Package ‘spatialnbda’

February 20, 2015

Type Package

Title Performs spatial NBDA in a Bayesian context

Version 1.0

Date 2014-09-16

Author Glenna Nightingale

Maintainer Glenna Nightingale <glenna.evans@gmail.com>

Description Network based diffusion analysis (NBDA) allows inference on the asocial and social transmission of information. This may involve the social transmission of a particular behaviour such as tool use, for example. For the NBDA, the key parameters estimated are the social effect and baseline rate parameters. The baseline rate parameter gives the rate at which the behaviour is first performed (or acquired) asocially amongst the individuals in a given population. The social effect parameter quantifies the effect of the social associations amongst the individuals on the rate at which each individual first performs or displays the behaviour. Spatial NBDA involves incorporating spatial information in the analysis. This is done by incorporating social networks derived from spatial point patterns (of the home bases of the individuals under study). In addition, a spatial covariate such as vegetation cover, or slope may be included in the modelling process.

LazyData true

Depends SocialNetworks(>= 1.1), mvtnorm(>= 0.9)

License GPL

NeedsCompilation no

Repository CRAN

Date/Publication 2014-09-19 00:49:05

R topics documented:

`FormatData` *Formats the data for NBDA*

Description

Formats the data for NBDA

Usage

```
FormatData(a, b, c, d, e, f, g,h,networkdata)
```

Arguments

`a`	diffusion times
`b`	social network
`c`	unique id for individual at that particular event
`d`	group number
`e`	diffusion number
`f`	event number
`g`	alternative social network
`h`	spatial covariate
`networkdata`	list of social networks

idarray *Individual id's for RJMCMC Example 1.*

Description

These data consist of unique id's for individuals recorded in ten diffusions.

Ids *This dataset contains the unique id for each individual in each group. This is part of the data example to demonstrate modelling NBDA with random effects.*

Description

The ids are the unique id for each individual that was observed to first solve a task or demonstrate a new behaviour.

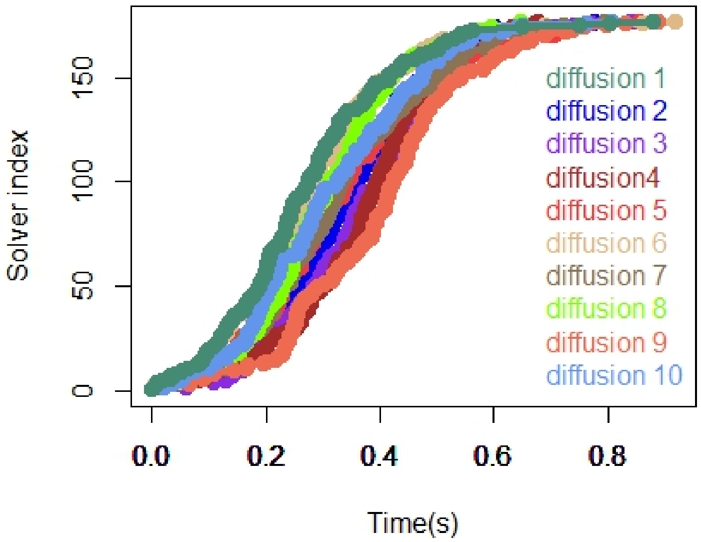
Usage

```
data(Ids)
```

mcmc *Performs spatial NBDA in a Bayesian context*

Description

The model considered is the full model which contains two parameters: the baseline rate parameter, and the social parameter. The hazard function used therefore contains two basic components: lambda0, the baseline rate parameter and the social parameter, s'. The hazard function for individual i at time t is expressed as: lambda(it) = lambda0 + (s'*Sum Aij_j)* z_j(t). In the hazard function, Sum Aij_j represents the sum of the interactions of individuals j on individual i. The term z_j(t) =1, if individual j has previously displayed the behaviour under study by time t, and zero otherwise. The example provided describes the analysis of the times shown in the figure above:



Usage

```
mcmc(formatteddata, its, pilot_tuner1, pilot_tuner2, start1, start2)
```

Arguments

formatteddata	data formatted using the function FormatData
its	number of iterations
pilot_tuner1	tuning parameter for the social effect
pilot_tuner2	tuning parameter for the asocial effect
start1	start value for the social parameter
start2	start value for the asocial parameter

