Package ‘GeNetIt’

April 1, 2020

Type Package
Title Spatial Graph-Theoretic Genetic Gravity Modelling
Version 0.1-4
Date 2020-03-31
Description Implementation of spatial graph-theoretic genetic gravity models. The model framework is applicable for other types of spatial flow questions. Includes functions for constructing spatial graphs, sampling and summarizing associated raster variables and building unconstrained and singly constrained gravity models.

Depends R (>= 3.6.0)
Importsexactextractr, methods, nlme, sp, spdep, raster, rgeos, sf
Maintainer Jeffrey S. Evans <jeffrey_evans@tnc.org>
License GPL-3
URL https://github.com/jeffreyevans/GeNetIt
NeedsCompilation no
Repository CRAN
Encoding UTF-8
LazyData true
RoxygenNote 7.1.0
Author Jeffrey S. Evans [aut, cre], Melanie Murphy [aut]
Date/Publication 2020-04-01 13:30:03 UTC

R topics documented:

area.graph.statistics .......................................................... 2
build.node.data ............................................................... 2
compare.models ............................................................... 3
dmatrix.df ................................................................. 4
dps ................................................................. 5
area.graph.statistics  Statistics for edges (lines) based on a defined scale (area).

Description
Samples rasters for each edge and calculates specified statistics for buffer distance

Usage
area.graph.statistics(...)

Arguments
...

Parameters to be passed to the modern version of the function

Note
Please note that this function has been deprecated, please use graph.statistics with the buffer argument.

build.node.data  Build node data

Description
Helper function to build the origin/destination node data structure.

Usage
build.node.data(x, group.ids, from.parms, to.parsms = NULL)
compare.models

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>A data.frame containing node (site) data</td>
</tr>
<tr>
<td>group.ids</td>
<td>Character vector of unique identifier that can be used to join to graph</td>
</tr>
<tr>
<td>from.parms</td>
<td>Character vector of independent &quot;from&quot; variables</td>
</tr>
<tr>
<td>to.parms</td>
<td>Character vector of independent &quot;to&quot; variables. If NULL is the same as from.parms</td>
</tr>
</tbody>
</table>

Value
data.frame

Note

Unless a different set of parameters will be used as the destination (to) there is no need to define the argument "to.parms" and the "from.parms" will be used to define both set of parameters.

The resulting data.frame represents the origin (from) and destination (to) data structure for use in gravity model. This is node structure is also know in the gravity literature as producer (from) and attractor (to).

Author(s)

Jeffrey S. Evans <jeffrey_evans@tnc.org> and Melanie Murphy <melanie.murphy@uwyo.edu>

Examples
data(ralu.site)

# Build from/to site (node) level data structure
site.parms = c("AREA_m2", "PERI_m", "Depth_m", "TDS")
site <- build.node.data(ralu.site@data, group.ids = c("SiteName"),
                         from.parms = site.parms )

compare.models

Compare gravity models

Description

Prints diagnostic statistics for comparing gravity models

Usage

compare.models(...)

Arguments

... gravity model objects
dmatrix.df

Distance matrix to data.frame

Description

Coerces distance matrix to a data.frame object

Usage

```r
dmatrix.df(x, rm.diag = TRUE)
```
Arguments

x               Symmetrical distance matrix  
rm.diag        (TRUE/FALSE) remove matrix diagonal, self values.

Value

data.frame object representing to and from values

Note

Function results in data.frame object with "X1" (FROM), "X2" (TO) and "distance" columns. The FROM column represents to origin ID, TO represents destination ID and distance is the associated matrix distance. These results can be joined back to the graph object using either the origin or destination ID's.

