

# Package ‘stam’

February 15, 2012

**Version** 0.0-1

**Date** 2010-01-01

**Depends** R (>= 2.10.0),np,sp

**Title** Spatio-Temporal Analysis and Modelling

**Author** Zhijie Zhang <epistat@gmail.com>

**Maintainer** Zhijie Zhang <epistat@gmail.com>

**Description** stam is an evolving package that target on the various methods to conduct Spatio-Temporal Analysis and Modelling,including Exploratory Spatio-Temporal Analysis and Inferred Spatio-Temporal Modelling.

**License** GPL (>= 2)

**Repository** CRAN

**Date/Publication** 2010-02-07 19:27:03

## R topics documented:

DateConversion . . . . .	2
stkde . . . . .	3
stkde.base . . . . .	5
stkde.sig . . . . .	7
UorL . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

DateConversion	<i>Conversion between Different Date Format</i>
----------------	---

---

**Description**

DateConversion was used to read and output the date variable with your preferred format.

**Usage**

```
DateConversion(DateVar,DateIn,DateOut)
```

**Arguments**

DateVar	Specify your date variable.
DateIn	Specify the date format for your original variable,e.g.%m/%d/%Y.
DateOut	Specify the date format that you hope to output,e.g.%d/%m/%Y.

**Details**

DateConversion is an easy function to convert a date variable between different formats,which is very useful for your manipulation on a dataset with date variables without time inside.

**Value**

DateConversion returns the date format that you expected directly.

**Note**

For dates,  
More details,see above reference.

**Author(s)**

Zhijie Zhang, <epistat@gmail.com>

**References**

Spector P.*Data Manipulation with R.*,Springer Science+Business Media,LLC. 2008.(Chapter4:Dates.)

**See Also**

as.POSIX\* for Date-time Conversion Functions #as.Date,Sys.Date,POSIXct,POSIXlt #Dates for dates without times. #strptime for conversion to and from character representations. #Sys.time for clock time as a POSIXct object. #difftime for time intervals. #cut.POSIXt, seq.POSIXt, round.POSIXt and trunc.POSIXt for methods for these classes. #weekdays.POSIXt for convenience extraction functions

**Examples**

```
## Not run:
#Example 1
a<-"10/20/1999"
DateConversion(a,DateIn="%m/%d/%Y",DateOut="%d/%m/%Y")
#Example 2
b<-"27/12/2000"
DateConversion(b,DateIn="%d/%m/%Y",DateOut="%m/%d/%Y")
#Example 3
c<-"20001223"
Date_New1<-DateConversion(c,DateIn="%Y%m%d",DateOut="%m/%d/%Y")
Date_New1
Date_New2<-DateConversion(c,DateIn="%Y%m%d",DateOut="%d/%m/%Y")
Date_New2

## End(Not run)
```

---

 stkde

*Spatio-Temporal Kernel Density Estimation with Density Contours*


---

**Description**

stkde calculates the three dimensional kernel density estimation of spatio-temporal mixed data, continuous space and discrete time.

**Usage**

```
stkde(xlong,ylat,ztime,xgrids,ygrids,breaks,alpha,nrowpar,...)
```

**Arguments**

xlong	Projected planar coordinates of longitude.
ylat	Projected planar coordinates of latitude.
ztime	The integer variable, such as YEAR, 1990, 1991 or 1, 2.
xgrids	Number of grids to evaluate the density in the x direction.
ygrids	Number of grids to evaluate the density in the y direction.
breaks	breaks is to be used to specify the interval size for a numeric vector of probabilities with values in [0,1]. Defaults to the 0.05.
alpha	Specify the density level for indicating the statistically significant regions. Its default value is 0.05.
nrowpar	specify the number of rows when plotting the figures in a panel. The default number is 1.
...	additional arguments supplied to control various aspects of stkde. These arguments are the same as npudensbw in the np package, see details there.

**Details**

stkde is a method to conduct the spatio-temporal kernel density estimation, when the time variable is discrete or categorical variable, not continuous variable.

**Value**

stkde returns a stkde object, with the following components:  
 bw: bandwidth(s), scale factor(s) or nearest neighbours for the data.  
 dens: kernel estimation of the density (cumulative distribution) at the evaluation points.