Examples

```
# library(calibrate)
# loading the x and y spatial coordinates to construct the spatially derived
#social network

data(Xx)
data(Yy)

X <- cbind(Xx,Yy)
```

```
plot(X[,1],X[,2],pch=16,cex=1,xlim=c(0,1),ylim=c(0,1),xlab="x",
ylab="y",main="",cex.axis=2,cex.lab=2)

areas = calculate.areas(X[,1], X[,2], rep(0.2,length(X[,1])), 1000)
spatialareas = areas
len = length(X[,1])

spatialnetwork = matrix(0,nrow=len,ncol=len)
for(i in 1:len){
  for(j in i:len){
    template = spatialareas[[i]][j]
    spatialnetwork[i,j] = spatialnetwork[j,i] = template
    #spatialareas[[i]]=NULL

  }

}

# loading the times and ids to plot the diffusion times and run nbda

data(Times)
data(Ids)
numdiff = 10

plot_colors = colors()[c(12,28,31,32,34,37,41,47,59,62,146,176,258,
117,154,625,563,376,113,556)]

for(i in 1:numdiff){
  a = (i-1) * (len)
  b = a + (len)
  startindex = a + 1
  endindex = b
 plot(Times[startindex:endindex,1],c(1:len),type="o",lwd=4,col=plot_colors[i],ylab="Solver index",
  main="",xlab="Time(s)",yaxt='n',ylim=c(0,len),xlim=range(Times))
 #textxy(c(1:len), Times[startindex:endindex,1], Ids[startindex:endindex,1],cex = .8,col="red")
  par(new=TRUE)
}
par(new=TRUE)
plot( Times[1:len,1],c(1:len),type="o",lwd=4,col=plot_colors[1],ylab="",main="",xlab="",
     ylim=c(0,len),xlim=range(Times))



Diffusions = rep(1,len)
for(i in 2:numdiff){
  addon = rep(i,len)
  Diffusions = c(Diffusions,addon)

}
Groups = rep(1,length(Times[,1]))
Events = c(1:length(Times[,1]))
space = rep(1,length(Times[,1]))
spatialnetwork = 1*spatialnetwork
```

```
shape = FormatData(Times[,1],spatialnetwork,Ids[,1],Groups,Diffusions,Events,spatialnetwork)

# running nbda to obtain posterior estimates of the social and
# baseline rate parameters

#ptm <- proc.time()
#mcmc(shape,10000,0.05,0.05,-3,-5)
#proc.time() - ptm
```

mcmcre *Performs NBDA with individual level random effects*

Description

The incorporation of random effects accounts for heterogeneity contributed by individual aptitudes of the individuals concerned. The baseline rate is then scaled by the random effects.

Usage

```
mcmcre(formatteddata, its, pilot_tuner1, pilot_tuner2, start1, start2)
```

Arguments

`formatteddata`	Formatted data using the function FormatData
`its`	Number of iterations
`pilot_tuner1`	Tuner for the social parameter
`pilot_tuner2`	Tuner for the asocial parameter
`start1`	Start value for the social parameter
`start2`	Start value for the asocial parameter

Value

The output is a list that contains: (i) The siumulated values for each parameter (ii) The posterior summaries each random effect parameter, (iii) The posterior summaries for the social and asocial parameters Trace plots for the social and asocial parameters are provided together with a density and acf plot for the social parameter.

Examples

```
# library(calibrate)
# loading the x and y spatial coordinates to construct the spatially derived
#social network
```

```
data(Xx)
data(Yy)

X <- cbind(Xx,Yy)
plot(X[,1],X[,2],pch=16,cex=1,xlim=c(0,1),ylim=c(0,1),xlab="x",
ylab="y",main="",cex.axis=2,cex.lab=2)

areas = calculate.areas(X[,1], X[,2], rep(0.2,length(X[,1])), 1000)
spatialareas = areas
len = length(X[,1])

spatialnetwork = matrix(0,nrow=len,ncol=len)
for(i in 1:len){
  for(j in i:len){
    template = spatialareas[[i]][j]
    spatialnetwork[i,j] = spatialnetwork[j,i] = template
    #spatialareas[[i]]=NULL

  }

}

# loading the times and ids to plot the diffusion times and run nbda

data(Times)
data(Ids)
numdiff = 10

plot_colors = colors()[c(12,28,31,32,34,37,41,47,59,62,146,176,258,117,154,625,563,376,113,556)]

for(i in 1:numdiff){
  a = (i-1) * (len)
  b = a + (len)
  startindex = a + 1
  endindex = b
 plot(Times[startindex:endindex,1],c(1:len),type="o",lwd=4,col=plot_colors[i],ylab="Solver index",
  main="",xlab="Time(s)",yaxt='n',ylim=c(0,len),xlim=range(Times))
 #textxy(c(1:len), Times[startindex:endindex,1], Ids[startindex:endindex,1],cex = .8,col="red")
  par(new=TRUE)
}
par(new=TRUE)
plot( Times[1:len,1],c(1:len),type="o",lwd=4,col=plot_colors[1],ylab="",main="",xlab="",
     ylim=c(0,len),xlim=range(Times))



Diffusions = rep(1,len)
for(i in 2:numdiff){
  addon = rep(i,len)
  Diffusions = c(Diffusions,addon)

}
Groups = rep(1,length(Times[,1]))
```

```
Events = c(1:length(Times[,1]))
space = rep(1,length(Times[,1]))
spatialnetwork = 1*spatialnetwork
shape = FormatData(Times[,1],spatialnetwork,Ids[,1],Groups,Diffusions,Events,spatialnetwork)

# running nbda to obtain posterior estimates of the social and
# baseline rate parameters

#ptm <- proc.time()
#mcmc(shape,10000,0.05,0.05,-3,-5)
#proc.time() - ptm


# running nbda to obtain posterior estimates of the
# social, baseline rate and random effect parameters

#ptm <- proc.time()
#mcmcre(shape,10000,0.05,0.05,-3,-5)
#proc.time() - ptm
```

nullmcmc — *The spatial NBDA null model is considered for this analysis. The baseline rate parameter is estimated.*

