Package ‘sharpshootR’

February 20, 2015

Type Package
Title A Soil Survey Toolkit
Description Miscellaneous soil data management, summary, visualization, and conversion utilities to support soil survey.
Version 0.7-2
Date 2015-02-19
Author USDA-NRCS Soil Survey Staff
Maintainer Dylan Beaudette <dylan.beaudette@ca.usda.gov>
License GPL (>= 2)
Repository CRAN
URL http://aqp.r-forge.r-project.org/
Suggests MASS, rgdal, spdep
Depends R (>= 2.15.0)
Imports aqp, ape, soilDB, igraph, cluster, lattice, latticeExtra,
      vegan, sp, reshape2, Hmisc, scales, circular, RColorBrewer,
      plyr
Repository/R-Forge/Project aqp
Repository/R-Forge/Revision 996
Repository/R-Forge/DateTimeStamp 2015-02-19 20:00:45
Date/Publication 2015-02-19 22:34:22
NeedsCompilation no

R topics documented:

  sharpshootR-package .................................................. 2
  amador ................................................................. 2
  aspect.plot ............................................................ 3
  CDEC.snow.courses ................................................... 4
  CDEC.query ............................................................ 5
  CDECsnowQuery ....................................................... 8
sharpshootR-package  A collection of functions to support soil survey

Description

A collection of functions to support soil survey

amador  SSURGO Data Associated with the Amador Soil Series

Description

SSURGO Data Associated with the Amador Soil Series

Usage

data(amador)

Format

A subset of data taken from the "component" table of SSURGO

- mukey  map unit key
- compname  component name
- compperc_r  component percentage

Source

USDA-NRCS SSURGO Database
aspect.plot  

Plot Aspect Data

Description

Plot a graphical summary of multiple aspect measurements on a circular diagram.

Usage

aspect.plot(p, q=c(0.05, 0.5, 0.95), p.bins = 60, p.bw = 30, stack=TRUE, p.axis = seq(0, 350, by = 10), plot.title = NULL, line.col='RoyalBlue', line.lwd=1, line.lty=2, arrow.col=line.col, arrow.lwd=1, arrow.lty=1, arrow.length=0.15, ...)

Arguments

p  a vector of aspect angles in degrees, measured clock-wise from North
q  a vector of desired quantiles
p.bins  number of bins to use for circular histogram
p.bw  bandwidth used for circular density estimation
stack  TRUE/FALSE, should the individual points be stacked into p.bins number of bins and plotted
p.axis  a sequence of integers (degrees) describing the circular axis
plot.title  an informative title
line.col  density line color
line.lwd  density line width
line.lty  density line line style
arrow.col  arrow color
arrow.lwd  arrow line width
arrow.lty  arrow line style
arrow.length  arrow head length
...  further arguments passed to plot.circular

Details

Spread and central tendency are depicted with a combination of circular histogram and kernel density estimate. The circular mean, and relative confidence in that mean are depicted with an arrow: longer arrow lengths correspond to greater confidence in the mean.
Note

Manual adjustment of p.bw may be required in order to get an optimal circular density plot. This function requires the package circular, version 0.4-7 or later.

Author(s)

D.E. Beaudette

Examples

```r
# simulate some data
p.narrow <- runif(n=25, min=215, max=280)
p.wide <- runif(n=25, min=0, max=270)

# set figure margins to 0, 2-column plot
par(mar=c(0,0,0,0), mfcol=c(1,2))

# plot
aspect.plot(p.narrow, p.bw=10, plot.title='Soil A', pch=21, col='black', bg='RoyalBlue')
aspect.plot(p.wide, p.bw=10, plot.title='Soil B', pch=21, col='black', bg='RoyalBlue')
```

---

CDEC.snow.courses

**CDEC Snow Course List**

**Description**

The CDEC snow course list, updated 2014

**Usage**

data(CDEC.snow.courses)

**Format**

A data frame with 261 observations on the following 8 variables.

course_number  course number
name  connotative course label
id  course ID
elev_feet  course elevation in feet
latitude  latitude
longitude  longitude
april.1 Avg. inches  average inches of snow as of April 1st
agency  responsible agency
Source

Data were scraped from http://cdec.water.ca.gov/misc/SnowCourses.html, 2014.

Examples

```r
data(CDEC.snow.courses)
head(CDEC.snow.courses)
```

Description

Get water-related data (California only) from the CDEC website.