**Note**

If you are using data of mixed types, then it is advisable to use the data.frame function to construct your input data and not cbind, since cbind will typically not work as intended on mixed data types and will coerce the data to the same type.

**Author(s)**

Zhijie Zhang, <epistat@gmail.com>

**References**

Li, Q. and Racine, J.S. *Nonparametric Econometrics: Theory and Practice*, Princeton University Press. 2007.  
 Hayfield, T. and Racine, J.S. "Nonparametric Econometrics: The np Package," *Journal of Statistical Software*, 2008, 27(5): <http://www.jstatsoft.org/v27/i05/>.

**See Also**

npudensbw(np), npudens(np)

**Examples**

```
## Not run:
#Example1-uneven number of years
#Dataset1
# We will generate a 3 different stages' case points.
# The higher density are in the off-diagonal direction.
x1<-c(runif(100,0,1),runif(50,0.67,1))
y1<-c(runif(100,0,1),runif(50,0.67,1))
d1<-data.frame(x1,y1)
colnames(d1)<-c("x","y")
x2<-c(runif(100,0,1),runif(50,0.33,0.67))
y2<-c(runif(100,0,1),runif(50,0.33,0.67))
d2<-data.frame(x2,y2)
colnames(d2)<-c("x","y")
x3<-c(runif(100,0,1),runif(50,0,0.33))
y3<-c(runif(100,0,1),runif(50,0,0.33))
d3<-data.frame(x3,y3)
colnames(d3)<-c("x","y")
d<-rbind(d1,d2,d3)
```

```

d$tf<-c(rep(1,150),rep(2,150),rep(3,150))
#d is the simulated data
#d[1,]
#plot(d1);points(d2,col="red");points(d3,col="green")
#Key Code
#attach(d)
samkde<-stkde(xlong=d$x,ylat=d$y,ztime=d$tf,xgrids=20,ygrids=20,
              breaks=0.05,alpha=0.05,nrowspar=1,bwmethod="cv.ml")
samkde$bw
samkde$dens
#Example2-even number of years
#Dataset2
x12<-c(runif(100,0,1),runif(50,0.67,1))
y12<-c(runif(100,0,1),runif(50,0.67,1))
d12<-data.frame(x12,y12)
colnames(d12)<-c("x","y")
x22<-c(runif(100,0,1),runif(50,0.33,0.67))
y22<-c(runif(100,0,1),runif(50,0.33,0.67))
d22<-data.frame(x22,y22)
colnames(d22)<-c("x","y")
d2<-rbind(d12,d22)
d2$tf<-c(rep(1,150),rep(2,150))
colnames(d2)<-c("xlong","ylat","ztime")
#Running the function
samkde2<-stkde(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,
              alpha=0.05,nrowspar=1,bwmethod="cv.ml")
samkde2$bw
samkde2$dens

## End(Not run)

```

---

 stkde.base

*Spatio-Temporal Kernel Density Estimation*


---

## Description

stkde.base calculates the three dimensional kernel density estimation of spatio-temporal mixed data,continuous space and discrete time.

## Usage

```
stkde.base(xlong,ylat,ztime,xgrids,ygrids,breaks,...)
```

## Arguments

xlong	Projected planar coordinates of longitude.
ylat	Projected planar coordinates of latitude.
ztime	The integer variable,such as YEAR,1990,1991 or 1,2.
xgrids	Number of grids to evaluate the density in the x direction.

ygrids	Number of grids to evaluate the density in the y direction.
breaks	breaks is to be used to specify the interval size for a numeric vector of probabilities with values in [0,1]. Defaults to the 0.05.
...	additional arguments supplied to control various aspects of stkde. These arguments are the same as npudensbw in the np package, see details there.

### Details

stkde.base is a method to conduct the spatio-temporal kernel density estimation, when the time variable is discrete or categorical variable, not continuous variable.

### Value

stkde.base returns a stkde object, with the following components:  
 #bw: bandwidth(s), scale factor(s) or nearest neighbours for the data.  
 #dens: kernel estimation of the density (cumulative distribution) at the evaluation points.