Author(s)

Jeffrey S. Evans <jeffrey.evans@tnc.org> and Melanie Murphy <melanie.murphy@uwyo.edu>

Examples

library(sp)
pts <- cbind( x=runif(15, 480933, 504250), y=runif(15, 4479433, 4535122))
pts <- SpatialPointsDataFrame(pts,
data.frame(ID=paste("ob",1:nrow(pts),sep="")))

# Create distance matrix
dm <- spDists(pts, pts)
colnames(dm) <- pts@data[,"ID"]
rownames(dm) <- pts@data[,"ID"]

# Coerce to data.frame with TO and FROM ID's and associated distance
dm.df <- dmatrix.df(dm)
head(dm.df)

dps

dps genetic distance matrix for Columbia spotted frog (Rana luteiventris)

Description

Subset of data used in Murphy et al., (2010)

Format

A 29 x 29 genetic distance matrix:
References


---

**flow**

*Convert distance to flow*

---

**Description**

Converts distance to flow (1-d) with or without data standardization

**Usage**

```r
flow(x, standardize = FALSE, rm.na = FALSE, diag.value = NA)
```

**Arguments**

- `x`: A numeric vector or matrix object representing distances
- `standardize`: (FALSE/TRUE) Row-standardize the data before calculating flow
- `rm.na`: (TRUE/FALSE) Should NA's be removed, if FALSE (default) they will be retained in the results
- `diag.value`: If `x` is a matrix, what diagonal matrix values should be used (default is NA)

**Value**

A vector or matrix representing flow values

**Author(s)**

Jeffrey S. Evans <jeffrey_evans@tnc.org> and Melanie Murphy <melanie.murphy@uwyo.edu>

**Examples**

```r
#### On a distance vector
flow(runif(10, 0,1))
flow(runif(10, 0,500), standardize = TRUE)

# With NA's
d <- runif(10, 0,1)
d[2] <- NA
flow(d)
flow(d, rm.na=TRUE)

#### On a distance matrix
dm <- as.matrix(dist(runif(5,0,1), diag = TRUE, upper = TRUE))
flow(dm)
```
graph.statistics  
Point sample and statistics for edges (lines)

Description
Samples rasters for each edge and calculates specified statistics

Usage
graph.statistics(x, r, stats = c("min", "mean", "max"), buffer = NULL)

Arguments
- x: sp SpatialLinesDataFrame or sf LINE object
- r: A rasterLayer, rasterStack or rasterBrick object
- stats: Statistics to calculate. If vectorized, can pass a custom statistic function.
- buffer: Buffer distance, radius in projection units. For statistics based on edge buffer distance

Value
data.frame object of statistics

Note
...

Examples

library(sp)
library(spdep)
library(raster)
data(rasters)
data(ralu.site)
xvars <- stack(rasters)
dist.graph <- knn.graph(ralu.site, row.names = ralu.site$SiteName, max.dist = 1500)
str(dist.graph@data)

skew <- function(x, na.rm = TRUE) {
  if (na.rm) x <- x[!is.na(x)]
  sum((x - mean(x))^3) / (length(x) * sd(x)^3)
}

# Moments on continuous raster data
system.time(
  stats <- graph.statistics(dist.graph, r = xvars[-6],
      stats = c("min", "median", "max", "var", "skew"))
)

# Proportional function on nominal raster data
p <- function(x) { length(x[x < 52]) / length(x) }

system.time(
  nstats <- graph.statistics(dist.graph, r = xvars[6],
      stats = "p")
)

# Based on 500m buffer distance around line(s)

system.time(
  stats <- graph.statistics(dist.graph, r = xvars[-6],
      stats = c("min", "median", "max", "var", "skew"),
      buffer = 500)
)

dist.graph@data <- data.frame(dist.graph@data, stats, nstats)
str(dist.graph@data)