Description

The model considered contains only one parameter: the baseline rate parameter, and represents the rate at which the behaviour under consideration is acquired.

Usage

```
nullmcmc(formatteddata, its, pilot_tuner, start)
```

Arguments

`formatteddata`	Formatted data using the FormatData function.
`its`	Number of iterations.
`pilot_tuner`	Tuning parameter
`start`	Start value.

papernests *papernests*

Description

This dataset provides the unique id of each individual nest in the order of the first observation of the behaviour under consideration. This is for the spatial NBDA example with an environmental covariate..

Usage

```
data(papernests)
```

Format

A data frame with 70 observations.

`Nests` a numeric vector

Examples

```
data(papernests)
```

papertimes *papertimes*

Description

This dataset provides the diffusion times for the dataset used for spatial NBDA example with an environmental covariate.. The values represent the times at which the behaviour under consideration was first observed at a given nest.

Usage

```
data(papertimes)
```

Format

A data frame with 70 observations.

`Times` a numeric vector

Examples

```
data(papertimes)
## maybe str(papertimes) ; plot(papertimes) ...
```

rjmcmc *Model discriminiation in a Bayesian context for spatial NBDA.*

Description

An RJMCMC algorithm is used to achieve model discrimination between the null model which contains only the baseline rate parameter and the full model which contains both the baseline rate and social parameters.

Usage

```
rjmcmc(formatteddata, its, pilot_tuner1, pilot_tuner2, start1, start2, start3, p1, p2)
```

Arguments

formatteddata	Formatted data using the FormatData function.
its	Number of iterations
pilot_tuner1	Tuner for proposal distribution for the social parameter.
pilot_tuner2	Tuner for the proposal distribution for the baseline rate parameter.
start1	Start value for the social parameter
start2	Start value for the baseline rate parameter
start3	Start model
p1	Unifor prior variance tuner for the baseline rate
p2	Uniform prior variance tuner for the social parameter

Details

It is important to check that the chains have mixed which using this function. A rough way would be to view the trace plots printed.

Value

The output is a table with the number of iterations for which the Markov chain spent in each visited model.

Examples

```
#Example 1
data(timearray)
data(idarray)
data(socialx)
data(socialy)

Times = timearray[,1]
Ids = idarray[,1]
```