### Note

If you are using data of mixed types, then it is advisable to use the data.frame function to construct your input data and not cbind, since cbind will typically not work as intended on mixed data types and will coerce the data to the same type.

### Author(s)

Zhijie Zhang, <epistat@gmail.com>

### References

Li, Q. and Racine, J.S. *Nonparametric Econometrics: Theory and Practice*, Princeton University Press. 2007.  
 Hayfield, T. and Racine, J.S. "Nonparametric Econometrics: The np Package," *Journal of Statistical Software*, 2008, 27(5): <http://www.jstatsoft.org/v27/i05/>.

### See Also

npudensbw(np), npudens(np)

### Examples

```
## Not run:
# EXAMPLE: Simulated dataset
# We will generate a 3 different stages' case points.
# The higher density are in the off-diagonal direction.
x1<-c(runif(100,0,1),runif(50,0.67,1))
y1<-c(runif(100,0,1),runif(50,0.67,1))
d1<-data.frame(x1,y1)
colnames(d1)<-c("x","y")
x2<-c(runif(100,0,1),runif(50,0.33,0.67))
y2<-c(runif(100,0,1),runif(50,0.33,0.67))
```

```

d2<-data.frame(x2,y2)
colnames(d2)<-c("x","y")
x3<-c(runif(100,0,1),runif(50,0,0.33))
y3<-c(runif(100,0,1),runif(50,0,0.33))
d3<-data.frame(x3,y3)
colnames(d3)<-c("x","y")
d<-rbind(d1,d2,d3)
d$tf<-c(rep(1,150),rep(2,150),rep(3,150))
#d is the simulated data
#d[,1,]
#plot(d1);points(d2,col="red");points(d3,col="green")
#Key Code
#attach(d)
samkde<-stkde.base(xlong=d$x,ylat=d$y,ztime=d$tf,xgrids=20,ygrids=20,
breaks=0.05,bwmethod="cv.ml")

## End(Not run)

```

---

stkde.sig

*Spatio-Temporal Kernel Density Estimation with Significant P-Value contours*


---

## Description

stkde.sig calculates the three dimensional kernel density estimation of spatio-temporal mixed data,continuous space and discrete time. And also obtain the significant p-value contours to indicate the TRUE significant areas by the method of Monte Carlo.

## Usage

```
stkde.sig(xlong,ylat,ztime,xgrids,ygrids,breaks,sim,alpha,nrowpar,...)
```

## Arguments

xlong	Same as the function of stkde.
ylat	Same as the function of stkde.
ztime	Same as the function of stkde.
xgrids	Same as the function of stkde.
ygrids	Same as the function of stkde.
breaks	Same as the function of stkde.
sim	Specify the number of simulations for Monte Carlo (sim-1). The default value is 100 and the actual simulated number is 100-1=99.
alpha	Specify the significant level for generating the statistically significant p-value contours. Its default value is 0.05.
nrowpar	specify the number of rows when plotting the figures in a panel. The default number is 1.
...	additional arguments supplied to control various aspects of stkde.These arguments are the same as npudensbw in the np package, see details there.

**Details**

stkde is a method to conduct the spatio-temporal kernel density estimation with significant p-value contours to indicate the statistically significant area, when the time variable is discrete or categorical variable, not continuous variable.

**Value**

stkde returns a stkde object, with the following two arrays. Their dimensions are xgrids, ygrids and tlength, respectively:

dens: kernel estimation of the density (cumulative distribution) at the evaluation points.

pvalue: P values for the high density to be significant high values.

**Note**

This method is important for deleting the false-positive results of stkde.

**Author(s)**

Zhijie Zhang, <epistat@gmail.com>

**References**

Li, Q. and Racine, J.S. *Nonparametric Econometrics: Theory and Practice*, Princeton University Press. 2007.

Hayfield, T. and Racine, J.S. "Nonparametric Econometrics: The np Package," *Journal of Statistical Software*, 2008, 27(5): <http://www.jstatsoft.org/v27/i05/>.

Zhang Z, Clark AB, Bivand R, Chen Y, Carpenter TE, Peng W, Zhou Y, Zhao G, Jiang Q. "Nonparametric spatial analysis to detect high-risk regions for schistosomiasis in Guichi, China," *Trans R Soc Trop Med Hyg.* 2009, 103(10):1045-1052.

