Package ‘GeNetIt’

June 15, 2023

Type Package
Title Spatial Graph-Theoretic Genetic Gravity Modelling
Version 0.1-6
Date 2023-06-04
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 (>= 4.2.0), nlme
Imports exactextractr, terra, sf, sfnetworks, igraph, dplyr, tidygraph, spdep
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 7.2.3
Author Jeffrey S. Evans [aut, cre], Melanie Murphy [aut]
Date/Publication 2023-06-15 00:02:12 UTC

R topics documented:

adj_matrix ................................................................. 2
area.graph.statistics ................................................... 3
build.node.data .......................................................... 3
compare.models ........................................................... 4
adj_matrix

Description

Creates a binary matrix of adjacencies based on from-to graph relationships (joins)

Usage

adj_matrix(i, j = NULL)

Arguments

i

a vector or, if j = NULL a data.frame with two columns indicating from-to relationships (joins)

j

If specified, i must be a vector of same length and the i,j vectors must represent joins

Value

A binary matrix

Author(s)

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

```r
library(sf)
data(ralu.site, package="GeNetIt")

p <- as(ralu.site, "sf")
g <- knn.graph(p[c(1,5,8,10,20,31),])
plot(st_geometry(g))

(ind <- sf::st_drop_geometry(g[,1:2])[1:10, ])
adj_matrix(ind)
adj_matrix(g$i[1:10], g$j[1:10])
```

---

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

```r
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

```r
build.node.data(x, group.ids, from.parms, to.parms = NULL)```
Arguments

x A data.frame containing node (site) data
group.ids Character vector of unique identifier that can be used to join to graph
from.parms Character vector of independent "from" variables
to.parms Character vector of independent "to" variables. If NULL is the same as from.parms

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 A. 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(sf::st_drop_geometry(ralu.site),
  group.ids = c("SiteName"),
  from.parms = site.parms )

compare.models

Compare gravity models

Description

Prints diagnostic statistics for comparing gravity models

Usage

closest.models(...)

Arguments

... gravity model objects
covariates

Details

Results include model name, AIX, BIC, log likelihood, RMSE and number of parameters

Value

data.frame of competing model statistics

Author(s)

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

References


Examples

```r
library(nlme)
data(ralu.model)
x = c("DEPTH_F", "HLI_F", "CTI_F", "cti", "ffp")
( null <- gravity(y = "DPS", x = c("DISTANCE"), d = "DISTANCE",
               group = "FROM_SITE", data = ralu.model, fit.method = "ML") )
( gm_h1 <- gravity(y = "DPS", x = x, d = "DISTANCE", group = "FROM_SITE",
               data = ralu.model, ln = FALSE, fit.method="ML") )
( gm_h2 <- gravity(y = "DPS", x = x[1:3], d = "DISTANCE", group = "FROM_SITE",
               data = ralu.model, ln = FALSE, fit.method="ML") )
( gm_h3 <- gravity(y = "DPS", x = x[c(4:5)], d = "DISTANCE", group = "FROM_SITE",
               data = ralu.model, ln = FALSE, fit.method="ML") )
#( gm_h4 <- gravity(y = "DPS", x = x[c(4:5)], d = "DISTANCE", group = "FROM_SITE",
# data = ralu.model, ln = FALSE, fit.method="REML") )
compare.models(null, gm_h1, gm_h2, gm_h3)
```

covariates

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

Description

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

A 30m LZW compressed tiff:
- **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


---

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.
dps

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

Examples

```r
library(sf)
pts <- data.frame(ID=paste0("ob",1:15), x=runif(15, 480933, 504250),
                  y=runif(15, 4479433, 4535122))
pts <- st_as_sf(pts, coords = c("x", "y"),
               crs = 32611, agr = "constant")

# Create distance matrix
dm <- st_distance(pts)
class(dm) <- setdiff(class(dm), "units")
attr(dm, "units") <- NULL
colnames(dm) <- pts$ID
rownames(dm) <- pts$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

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) the 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 A. Murphy <melanie.murphy@uwyo.edu>

Examples

#### 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.metrics**  

Graph Metrics

**Description**

Metrics on structural properties of graph (at nodes)

**Usage**