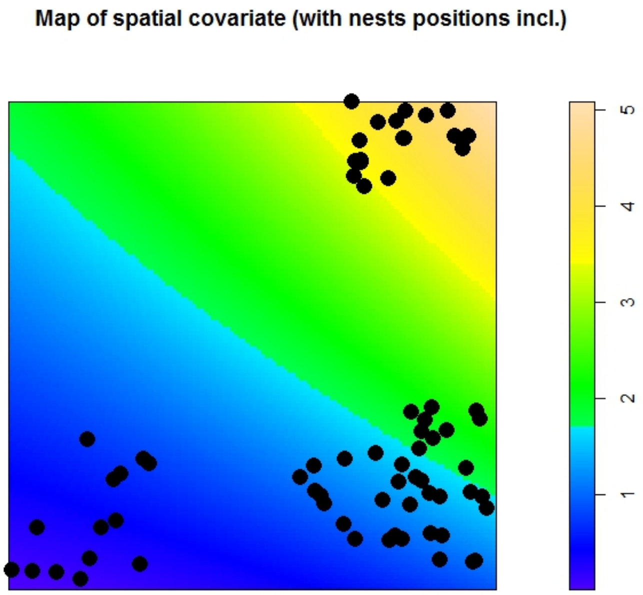
```
lenh = length(Times)
Groups = rep(1,length(Times))
Events = c(1:length(Times))

socialites = matrix(1,nrow=lenh,ncol=lenh)
x = socialx
y = socialy

plot(x[,1],y[,1],xlab=”x”,ylab=”y”,cex=2,pch=16,main=”Point pattern of nest positions”)

areas = calculate.areas(x[,1],y[,1],rep(0.2,lenh),1000)
spatialareas = areas
len = length(x[,1])
Diffusions = rep(1,len)
for(i in 2:10){
  addon = rep(i,len)
  Diffusions = c(Diffusions,addon)

}


spatialnetwork = matrix(0,nrow=len,ncol=len)
for(i in 1:len){
  for(j in i:len){
    template = spatialareas[[i]][j]
    spatialnetwork[i,j] = spatialnetwork[j,i] = template
    #spatialareas[[i]]=NULL

  }

}


shape = FormatData(Times,spatialnetwork,Ids,Groups,Diffusions,Events,spatialnetwork)



#ptm <- proc.time()
#mcmc(shape,10000,0.05,0.05,-3,-5)
#proc.time() - ptm

#ptm <- proc.time()
#rjmcmc(shape,10000,5,1,-3,-3,1,10,10)
#proc.time() - ptm


# Example 2
data(papertimes)
data(papernests)
data(x)
data(y)
z = array(0,c(length(x[,1]),1))# setting up array for storing spatial covariate information
```

```
for(i in 1:70){   # simulating spatial covariate information
xx = x[,1][i]
yy = y[,1][i]
z[i] = (3*xx + 14*yy) * exp(2 * (.4*xx - 1))
}



Times = papertimes[,1]
Ids = papernests[,1]
Diffusions = rep(1,length(Times))
Groups = rep(1,length(Times))
Events = c(1:length(Times))
socialites = matrix(1,nrow=70,ncol=70)


plot(x[,1],y[,1],xlab="x",ylab="y",cex=2,pch=16,main="Point pattern of nest positions")



areas = calculate.areas(x[,1],y[,1],rep(0.05,70),1000)
spatialareas = areas
len = length(x[,1])

spatialnetwork = matrix(0,nrow=len,ncol=len)
for(i in 1:len){
  for(j in i:len){
    template = spatialareas[[i]][j]
    spatialnetwork[i,j] = spatialnetwork[j,i] = template
    #spatialareas[[i]]=NULL

  }

}


shape = FormatData(Times,spatialnetwork,Ids,Groups,Diffusions,Events,spatialnetwork,z)


#ptm <- proc.time()
#mcmc(shape,10000,5,1,-5,-6)
#proc.time() - ptm

#ptm <- proc.time()
#nullmcmc(shape,10000,1,-5)
#proc.time() - ptm


#ptm <- proc.time()
#rjmcmc(shape,10000,5,1,0,0,2,5,5)
#proc.time() - ptm
```

`smcmc` *Performs spatial NBDA in a Bayesian context with an environmental covariate*

Description

Performs spatial network based diffusion analysis in a Bayesian context. This analysis includes values of an environmental covariate in the modelling process. The figure below depicts a point pattern formed by nest/home-range locations superimposed over a plot of the environmental covariate expressed as an image. This type of data can be analysed by this function (given the accompanying diffusion times and id's). The example dataset is analysed using this function (in addition to mcmc)



Usage

```
smcmc(formatteddata, its, pilot_tuner1, pilot_tuner2,
pilot_tuner3, start1, start2, start3)
```

Arguments

`formatteddata`	Formatted data using the function FormatData
`its`	Number of iterations

pilot_tuner1	pilot tuner for the social parameter
pilot_tuner2	pilot tuner for the baseline parameter
pilot_tuner3	pilot tuner for the spatial/environmental parameter
start1	start value for the social parameter
start2	start value for the baseline rate parameter
start3	start value for the spatial/environmental parameter

Value

The output is a list that contains: (i) The posterior simulated values for each parameter, and (ii) The posterior summaries for each parameter Trace plots for the social and asocial parameters are provided together with a density and acf plot for the social parameter.

Author(s)