**See Also**

npudensbw(np), npudens(np), stkde

**Examples**

```
## Not run:
#Example1-uneven number of years
#Dataset1
x1<-c(runif(100,0,1),runif(50,0.67,1))
y1<-c(runif(100,0,1),runif(50,0.67,1))
d1<-data.frame(x1,y1)
colnames(d1)<-c("x","y")
x2<-c(runif(100,0,1),runif(50,0.33,0.67))
y2<-c(runif(100,0,1),runif(50,0.33,0.67))
d2<-data.frame(x2,y2)
colnames(d2)<-c("x","y")
x3<-c(runif(100,0,1),runif(50,0,0.33))
y3<-c(runif(100,0,1),runif(50,0,0.33))
```

```

d3<-data.frame(x3,y3)
colnames(d3)<-c("x","y")
d<-rbind(d1,d2,d3)
d$tf<-c(rep(1,150),rep(2,150),rep(3,150))
colnames(d)<-c("xlong","y1at","ztime")
#Running the function
stkde.sig(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=1)
#reports the time spent in garbage collection so far in the R session while GC timing was enabled
gc.time(stkde.sig(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=1))
#Return CPU (and other) times that expr used.
system.time(stkde.sig(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=1))
#determines how much real and CPU time (in seconds) the currently running R process has already taken
proc.time(stkde.sig(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=1))
#
#Example2-even number of years
#Dataset2
x12<-c(runif(100,0,1),runif(50,0.67,1))
y12<-c(runif(100,0,1),runif(50,0.67,1))
d12<-data.frame(x12,y12)
colnames(d12)<-c("x","y")
x22<-c(runif(100,0,1),runif(50,0.33,0.67))
y22<-c(runif(100,0,1),runif(50,0.33,0.67))
d22<-data.frame(x22,y22)
colnames(d22)<-c("x","y")
d2<-rbind(d12,d22)
d2$tf<-c(rep(1,150),rep(2,150))
colnames(d2)<-c("xlong","y1at","ztime")
#Running the function
stkde.sig(d2[,1],d2[,2],d2[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=2)
#reports the time spent in garbage collection so far in the R session while GC timing was enabled
gc.time(stkde.sig(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=2))
#Return CPU (and other) times that expr used.
system.time(stkde.sig(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=2))
#determines how much real and CPU time (in seconds) the currently running R process has already taken
proc.time(stkde.sig(d[,1],d[,2],d[,3],xgrids=20,ygrids=20,breaks=0.05,sim=3,alpha=0.05,nrowspar=2))

## End(Not run)

```

---

UorL

*Converting all the Letters of Character Variable into Uppercase or Lowercase*

---

## Description

UorL was used to convert the letters for a character variable into uppercase or lowercase. The reason we need to do this is sometimes we may be required to compared the same character variable from different files or sources for a consistency check.

## Usage

```
UorL(x, charlower=TRUE)
```

**Arguments**

x	Specify the character variable whose letters to be changed into uppercase or lowercase.
charlower	Specify whether you want to get the uppercase letters or lowercase letters. It defaults to change all the letters into lowercase.

**Details**

UorL is an easy function to convert the letters for a character variable into uppercase or lowercase.

**Value**

UorL returns the results of conversion directly.

**Note**

None.

**Author(s)**

Zhijie Zhang, <epistat@gmail.com>

**See Also**

See also toupper and tolower functions.

**Examples**

```
## Not run:
#Example
l<-"IamAGenius"
UorL(l)
UorL(l,charlower=TRUE)
UorL(l,charlower=T)
UorL(l,charlower=FALSE)
UorL(l,charlower=F)

## End(Not run)
```

# Index

\*Topic **character**

UorL, [9](#)

\*Topic **date**

DateConversion, [2](#)

\*Topic **nonparametric,spatio-temporal analysis**

stkde, [3](#)

stkde.base, [5](#)

stkde.sig, [7](#)

\*Topic

DateConversion, [2](#)

UorL, [9](#)

DateConversion, [2](#)

stkde, [3](#)

stkde.base, [5](#)

stkde.sig, [7](#)

UorL, [9](#)