Usage

```r
CDECquery(id, sensor, interval = "D", start, end)
```

Arguments

- **id**: station ID (e.g. 'spw'), see details
- **sensor**: the sensor ID (e.g. 45), see details
- **interval**: character, 'D' for daily, 'H' for hourly, 'M' for monthly
- **start**: starting date, in the format 'YYYY-MM-DD'
- **end**: ending date, in the format 'YYYY-MM-DD'

Details

1. Station IDs can be found here: http://cdec.water.ca.gov/staInfo.html
2. Sensor IDs can be found using this URL: http://cdec.water.ca.gov/cgi-progs/queryCSV?
   ```
   station_id=, followed by the station ID.
   ```
3. Reservoir capacities can be found here: http://cdec.water.ca.gov/misc/resinfo.html
4. A new interactive map of CDEC stations can be found here: http://cdec.water.ca.gov/cdecstation/

Value

a data.frame object with the following fields: 'datetime', 'year', 'month', 'value'.

Author(s)

D.E. Beaudette
References

http://cdec.water.ca.gov/queryCSV.html

See Also

CDECsnowQuery

Examples

```r
## Not run:
library(latticeExtra)
library(plyr)
library(e1071)

# get daily reseroir storage (ac. ft) from Pinecrest, New Melones and Lyons reserosirs
pinecrest <- CDECquery(id='swb', sensor=15, interval='D', start='2012-09-01', end='2015-01-01')
new.melones <- CDECquery(id='nml', sensor=15, interval='D', start='2012-09-01', end='2015-01-01')
lyons <- CDECquery(id='lys', sensor=15, interval='D', start='2012-09-01', end='2015-01-01')

# compute storage capacity
pinecrest$capacity <- pinecrest$value / 18312 * 100
new.melones$capacity <- new.melones$value / 2400000 * 100
lyons$capacity <- lyons$value / 6228 * 100

# combine
g <- make.groups(new.melones, lyons, pinecrest)

# resonable date scale
r <- range(g$datetime)
s.r <- seq(from=r[1], to=r[2], by='1 month')

# better colors
tps <- list(superspose.line=list(lwd=2, col=brewer.pal(n=3, name='Set1')))

# plot
xyplot(capacity ~ datetime, groups=g, which=1, type='l',
    xlab='', ylab='Capacity (ac. ft)
    scales=list(x=list(at=s.r, labels=format(s.r, "
    auto.key=list(columns=3, lines=TRUE, points=FALSE),
    par.settings=tps,
    panel=function(...) {
        panel.abline(h=seq(0, 100, by=10), col='grey')
        panel.abline(v=s.r, col='grey')
        panel.xyplot(...)
    })

##
# New Melones monthly data, retrieve as far back in time as possible
new.melones.monthly <- CDECquery(id='nml', sensor=15, interval='M',
    start='1900-01-01', end='2015-01-01')
```
# convert to pct. capacity
new.melones.monthly$capacity <- new.melones.monthly$value / 2400000 * 100

# make a nice color ramp function
cols <- colorRampPalette(brewer.pal(9, 'Spectral'),
                         space='Lab', interpolate='spline')

# plot, each pixel is colored by the total precip by year/month
levelplot(capacity ~ year * month, data=new.melones.monthly, col.regions=cols, xlab='', ylab='', scales=list(x=list(tick.number=20)), main='New Melones Capacity ('

##

## get daily precip totals from Stan Powerhouse
x <- CDECquery(id='spw', sensor=45, interval='D', start='1900-01-01', end='2015-01-01')

# compute total precip by year/month
a <- ddply(x, c('year', 'month'), summarize, s=sum(value, na.rm=TRUE))

# convert monthly precipitation values into Z-scores by month
a.scaled <- ddply(a, 'month', summarize, year=year, scaled.ppt=scale(s))

# make a nice color ramp function, scaled by the skewness of the underlying distribution
cols <- colorRampPalette(brewer.pal(9, 'Spectral'),
                         space='Lab', interpolate='spline', bias=skewness(a.scaled$scaled.ppt, na.rm=TRUE))

# plot, each pixel is colored by the total precip by year/month
levelplot(scaled.ppt ~ year * month, data=a.scaled, col.regions=cols, xlab='', ylab='', scales=list(x=list(tick.number=10)), main='Monthly Total Precipitation (as z-score) SPW')

##

## get pre-aggregated monthly data from Sonora RS
x <- CDECquery(id='sor', sensor=2, interval='M', start='1900-01-01', end='2015-01-01')

# make a nice color ramp function, scaled by the skewness of the underlying distribution
cols <- colorRampPalette(brewer.pal(9, 'Spectral'), space='Lab',
                         interpolate='spline', bias=skewness(x$value, na.rm=TRUE))

# plot
levelplot(value ~ year * month, data=x, col.regions=cols, xlab='', ylab='', scales=list(x=list(tick.number=20)), main='Monthly Total Precipitation (inches) SOR')

## End(Not run)
CDECsnowQuery

Get snow survey data (California only) from the CDEC website.