```r
graph.metrics(  
  x,  
  node.pts,  
  node.name = NULL,  
  direct = FALSE,  
  metric = c("betweenness", "degree", "closeness")
)
```

**Arguments**

- `x`: knn graph object from GeNetIt::knn.graph (sf LINESTRING)
- `node.pts`: sf POINT or sp SpatialPointsDataFrame object used as nodes to build x
- `node.name`: Column name in node.pts object that acts as the provides the unique ID. If not defined, defaults to row.names of node.pts
- `direct`: (FALSE/TRUE) Evaluate directed graph
- `metric`: ...

**Note**

Please note: graph metrics are not valid for a saturated graph (all connections)

**Author(s)**

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

**Examples**

```r
library(sf)
data(ralu.site, package="GeNetIt")

graph <- knn.graph(ralu.site, row.names=ralu.site$SiteName,  
  max.dist = 2500)  
plot(st_geometry(graph))

( m <- graph.metrics(graph, ralu.site, "SiteName") )

ralu.site <- merge(ralu.site, m, by="SiteName")  
# plot node betweenness
```
graph.statistics

Statistics for edges (lines)

Description

Extracts raster values 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 terra SpatRast or raster rasterLayer, rasterStack, 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

If the buffer argument is specified that, raster values within the specified buffer radius are extracted and included in the derived statistic(s). Else-wise, the statistics are derived from raster values that directly intersect each edge.

Author(s)

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

```r
library(sf)
library(terra)

data(ralu.site)
xvars <- rast(system.file("extdata/covariates.tif", package="GeNetIt"))

(dist.graph <- knn.graph(ralu.site, row.names = ralu.site$SiteName,
                         max.dist = 1500))

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)
}
```

Gravit

**Gravity model**

**Description**

Implements Murphy et al., (2010) gravity model via a linear mixed effects model
Usage

```r
gravity(
  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
  fit.method = c("REML", "ML"),  # Method used to fit model c("REML", "ML")
  ln = TRUE,  # Natural log transform data (TRUE/FALSE)
  constrained = TRUE,  # Specify constrained model, if FALSE a linear model (lm) is run (TRUE/FALSE)
  ...  # Additional argument passed to nlme or lm
)
```

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
- `fit.method`: Method used to fit model c("REML", "ML")
- `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 call
- `fixed.formula`: Model formula for fixed effects
- `random.formula`: Model formula for random (group) effects (only for constrained models)
- `gravity`: Gravity model
- `fit`: Model Fitted Values
- `AIC`: AIC value for selected model
- `RMSE`: Root Mean Squared Error (based on bias corrected back transform)
- `log.likelihood`: Restricted log-likelihood at convergence
group.names Column name of grouping variable
groups Values of grouping variable
x data.frame of x variables
y Vector of y variable
constrained TRUE/FALSE indicating if model is constrained

Note
Depends: nlme, lattice

Author(s)
Jeffrey S. Evans <jeffrey_evans@tnc.org> and Melanie A. 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)

# Gravity model
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: ", 
  paste("",x1,"", "=", ll, 
  "\n")
}
# Distance only (IBD) model
gravity(y = "DPS", d = "DISTANCE", group = "FROM_SITE",
data = ralu.model, ln = FALSE)

<table>
<thead>
<tr>
<th>gravity.es</th>
<th>Effect Size</th>
</tr>
</thead>
</table>

**Description**

Cohen’s D effect size for gravity models

**Usage**

`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

**Author(s)**

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

**References**


**Examples**

```r
library(nlme)
data(ralu.model)

x = c("DEPTH_F", "HLI_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**

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

**Arguments**

- `x`: sf POINTS 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)
- `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 representing duplicate edges ie, from-to and to-from
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 not projected

Note

...

Author(s)

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

References

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

Examples

library(sf)
  data(ralu.site, package="GeNetIt")

# Saturated spatial graph
sat.graph <- knn.graph(ralu.site, row.names=ralu.site$SiteName)
head(sat.graph)

# Distanced constrained spatial graph
dist.graph <- knn.graph(ralu.site, row.names=ralu.site$SiteName, max.dist = 5000)

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

raster statistics for nodes

Description

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

Usage

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

Author(s)

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

Examples

library(sf)
library(terra)
data(ralu.site)
xvars <- rast(system.file("extdata/covariates.tif", package="GeNetIt"))

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)
system.time( {
  stats <- node.statistics(ralu.site, r = xvars[[6]])
  }
)
```r
# with 1000m buffer (values around points)
system.time(
  stats <- node.statistics(ernalu.site, r = xvars[-6], buffer = 1000,
                        stats = c("min", "median", "max", "var", "skew"))
)
```