---

**gravity**

**Gravity model**

**Description**

Implements Murphy et al., (2010) gravity model

**Usage**

`gravity(y, x, d, group, data, ln = TRUE, constrained = TRUE, ...)`

**Arguments**

- `y`: Name of dependent variable
- `x`: Character vector of independent variables
- `d`: Name of column containing distance
- `group`: Name of grouping column (from or to)
- `data`: data.frame object containing model data
- `ln`: Natural log transform data (TRUE/FALSE)
- `constrained`: Specify constrained model, if FALSE a linear model (lm) is run (TRUE/FALSE)
- `...`: Additional argument passed to nlme or lm
Details
The "group" factor defines the singly constrained direction (from or to) and the grouping structure for the origins. To specify a null (distance only or IBD) model just omit the x argument.
By default constrained models are fit by maximizing the restricted log-likelihood (REML), for maximum likelihood use the type="ML" argument which is passed to the lme function. If ln=TRUE the input data will be log transformed

Value
formula Model formula
gravity Gravity model
AIC AIC value for selected model
log.likelihood Restricted log-likelihood at convergence
x data.frame of independent variables
y Vector of dependent variable
groups Ordered factor vector of grouping variable
fit Model Fitted Values

Note
Depends: nlme, lattice

Author(s)
Jeffrey S. Evans <jeffrey_evans@tnc.org> and Melanie Murphy <melanie.murphy@uwyo.edu>

References
Murphy, M. A. & J.S. Evans. (in prep). GenNetIt: graph theoretical gravity modeling for landscape genetics

See Also
groupedData for how grouping works in constrained model
lme for constrained model ... options
lm for linear model ... options

Examples
library(nlme)
data(ralu.model)
str(ralu.model)

# Gravity model
```r
x = c("DEPTH_F", "HLI_F", "CTI_F", "cti", "ffp")
( gm <- gravity(y = "DPS", x = x, d = "DISTANCE", group = "FROM_SITE", 
    data = ralu.model, ln = FALSE) )

# Plot gravity results
par(mfrow=c(2,3))
for (i in 1:6) { plot(gm, type=i) }

# log likelihood of competing models
x = c("DEPTH_F", "HLI_F", "CTI_F", "cti", "ffp")
for(i in x[-1]) {
x1 = c(x[1], x[-which(x %in% i)])
ll <- gravity(y = "DPS", x = x1, d = "DISTANCE", group = "FROM_SITE", 
    data = ralu.model, ln = FALSE)$log.likelihood
  cat("log likelihood for parameter set: ", "(" , x1 , ")", " = \n", ll , ")
}

# Distance only (IBD) model
gravity(y = "DPS", d = "DISTANCE", group = "FROM_SITE", 
    data = ralu.model, ln = FALSE)
```

---

**gravity.es**  
**Effect Size**

**Description**

Cohen’s D effect size for gravity models

**Usage**

```r
gravity.es(x, actual.n = FALSE, alpha = 0.95)
```

**Arguments**

- `x`  
  gravity model object

- `actual.n`  
  (FALSE/TRUE) Use actual N or degrees of freedom in calculating Confidence Interval

- `alpha`  
  confidence interval

**Details**

Calculate Cohen’s D statistic for each effect in a gravity model object

**Value**

data.frame of parameter effect size
**Descriptive Statistics**

- **knn.graph**

- **Author(s)**
  Jeffrey S. Evans <jeffrey.evans@tnc.org> and Melanie Murphy <melanie.murphy@uwyo.edu>

- **References**

- **Examples**
  ```
  library(nlme)
  data(ralu.model)
  x = c("DEPTH_F", "HLL_F", "CTI_F", "cti", "ffp")
  gm_h1 <- gravity(y = "DPS", x = x, d = "DISTANCE", group = "FROM_SITE",
                   data = ralu.model, ln = FALSE, method="ML")
  gravity.es(gm_h1)
  ```

---

**knn.graph**

*Saturated or K Nearest Neighbor Graph*

**Description**

Creates a kNN or saturated graph SpatialLinesDataFrame object

**Usage**

```
knng.graph(  
    x,  
    row.names = NULL,  
    k = NULL,  
    max.dist = NULL,  
    sym = FALSE,  
    long.lat = FALSE,  
    drop.lower = FALSE  
)
```