Description
Get snow survey data (California only) from the CDEC website.

Usage
CDECsnowQuery(course, start_yr, end_yr)

Arguments

- course: integer, course number (e.g. 129)
- start_yr: integer, the starting year (e.g. 2010)
- end_yr: integer, the ending year (e.g. 2013)

Details
This function downloads data from the CDEC website, therefore an internet connection is required. The ‘SWE’ column contains adjusted SWE if available (‘Adjusted’ column), otherwise the reported SWE is used (‘Water’ column).

Value
a data.frame object, see examples

Note
Snow course locations, ID numbers, and other information can be found here: http://cdec.water.ca.gov/misc/SnowCourses.html

Author(s)
D.E. Beaudette

References
http://cdec.water.ca.gov/cgi-progs/snowQuery

Examples
## Not run:
# get data for course numbe 129
x <- CDECsnowQuery(course=129, start_yr=2010, end_yr=2011)

## End(Not run)
create an adjacency matrix from a data.frame of component data

Description

Create an adjacency matrix from SSURGO component data

Usage

```
component.adj.matrix(d, mu='mukey', co='compname', wt='comp pct_r',
                     method='community.matrix', standardization='max', metric='jaccard',
                     rm.orphans=TRUE, similarity=TRUE)
```

Arguments

- `d`: a data.frame, typically of SSURGO data
- `mu`: name of the column containing the map unit ID (typically 'mukey')
- `co`: name of the column containing the component ID (typically 'compname')
- `wt`: name of the column containing the component weight percent (typically 'comp pct_r')
- `method`: one of either: 'community.matrix', or 'occurrence'; see details
- `standardization`: community matrix standardization method, passed to decostand
- `metric`: community matrix dissimilarity metric, passed to vegdist
- `rm.orphans`: logical, should map units with a single component be omitted? (typically yes)
- `similarity`: logical, return a similarity matrix? (if FALSE, a distance matrix is returned)

Details

Pending...

Value

A similarity matrix / adjacency matrix suitable for use with igraph functions or anything else that can accommodate a similarity matrix.

Author(s)

D.E. Beaudette
**diagnosticPropertyPlot**

*Diagostic Property Plot*

**Description**

Generate a graphical description of the presence/absence of soil diagnostic properties.

**Usage**

```r
diagnosticPropertyPlot(f, v, k, id = "pedon_id")
diagnosticPropertyPlot2(f, v, k, id = "pedon_id")
```

**Arguments**

- `f`: a SoilProfileCollection object
- `v`: a character vector of site-level attribute names that are boolean (e.g. TRUE/FALSE) data
- `k`: an integer, number of groups to highlight
- `id`: the name of the site-level attribute containing an ID to include next to leave of the diagnostic property dendrogram

**Details**

This function attempts to display several pieces of information within a single figure. First, soil profiles are sorted according to the presence/absence of diagnostic features named in `v`. Second, these diagnostic features are sorted according to their distribution among soil profiles. Third, a binary grid is established with row-ordering of profiles based on step 1 and column-ordering based on step 2. Blue cell represent the presence of a diagnostic feature. Soils with similar diagnostic features should 'clump' together. See examples below.

**Examples**

```r
# load sample data set
data(amonador)

# convert into adjacency matrix
m <- component.adj.matrix(amonador)

# plot network diagram, with Amador soil highlighted
plotSoilRelationGraph(m, s='amonador')
```
**Value**

a list is silently returned by this function, containing:

- `rd` a `data.frame` containing IDs and grouping code
- `profile.order` a vector containing the order of soil profiles (row-order in figure), according to diagnostic property values
- `var.order` a vector containing the order of variables (column-order in figure), according to their distribution among profiles

**Note**

diagnosticPropertyPlot2 is a lattice-based version, and requires a unique value passed to `id`.