Glenna Nightingale

Examples

```
library(SocialNetworks)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (formatteddata, its, pilot_tuner1, pilot_tuner2, pilot_tuner3,
    start1, start2, start3)
{
    TimeD = formatteddata[[1]][, 7]
    censored = formatteddata[[1]][, 6]
    Aij = formatteddata[[1]][, 8]
    NaiveD = formatteddata[[2]]
    spatcov = formatteddata[[1]][, 9]
    s0 = start1
    baseline_rate = lambda0 = start2
    environmental = beta0 = start3
    acceptcounter = 0
    Jumbo <- array(0, c(its, 3))
    newparam <- array(0.5, 3)
    CurrentParam <- array(0.5, 3)
    newparam[1] = (CurrentParam[1] <- s0)
    newparam[2] = (CurrentParam[2] <- lambda0)
    newparam[3] = (CurrentParam[3] <- beta0)
    updates <- function(CurrentParam, newparam) {
        GU3 <- runif(1, CurrentParam[1] - pilot_tuner1, CurrentParam[1] +
            pilot_tuner1)
        proposal = c(GU3, CurrentParam[2:3])
        num <- CpS(proposal)[[1]]
        den <- CpS(CurrentParam)[[1]]
        acc <- exp(num - den)
        acceptr <- min(1, acc)
        r <- runif(1)
        newparam[1] <- ifelse((r <= acceptr), GU3, CurrentParam[1])
        return(newparam[1])
    }
    updatelambda <- function(CurrentParam, newparam) {
        GU3 <- runif(1, CurrentParam[2] - pilot_tuner2, CurrentParam[2] +
            pilot_tuner2)
        proposal = c(CurrentParam[1], GU3, CurrentParam[3])
        num <- CpS(proposal)[[1]]
        den <- CpS(CurrentParam)[[1]]
        acc <- exp(num - den)
        acceptt <- min(1, acc)
        r <- runif(1)
        newparam[2] <- ifelse((r <= acceptt), GU3, CurrentParam[2])
        acceptcounter <- ifelse((r <= acceptt), 1, 0)
        list(newparam[2], acceptcounter)
    }
    updatecovariate <- function(CurrentParam, newparam) {
        GU3 <- runif(1, CurrentParam[3] - 5, CurrentParam[3] +
            5)
        proposal = c(CurrentParam[1:2], GU3)
        num <- CpS(proposal)[[1]]
        den <- CpS(CurrentParam)[[1]]
        acc <- exp(num - den)
        acceptr <- min(1, acc)
        r <- runif(1)
        newparam[3] <- ifelse((r <= acceptr), GU3, CurrentParam[3])
        return(newparam[3])
    }
    CpS = function(parameterproposal) {
        baseline = exp(parameterproposal[2])
        social_rate = exp(parameterproposal[1])
        spatialC = exp(parameterproposal[2])
        hazard = baseline * exp(spatialC) + (social_rate) * Aij
        uncensored = 1 - censored
        log_likelihood_u = sum(log(hazard * exp(-hazard * TimeD)) *
            uncensored) + sum(-hazard * TimeD * NaiveD)
        log_likelihood_c = sum(-hazard * censored)
        log_likelihood = log_likelihood_u + log_likelihood_c
        lambdaprior <- log(dunif(parameterproposal[2], -10, 10))
        sprior <- log(dunif(parameterproposal[1], -10, 10))
        sCprior <- log(dunif(parameterproposal[3], -10, 10))
        pzoid <- log_likelihood + lambdaprior + sprior + sCprior
        pzoid
    }
    for (t in 1:its) {
        CurrentParam[1] = Jumbo[t, 1] = updates(CurrentParam,
            newparam)[[1]]
        CurrentParam[2] = Jumbo[t, 2] = updatelambda(CurrentParam,
            newparam)[[1]]
        CurrentParam[3] = Jumbo[t, 3] = updatecovariate(CurrentParam,
            newparam)[[1]]
    }
    burnin = its/10
    par(mfrow = c(2, 2))
    plot(Jumbo[burnin:its, 1], type = ”l”, col = ”blue”, ylab = ”social effect”,
        main = ”Trace plot for social effect, s' ”, lwd = 2)
    plot(Jumbo[burnin:its, 2], type = ”l”, col = ”red”, ylab = ”asocial effect”,
        main = ”Trace plot for asocial effect, lambda0' ”, lwd = 2)
    plot(Jumbo[burnin:its, 3], type = ”l”, col = ”lightgoldenrod”,
        ylab = ”asocial effect”, main = ”Trace plot for spatial effect, beta0' ”,
        lwd = 2)
    params = c(mean(Jumbo[burnin:its, 1]), mean(Jumbo[burnin:its,
        2]), mean(Jumbo[burnin:its, 3]))
    creds = c(sd(Jumbo[burnin:its, 1]), sd(Jumbo[burnin:its,
        2]), sd(Jumbo[burnin:its, 3]))
    mcmcresults = list(Jumbo, params, creds)
    mcmcresults
  }

#---------------------------------------------------------------
#  Run spatial NBDA to estimate the social and asocial parameters
#  s and lambda.
#  The associations for the social network in this example are calculated
#  using an interaction function that assumes each individual has
#  an area of interaction or zone of influence.
#---------------------------------------------------------------




data(papertimes)
data(papernests)
data(x)
data(y)
z = array(0,c(length(x[,1]),1))# setting up array for storing spatial covariate information

for(i in 1:70){   # simulating spatial covariate information
xx = x[,1][i]
yy = y[,1][i]
z[i] = (3*xx + 14*yy) * exp(2 * (.4*xx - 1))
}



Times = papertimes[,1]
Ids = papernests[,1]
Diffusions = rep(1,length(Times))
Groups = rep(1,length(Times))
Events = c(1:length(Times))
socialites = matrix(1,nrow=70,ncol=70)


plot(x[,1],y[,1],xlab=”x”,ylab=”y”,cex=2,pch=16,main=”Point pattern of nest positions”)
```

```
areas = calculate.areas(x[,1],y[,1],rep(0.05,70),1000)
spatialareas = areas
len = length(x[,1])

spatialnetwork = matrix(0,nrow=len,ncol=len)
for(i in 1:len){
  for(j in i:len){
    template = spatialareas[[i]][j]
    spatialnetwork[i,j] = spatialnetwork[j,i] = template
    #spatialareas[[i]]=NULL

  }

}


shape = FormatData(Times,spatialnetwork,Ids,Groups,Diffusions,Events,spatialnetwork,z)
ptm <- proc.time()
smcmc(shape,10000,5,1,1,-5,-6,-5)
proc.time() - ptm
```

socialx *x coordinates for RJMCMC Example 1.*

Description

These coordinates are combined with the y coordinates to form a spatial point pattern.

socialy *y coordinates for RJMCMC Example 1*

Description

These coordinates are combined with the x coordinates to form a spatial point pattern.