**Arguments**

- `x` : sp SpatialPointsDataFrame object
- `row.names` : Unique row.names assigned to results
- `k` : K nearest neighbors, defaults to saturated (n(x) - 1)
- `max.dist` : Maximum length of an edge (used for distance constraint)
knn.graph

sym      Create symmetrical graph (FALSE/TRUE)
long.lat (FALSE/TRUE) Coordinates are longitude-latitude decimal degrees, in which case distances are measured in kilometers
drop.lower (FALSE/TRUE) Drop lower triangle of matrix (duplicate edges)

Value

SpatialLinesDataFrame object with:

• i Name of column in x with FROM (origin) index
• j Name of column in x with TO (destination) index
• from_ID Name of column in x with FROM (origin) region ID
• to_ID Name of column in x with TO (destination) region ID
• length Length of each edge (line) in projection units or kilometers if long.lat = TRUE

Note

...

Author(s)

Jeffrey S. Evans jeffrey_evans@tnc.org and Melanie Murphy melanie.murphy@uwyo.edu

References

Murphy, M. A. & J.S. Evans. (in prep). "GenNetIt: gravity analysis in R for landscape genetics"

Examples

library(sp)
data(ralu.site)

# Saturated spatial graph
sat.graph <- knn.graph(ralu.site, row.names=ralu.site@data[,"SiteName"])
head(sat.graph@data)

# Distanced constrained spatial graph
dist.graph <- knn.graph(ralu.site, row.names=ralu.site@data[,"SiteName"],
                        max.dist = 5000)

opar <- par(no.readonly=TRUE)
par(mfrow=c(1,2))
plot(sat.graph, col="grey")
points(ralu.site, col="red", pch=20, cex=1.5)
box()
title("Saturated graph")
plot(dist.graph, col="grey")
node.statistics

```r
points(ralu.site, col="red", pch=20, cex=1.5)
box()
title("Distance constrained graph")
par(opar)
```

descendant

node.statistics

raster statistics for nodes

Description

returns raster value or statistics (based on specified radius) for node

Usage

```r
node.statistics(x, r, buffer = NULL, stats = c("min", "median", "max"))
```

Arguments

- `x`: sp class SpatialPointsDataFrame object
- `r`: A rasterLayer, rasterStack or rasterBrick object
- `buffer`: Buffer distance, radius in projection units
- `stats`: Statistics to calculate. If vectorized, can pass a custom statistic function.

Value

data.frame object of at-node raster values or statistics

Note

If no buffer is specified, at-node raster values are returned

Examples

```r
library(sp)
library(spdep)
library(raster)
data(rasters)
data(ralu.site)
xvars <- stack(rasters)

skew <- function(x, na.rm = TRUE) {
  if (na.rm) x <- x[!is.na(x)]
  sum( (x - mean(x)) ^ 3 ) / ( length(x) * sd(x) ^ 3 )
}
```
# without buffer (values at point)

```r
system.time(
  stats <- node.statistics(rlalu.site, r = xvars[-6],
    stats = c("min", "median", "max", "var", "skew"))
)
```

# with 1000m buffer (values around points)

```r
system.time(
  stats <- node.statistics(rlalu.site, r = xvars[-6], buffer = 1000,
    stats = c("min", "median", "max", "var", "skew"))
)
```

dist.graph@data <- data.frame(dist.graph@data, stats, nstats)
str(dist.graph@data)

---

**plot.gravity**  
**Plot gravity model**

**Description**
Diagnostic plots gravity model with 6 optional plots.

**Usage**

```r
## S3 method for class 'gravity'
plot(x, type = 1, ...)
```

**Arguments**

- `x` Object of class gravity
- `type` Type of plot (default 1, model structure I)
- `...` Ignored

**Value**

defined plot

**Note**
Plot types available: 1 - Model structure I, 2 - Model structure II, 3 - Q-Q Normal - Origin random effects, 4 - Q-Q Normal - Residuals , 5 - Fitted values, 6 - Distribution of observed verses predicted

Depends: nlme, lattice

**Author(s)**

Jeffrey S. Evans <jeffrey_evans@tnc.org> and Melanie Murphy <melanie.murphy@uwyo.edu>
References
Murphy, M. A. & J.S. Evans. (in prep). "GenNetIt: gravity analysis in R for landscape genetics"

predict.gravity  Predict gravity model

Description
predict method for class "gravity"

Usage
## S3 method for class 'gravity'
predict(object, newdata, ...)