**Author(s)**

D.E. Beaudette and J.M. Skovlin

**Examples**

```r
## Not run:
# sample data, an SPC
data(gopheridge, package='soilDB')

# get depth class
sdc <- getSoilDepthClass(gopheridge)
site(gopheridge) <- sdc

# diagnostic properties to consider, no need to convert to factors
v <- c('lithic.contact', 'paralithic.contact', 'argillic.horizon', 'cambic.horizon', 'ochric.epipedon', 'mollic.epipedon', 'very.shallow', 'shallow', 'mod.deep', 'deep', 'very.deep')

# base graphics
x <- diagnosticPropertyPlot(gopheridge, v, k=5)

# lattice graphics
x <- diagnosticPropertyPlot2(gopheridge, v, k=3)

# check output
str(x)
```

## End(Not run)
dist.along.grad

Compute Euclidean distance along a gradient.

Description
This function computes Euclidean distance along points aligned to a given gradient (e.g. elevation).

Usage
dist.along.grad(coords, var, grad.scaled.min, grad.scaled.max)

Arguments

- `coords` a matrix of x and y coordinates in some projected coordinate system
- `var` a vector of the same length as `coords`, describing the gradient of interest
- `grad.scaled.min` min value of rescaled gradient values
- `grad.scaled.max` max value of rescaled gradient values

Details
This function is primarily intended for use within `plottransect`.

Value
A `data.frame` object:

- `scaled.grad` scaled gradient values
- `scaled.distance` cumulative distance, scaled to the interval of `0.5, nrow(coords) + 0.5`
- `distance` cumulative distance computed along gradient, e.g. transect distance
- `variable` sorted gradient values
- `x` x coordinates, ordered by gradient values
- `y` y coordinate, ordered by gradient values
- `grad.order` a vector index describing the sort order defined by gradient values

Note
This function is very much a work in progress, ideas welcome.

Author(s)
D.E. Beaudette

See Also
`plotTransect`
Duelling Dendrograms

Description

Graphically compare two related dendrograms

Usage

```r
dueling.dendrograms(p.1, p.2, lab.1 = "D1",
lab.2 = "D2", cex.nodelabels=0.75, arrow.length=0.05)
```

Arguments

- `p.1`: left-hand phylo-class dendrogram
- `p.2`: right-hand phylo-class dendrogram
- `lab.1`: left-hand title
- `lab.2`: right-hand title
- `cex.nodelabels`: character expansion size for node labels
- `arrow.length`: arrow head size

Details

Connector arrows are used to link nodes from the left-hand dendrogram to the right-hand dendrogram.

Author(s)

D. E. Beaudette

Examples

```r
library(aqp)
library(cluster)
library(ape)

# load sample dataset from aqp package
data(sp3)

# promote to SoilProfileCollection
depths(sp3) <- id ~ top + bottom

# compute dissimilarity using different sets of variables
# note that these are rescaled to the interval [0,1]
d.1 <- profile_compare(sp3, vars=c('clay', 'cec'), k=0, max_d=100, rescale.result=TRUE)
d.2 <- profile_compare(sp3, vars=c('clay', 'l'), k=0, max_d=100, rescale.result=TRUE)
```
hillslope.probability  

**Hillslope Probability via SDA**

**Description**

Hillslope position probability estimates from the SDA query service (SSURGO)

**Usage**

hillslope.probability(s)

**Arguments**

s  

a character vector of soil series names, in lower-case

**Details**

This function constructs and executes a query that is sent to the [http://sdmdataaccess.nrcs.usda.gov](http://sdmdataaccess.nrcs.usda.gov) webservice. Further information on the SDA webservice and query examples can be found at [http://sdmdataaccess.nrcs.usda.gov/QueryHelp.aspx](http://sdmdataaccess.nrcs.usda.gov/QueryHelp.aspx)

**Value**

A `data.frame` object with rows representing soil series, and columns representing probability estimates of that series occurring along the standard 2D hillslope positions.

**Note**

Probability values are computed from SSURGO data.
plotAvailWater

Author(s)

D.E. Beaudette

Examples

```r
## Not run:
library(soilDB)

# soil series of interest
s <- c('auburn', 'pardee', 'amador', 'pentz')

# generate hillslope probability table
hillslope.probability(s)

## End(Not run)
```

plotAvailWater | Visual Demonstration of Available Soil Water

Description

Generate a simplistic diagram of the various fractions of water held within soil pore-space.