`spatialnbda` *Performs spatial NBDA in a Bayesian context.*

Description

Network based diffusion analysis (NBDA) is conducted using a spatially derived social network. The modelling process may also include an environmental covariate such as vegetation cover or slope.

Details

Package:	spatba
Type:	Package
Version:	1.0
Date:	2014-09-16
License:	GPL

Author(s)

Maintainer: Glenna Nightingale <glenna.evans@gmail.com>

References

Bradbury, J. W., and S.L. Vehrencamp (1998). Principles of animal communication. Sinauer Associates.

Croft, D.P., James, R., and Krause, J. (2008). Exploring animal social networks. USA: Princeton University Press.

Funwi-Gabga, N., and Mateu, J. (2012). Understanding the nesting spatial behaviour of gorillas in the Kagwene Sanctuary, Cameroon. Stochastic Environmental Research and Risk Assessment, vol 26, pp. 793-811.

Hoppitt, W. and Laland, K. N. (2013). Social Learning: An Introduction to Mechanisms, Methods, and Models. Princeton University Press.

Examples

```
#library(SocialNetworks)


#------------------------------
#  calculating associations
#-------------------------------
```

```
# For a regular spatial point pattern with interaction radius = 0.06
# using an interaction function that uses pairwise Euclidean distances.

x = c(0.1023117, 0.1119260, 0.1625270, 0.3594291, 0.4220571, 0.4606205, 0.5927459,
 0.6847543, 0.7065195, 0.7760657, 0.9827536)

y = c(0.2525266, 0.3346728, 0.5275355, 0.2447207, 0.2765606, 0.4999600, 0.5928410,
 0.8356211, 0.2506116, 0.8994760, 0.1432255)

plot(x,y)
irset = c(rep(0.06,11))
calculateassociations(x,y,irset)


# For a clustered spatial point pattern with interaction radius=0.05
# using an interaction function that uses pairwise Euclidean distances.

 x =
  c(0.77302412, 0.82946034, 0.65776305, 0.62294479, 0.58577335, 0.39332654,
  0.36893684, 0.40518735, 0.53956642, 0.56596859, 0.62802969, 0.10380876,
 0.71058751, 0.65943692, 0.88056259, 0.90567566, 0.91166684, 0.89489341,
 0.92668619, 0.01544599, 0.30499431, 0.28249059, 0.30733518, 0.73165075,
 0.17712420, 0.80869511, 0.77351717, 0.75508022, 0.79445346, 0.73134413,
 0.62448310, 0.60180882, 0.66741081, 0.45884352, 0.45282315, 0.45614636,
 0.45270694, 0.44764728, 0.53259346)


 y=
 c(0.943378357, 0.933698623, 0.123641160, 0.146773076, 0.135097659, 0.978760171,
 0.981407654, 0.937111187, 0.080617391, 0.114438404, 0.061834776, 0.370322731,
 0.036576942, 0.003974257, 0.830356964, 0.837171526, 0.884801445, 0.797794654,
 0.844312417, 0.969982888, 0.672246284, 0.692111852, 0.671098280, 0.999097233,
 0.003736065, 0.255322335, 0.282689074, 0.310793806, 0.229047375, 0.266413304,
 0.324984514, 0.279652338, 0.287134158, 0.331962948, 0.365469720, 0.343868765,
0.378876999, 0.331915785, 0.368805652)


plot(x,y)
irset = c(rep(0.05,length(x)))
calculateassociations(x,y,irset)



# For a random spatial point pattern with interaction radius=0.05
# using an interaction function that uses pairwise Euclidean distances.

  x =
 c( 0.74905296, 0.38309725, 0.98627509, 0.02242039, 0.54703348, 0.59173730,
 0.82340399, 0.18718650, 0.49200511, 0.86098261, 0.24848640, 0.15843825,
0.72875205 )
y  =
 c(0.73521480, 0.01661629, 0.51564570, 0.61856835, 0.20815448, 0.29431260,
 0.35507188, 0.18940107, 0.98721494, 0.98129752, 0.76510267, 0.43541222,
0.04601392)


plot(x,y)
irset = c(rep(0.1,length(x)))
calculateassociations(x,y,irset)


#----------------------------------------------------------------
#  Run spatial NBDA to estimate the social and asocial parameters
#  s and lambda.
#  The associations for the social network in this example are calculated
#  using an interaction function that assumes each individual has
#  an area of interaction or zone of influence.
#----------------------------------------------------------------




data(papertimes)
data(papernests)
data(x)
data(y)
z = array(0,c(length(x[,1]),1))# setting up array for storing spatial covariate information

for(i in 1:70){   # simulating spatial covariate information
xx = x[,1][i]
yy = y[,1][i]
z[i] = (3*xx + 14*yy) * exp(2 * (.4*xx - 1))
}



Times = papertimes[,1]
Ids = papernests[,1]
Diffusions = rep(1,length(Times))
Groups = rep(1,length(Times))
Events = c(1:length(Times))
socialites = matrix(1,nrow=70,ncol=70)


plot(x[,1],y[,1],xlab="x",ylab="y",cex=2,pch=16,main="Point pattern of nest positions")



areas = calculate.areas(x[,1],y[,1],rep(0.05,70),1000)
spatialareas = areas
len = length(x[,1])

spatialnetwork = matrix(0,nrow=len,ncol=len)
for(i in 1:len){
  for(j in i:len){
    template = spatialareas[[i]][j]
```

```
        spatialnetwork[i,j] = spatialnetwork[j,i] = template
        #spatialareas[[i]]=NULL

      }

    }


    shape = FormatData(Times,spatialnetwork,Ids,Groups,Diffusions,Events,spatialnetwork,z)
    ptm <- proc.time()
    mcmc(shape,10000,5,1,-5,-6)
    proc.time() - ptm
```