Arguments
object Object of class gravity
newdata New data, matching model parameters, used for obtaining the predictions
... Arguments passed to predict.lme or predict.lm

Value
Model predictions

print.gravity  Print gravity model

Description
summary method for class "gravity"

Usage
## S3 method for class 'gravity'
print(x, ...)

Arguments
x Object of class gravity
... Ignored
ralu.model

Columbia spotted frog (Rana luteiventris) data for specifying gravity model. Note, the data.frame is already log transformed.

Description

Subset of data used in Murphy et al., (2010)

Format

A data.frame with 190 rows (sites) and 19 columns (covariates):

ARMI_ID Unique ID
FROM_SITE Unique from site ID
TO_SITE Unique to site ID
FST FST genetic distance
DPS DPS genetic distance
DISTANCE Graph edge distance
DEPTH_F At site water depth
HLI_F Heat Load Index
CTI_F Wetness Index
DEPTH_T At site water depth
HLI_T Heat Load Index
CTI_T Wetness Index
hli Heat Load Index
cti Wetness Index
ffp Frost Free Period
err27 Roughness at 27x27 scale
rsp Relative Slope Position
ridge Percent Ridge Line
hab_ratio Ratio of suitable dispersal habitat

References

**Subset of site-level spatial point data for Columbia spotted frog (Rana luteiventris)**

**Description**

Subset of data used in Murphy et al., (2010)

**Format**

A SpatialPointsDataFrame with 31 obs. of 17 variables:

- **SiteName** Unique site name
- **Drainage** Source drainage
- **Basin** source basin
- **Substrate** Wetland substrate
- **NWI** USFWS NWI Wetland type
- **AREA_m2** Area of wetland
- **PERI_m** Perimeter of wetland
- **Depth_m** Depth of wetland
- **TDS** ...
- **FISH** Fish present
- **ACB** ...
- **AUC** ...
- **AUCV** ...
- **AUCC** ...
- **AUF** ...
- **AWOOD** ...
- **AUFV** ...

**References**

**rasters**

*Subset of raster data for Columbia spotted frog (Rana luteiventris)*

---

**Description**

Subset of data used in Murphy et al., (2010)

**Format**

A raster RasterStack:

- **rows**: 426
- **columns**: 358
- **resolution**: 30 meter
- **projection**: `"+proj=utm +zone=11 +datum=NAD83 +units=m +no_defs +ellps=GRS80 +towgs84=0,0,0"`
- **cti**: Compound Topographic Index ("wetness")
- **err27**: Elevation Relief Ratio
- **ffp**: Frost Free Period
- **gsp**: Growing Season Precipitation
- **hil**: Heat Load Index
- **nlcd**: USGS Landcover

**References**


---

**summary.gravity**

*Summarizing Gravity Model Fits*

---

**Description**

Summary method for class "gravity".

**Usage**

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

**Arguments**

- **object**: Object of class gravity
- **...**: Ignored
Note

Summary of lme or lm gravity model, AIC, log likelihood and Root Mean Square Error (RMSE) of observed verses predicted
Index

area.graph.statistics, 2
build.node.data, 2
compare.models, 3
dmatrix.df, 4
dps, 5
flow, 6
graph.statistics, 7
gravity, 8
gravity.es, 10
groupedData, 9
knn.graph, 11
lm, 9
lme, 9
node.statistics, 13
plot.gravity, 14
predict.gravity, 15
print.gravity, 15
ralu.model, 16
ralu.site, 17
rasters, 18
summary.gravity, 18