---

### 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 A. 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**

```r
## S3 method for class 'gravity'
predict(
  object,
  newdata,
  groups = NULL,
  back.transform = c("none", "simple", "Miller", "Naihua"),
  ...
)
```

**Arguments**

- `object`: Object of class gravity
- `newdata`: New data used for obtaining the predictions, can be a data.frame or nffGroupedData
- `groups`: Grouping factor acting as random effect. If used, must match levels used in model, otherwise leave it null and do not convert to groupedData
- `back.transform`: Method to back transform data, default is none and log predictions will be returned.
- `...`: Arguments passed to predict.lme or predict.lm

**Details**

Please note that the entire gravity equation is log transformed so, your parameter space is on a log scale, not just y. This means that for a meaningful prediction the "newdata" also needs to be on a log scale.

For the back.transform argument, the simple back-transform method uses the form \( \exp(\hat{y})0.5 \times \text{variance} \) whereas Miller uses \( \exp(\sigma)0.5 \) as the multiplicative bias factor. Naihua regresses \( y - \exp(\hat{y}) \) with no intercept and uses the resulting coefficient as the multiplicative bias factor. The Naihua method is intended for results with non-normal errors. You can check the functional form by simply plotting \( y \) (non-transformed) against the fit. The default is to output the log scaled predictions.

**Value**

Vector of model predictions

**Author(s)**

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


Examples

library(nlme)

data(ralu.model)

back.transform <- function(y) exp(y + 0.5 * stats::var(y, na.rm=TRUE))
rmse = function(p, o){ sqrt(mean((p - o)^2)) }

x = c("DEPTH_F", "HLI_F", "CTI_F", "cti", "ffp")
sidx <- sample(1:nrow(ralu.model), 100)
train <- ralu.model[sidx,]
test <- ralu.model[-sidx,]

# Specify constrained gravity model
( gm <- gravity(y = "DPS", x = x, d = "DISTANCE", group = "FROM_SITE", data = train, ln = FALSE) )

( p <- predict(gm, test[,c(x, "DISTANCE")]) )
rmse(back.transform(p), back.transform(ralu.model[,"DPS"][-sidx]))

# WIth model sigma-based back transformation
( p <- predict(gm, test[,c(x, "DISTANCE")], back.transform = "simple") )
( p <- predict(gm, test[,c(x, "DISTANCE")], back.transform = "Miller") )
( p <- predict(gm, test[,c(x, "DISTANCE")], back.transform = "Naihua") )

# Using grouped data
test <- nlme::groupedData(stats::as.formula(paste(paste("DPS", 1, sep = " ~ "), "FROM_SITE", sep = " | ")), data = test[,c("DPS", "FROM_SITE", x, "DISTANCE")])

( p <- predict(gm, test, groups = "FROM_SITE") )
( y.hat <- back.transform(ralu.model[,"DPS"][-sidx]) )
na.idx <- which(is.na(p))
rmse(back.transform(p)[-na.idx], y.hat[-na.idx])

# Specify unconstrained gravity model (generally, not recommended)
( gm <- gravity(y = "DPS", x = x, d = "DISTANCE", group = "FROM_SITE", data = train, ln = FALSE, constrained=TRUE) )

( p <- predict(gm, test[,c(x, "DISTANCE")]) )
rmse(back.transform(p), back.transform(ralu.model[,"DPS"][-sidx]))
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

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

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

Format
An sf POINT object with 31 obs. of 17 variables:

SiteName  Unique site name
Drainage  Source drainage
Basin  source basin
Substrate  Wetland substrate
NW1  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  ...
summary.gravity

References

---

summary.gravity Summarizing Gravity Model Fits

Description
Summary method for class "gravity".

Usage
## 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

adj_matrix, 2
area.graph.statistics, 3
build.node.data, 3
covariates, 5
dmatrix.df, 6
dps, 7
flow, 8
graph.metrics, 9
graph.statistics, 10
gravity, 11
gravity.es, 14
groupedData, 13
knn.graph, 15
lm, 13
lme, 13
node.statistics, 17
plot.gravity, 18
predict.gravity, 19
print.gravity, 21
ralu.model, 21
ralu.site, 22
summary.gravity, 23