timearray *Diffusions times for RJMCMC Example 1*

Description

These data consist of diffusion times for ten diffusions.

Times *This dataset contains the diffusion times using in the This is part of the data example to demonstrate modelling NBDA with random effects.*

Description

The times are the times for which each individual was first observed to solve a task or demonstrate a new behaviour.

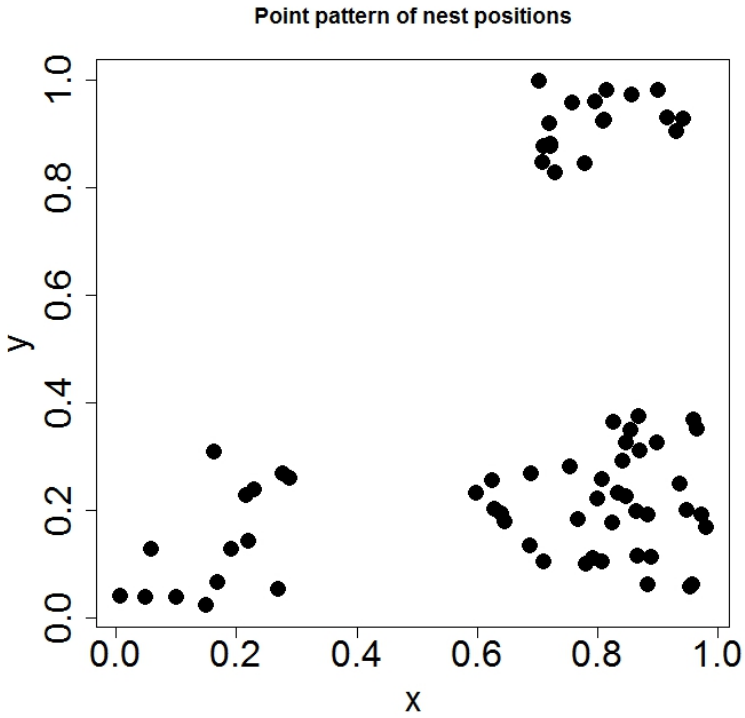
Usage

```
data(Times)
```

`x` *x*

Description

This dataset provides the x coordinates for nest positions for the data in the spatial NBDA example with an environmental covariate. These values are used to create the point pattern associated with the spatial NBDA dataset as shown below:



Usage

```
data(x)
```

Format

A data frame with 70 observations on the following variable.

`x` a numeric vector

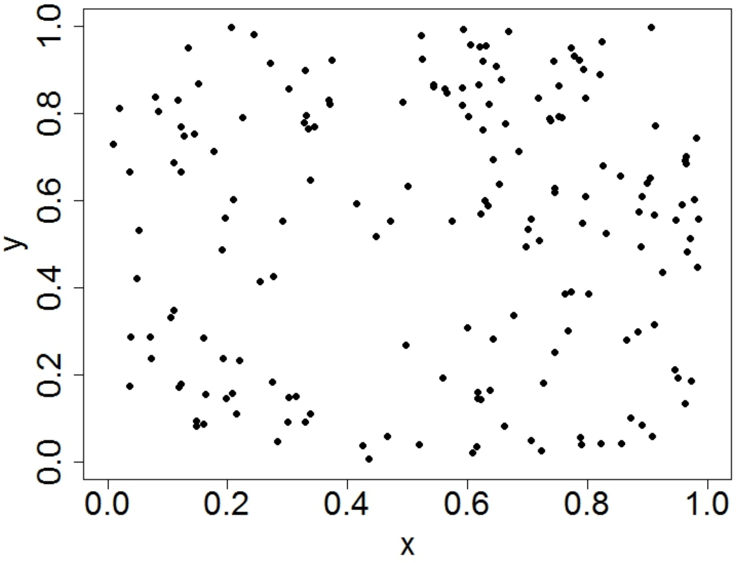
Examples

```
data(x)
## maybe str(x) ; plot(x) ...
```

