed parameter. Specifically:

The start parameter allows to specify (as starting values for the optimization) the parameters to be estimated. The fixed parameter allows to fix some of the parameters.

The maxdist parameter set the maximum spatial distance below which pairs of sites with inferior distances are considered in the euclidean-likelihood. This can be lower of the maximum spatial distance. **Note** that this corresponds to use a weighted composite-likelihood with binary weights. Pairs with distance less than maxdist have weight 1 and are included in the likelihood computation, instead those with greater distance have weight 0 and then excluded. The default is NULL, in this case the effective maximum spatial distance between sites is considered.

The same arguments of maxdist are valid for maxtime but here the weighed composite-likelihood regards the case of spatial-temporal field. At the moment is implemented only for Gaussian RFs. The default is NULL, in this case the effective maximum temporal lag between pairs of observations is considered.

The weighted parameter specifies if the likelihoods forming the composite-likelihood must be weighted. If TRUE the weights are selected by opportune procedures that improve the efficient of the maximum composite-likelihood estimator (not implemented yet). If FALSE the efficient improvement procedure is not used.

For computing the standard errors by the sub-sampling procedure, winconst and winstp parameters represent respectively a positive constant used to determine the sub-window size and the step with which the sub-window moves.

In the spatio-temporal case the subsampling is meant only in time as described by Li et al. (2007). Thus, winconst represents the length of the temporal sub-window. By default the size of the sub-window is computed following the rule established in Li et al. (2007). By default winstp is the time step.

Value

Returns an object of class STBEUFit. An object of class STBEUFit is a list containing at most the following components:

param The vector of parameters' estimates;

Author(s)

Moreno Bevilacqua, <moreno.bevilacqua@uv.cl>, <https://sites.google.com/a/uv.cl/moreno-bevilacqua/home>, Víctor Morales Oñate, <victor.morales@uv.cl>, <https://sites.google.com/site/moralesonatevictor/>

References

Composite-likelihood:

Varin, C., Reid, N. and Firth, D. (2011). An Overview of Composite Likelihood Methods. *Statistica Sinica*, **21**, 5–42.

Varin, C. and Vidoni, P. (2005) A Note on Composite Likelihood Inference and Model Selection. *Biometrika*, **92**, 519–528.

Weighted Composite-likelihood for binary RFs:

Patrick, J. H. and Subhash, R. L. (1998) A Composite Likelihood Approach to Binary Spatial Data. *Journal of the American Statistical Association, Theory & Methods*, **93**, 1099–1111.

Weighted Composite-likelihood for Gaussian RFs:

Bevilacqua, M. Gaetan, C., Mateu, J. and Porcu, E. (2012) Estimating space and space-time covariance functions for large data sets: a weighted composite likelihood approach. *Journal of the American Statistical Association, Theory & Methods*, **107**, 268–280.

Bevilacqua, M. Gaetan, C. (2013) On composite likelihood inference based on pairs for spatial Gaussian RFs *Technical Report, Department of Statistics, de Valparaiso University*.

Sub-sampling estimation:

Carlstein, E. (1986) The Use of Subseries Values for Estimating the Variance. *The Annals of Statistics*, **14**, 1171–1179.

Heagerty, P. J. and Lumley T. (2000) Window Subsampling of Estimating Functions with Application to Regression Models. *Journal of the American Statistical Association, Theory & Methods*, **95**, 197–211.

Lee, Y. D. and Lahiri S. N. (2002) Variogram Fitting by Spatial Subsampling. *Journal of the Royal Statistical Society. Series B*, **64**, 837–854.

Li, B., Genton, M. G. and Sherman, M. (2007). A nonparametric assessment of properties of space-time covariance functions. *Journal of the American Statistical Association*, **102**, 736–744

Examples

```
# "model=1" is Double Exponential and "model=2" is Gneiting
rm(list=ls())
require(GeoModels)
require(MCMCpack)
require(STBEU)

####location sites #####
lambda=8
xx=seq(-lambda,lambda);
coords=as.matrix(expand.grid(xx,xx))   ###regular
nrow(coords)
# plot(coords)
#set.seed(15)                                     ### not regular
#pp<-runifpoint(4*(lambda)^2, win=owin(c(-lambda,lambda),c(-lambda,lambda)))   ### not regular
#coords<-cbind(pp$x,pp$y)
### not regular
####temporal instants #####
times=seq(1,5,1)

type_dist=1          ### type of distance      1=euclidean 2=chordal 3=geodesic
maxdist=2
maxtime=2
```

```
#####
model=1 # 1=double exponential      2 =gneiting

if (model == 1) {
  # exponential model
  cov.model <- "exp_exp"
  cc=1
  #####
  mean=0
  nugget=0
  scale_s<-3/3
  scale_t<-3/3
  sill=1
  param=list(nugget=nugget,mean=mean,scale_t=scale_t,scale_s=scale_s,sill=sill)
  fixed=list(nugget=0)
  fix=c(nugget=nugget)
}

#####
set.seed(276)
#####
# Simulation of the Gaussian random field:
data <- GeoSim(coordx=coords,coordt=times,corrmodel=cov.model, param=param)$data
#####
mm=mean(c(data))
vv=var(c(data))

#####
# parameters for the subsampling ###
#####
coordx=coords[,1]
coordy=coords[,2]
LX=abs(range(coordx)[1]-range(coordx)[2])
LY=abs(range(coordy)[1]-range(coordy)[2])
lato_fin=3 #changing window size
lx=lato_fin      #lunghezza lato x quadrato subfinestra
ly=lato_fin      #lunghezza lato y quadrato subfinestra
winc=c(lx/sqrt(LX),ly/sqrt(LY))
winstp= 1### 1/lato_fin complete overlapping in space 1 "no" overlapping in space
#####
winc_t=4  ### length of temporal window
winstp_t=1  ### 0.5 half overlapping 1 "no" overlapping
#####
theta=c(mean = mm,sill = vv,scale_s = scale_s,scale_t =scale_t)      #starting value
weighted=0      #weigthed version 1=yes 0=no
### eucliden likelihood #####
type_subs=1  ### type of subsampling 1=in space 2= in time 3 =spacetime
tCPU = proc.time()
GPU = 0;local = c(1,1)
SubSampTimeEval(theta,fix,coords,times,cc,data,type_dist,maxdist ,maxtime,winc,
                winstp,winc_t,winstp_t,type_subs,weighted,GPU = GPU, local = local)
```


Index

*Topic **Composite**

STBEUFit, [2](#)

STBEUTimeEvalOcl, [16](#)

SubSampTimeEval, [20](#)

*Topic **Devices**

DevOpenCL, [1](#)

DevOpenCL, [1](#)

STBEUFit, [2](#)

STBEUTimeEvalOcl, [16](#)

SubSampTimeEval, [20](#)