Xx *x coordinates for data example for NBDA with random effects at the individual level. x and y coordinates are required to construct the spatially dervied social network.*

Description

This dataset provides the x coordinates for nest positions for the data in the spatial NBDA example where random effects are considered. These values are used to create the point pattern (as shown below) associated with the spatial NBDA dataset.



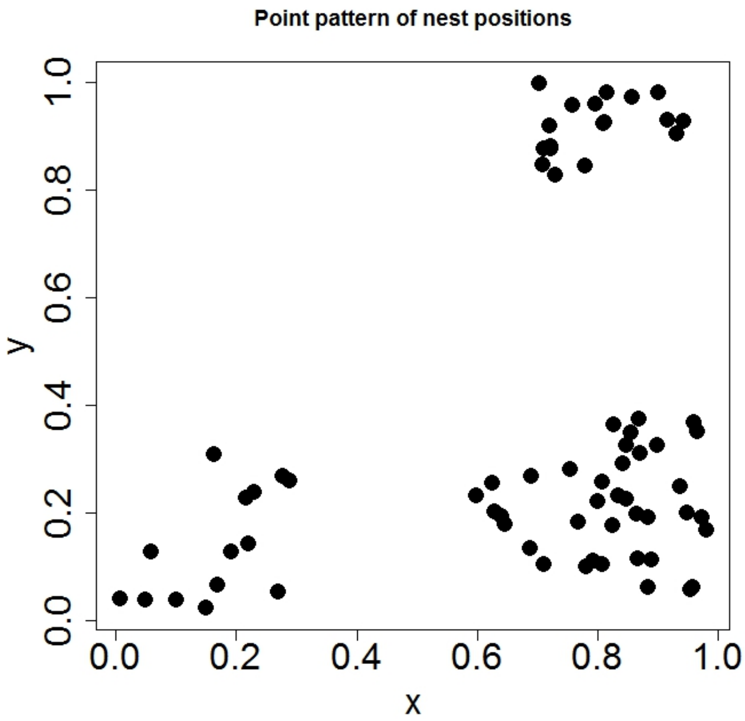
Usage

```
data(Xx)
```

y *y*

Description

This dataset provides the y coordinates for nest positions for the data in the spatial NBDA example with an environmental covariate . These values are used to create the point pattern associated with



the spatial NBDA dataset as shown below:

Usage

```
data(y)
```

Format

A data frame with 70 observations on the following variable.

`y` a numeric vector

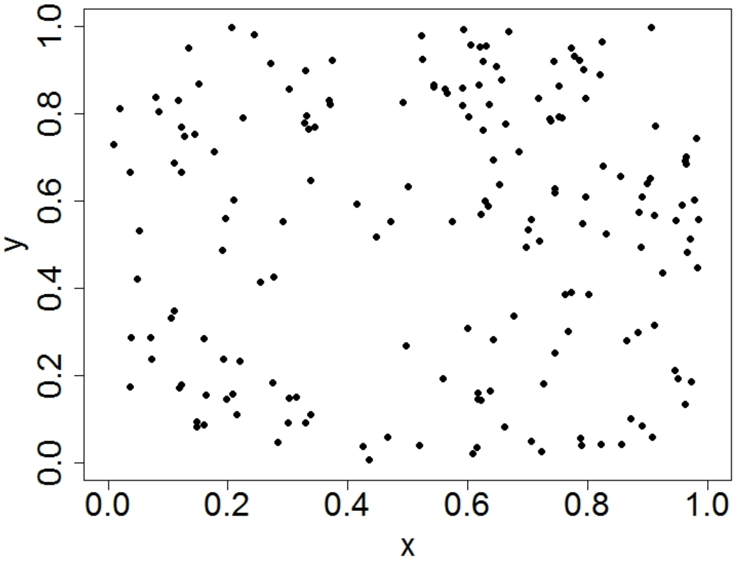
Examples

```
data(y)
```

Yy	*y coordinates for data example for NBDA with random effects at the individual level. x and y coordinates are required to construct the spatially dervied social network.*

Description

This dataset provides the x coordinates for nest positions for the data in the spatial NBDA example where random effects are considered. These values are used to create the point pattern (as shown below) associated with the spatial nbda dataset.



Usage

```
data(Yy)
```

Index