Usage

```r
plotAvailWater(x, width = 0.25, cols = c(grey(0.5),
"DarkGreen", "LightBlue", "RoyalBlue"), name.cex = 0.8)
```

Arguments

- `x`: a data.frame containing sample names and water retention data, see examples below
- `width`: vertical width of each bar graph
- `cols`: a vector of colors used to symbolize 'solid phase', 'unavailable water', 'available water', and 'gravitational water'
- `name.cex`: character scaling of horizon names, printed on left-hand side of figure

Author(s)

D.E. Beaudette
Examples

```r
# demonstration
s <- data.frame(
  name=c('loamy sand', 'sandy loam', 'silt loam', 'clay loam'),
  pwp=c(0.05, 0.1, 0.18, 0.2),
  fc=c(0.1, 0.2, 0.38, 0.35),
  sat=c(0.25, 0.3, 0.45, 0.4))

s$solid <- with(s, 1-sat)

par(mar=c(5, 6, 0.5, 0.5))
plotAvailWater(s, name.cex=1.25)

# use some real data from SSURGO
## Not run:
library(soilDB)

q <- "SELECT hzdept_r as hztop, hzdepb_r as hzbotttom,
  hzname as name, wsatiated_r/100.0 as sat,
  wthirdbar_r/100.0 as fc, wfiveeenbar_r/100.0 as pwp, awc_r as awc
FROM chorizon
WHERE cokey IN (SELECT cokey from component where compname = 'dunstone')
AND wsatiated_r IS NOT NULL
ORDER BY cokey, hzdept_r ASC;"

x <- SDA_query(q)
x <- unique(x)
x <- x[order(x$name), ]
x$solid <- with(x, 1-sat)

par(mar=c(5, 5, 0.5, 0.5))
plotAvailWater(x)
```

## plotSoilRelationGraph

Plot a component relation graph based on an adjacency or similarity matrix.

### Usage

```r
plotSoilRelationGraph(m, s='', plot.style='network', spanning.tree=NULL, 
del.edges=NULL, vertex.scaling.factor=2, edge.scaling.factor=1, 
edge.transparency=1, edge.col=grey(0.5), edge.highlight.col='royalblue',
g.layout=layout.fruchterman.reingold,
...)
```
plotSoilRelationGraph

Arguments

- **m**: adjacency matrix
- **s**: central component; an empty character string is interpreted as no central component
- **plot.style**: plot style (‘network’, or ‘dendrogram’)
- **spanning.tree**: plot the minimum or maximum spanning tree (‘min’, ‘max’), or, max spanning tree plus edges with weight greater than the n-th quantile specified in ‘spanning.tree’. See details and examples.
- **del.edges**: optionally delete edges with weights less than the specified quantile (0-1)
- **vertex.scaling.factor**: scaling factor applied to vertex size
- **edge.scaling.factor**: optional scaling factor applied to edge width
- **edge.transparency**: optional transparency setting for edges (0-1)
- **edge.col**: edge color, applied to all edges
- **edge.highlight.col**: edge color applied to all edges connecting to component named in s
- **g.layout**: an igraph layout function, defaults to layout.fruchterman.reingold
- **...**: further arguments passed to plotting function

Details

Vertex size is based on a normalized index of connectivity: size = sqrt(degree(g)/max(degree(g))) * scaling.factor. Edge width can be optionally scaled by edge weight by specifying an edge.scaling.factor value. The maximum spanning tree represents a sub-graph where the sum of edge weights are maximized. The minimum spanning tree represents a sub-graph where the sum of edge weights are minimized. The maximum spanning tree is likely a more useful simplification of the full graph, in which only the strongest relationships (e.g. most common co-occurrences) are preserved.

The maximum spanning tree + edges with weights > n-th quantile is an experimental hybrid. The 'backbone' of the graph is created by the maximum spanning tree, and augmented by 'strong' auxiliary edges-- defined by a value between 0 and 1.

Value

an igraph ‘graph’ object is invisibly returned

Note

This function is a work in progress, ideas welcome.

Author(s)

D.E. Beaudette
Examples

# load sample data set
data(amatador)

# create weighted adjacency matrix (see ?component.adj.matrix for details)
m <- component.adj.matrix(amatador)

# plot network diagram, with Amador soil highlighted
plotSoilRelationGraph(m, s='amatador')

# dendrogram representation
plotSoilRelationGraph(m, s='amatador', plot.style='dendrogram')

# compare methods
m.o <- component.adj.matrix(amatador, method='occurrence')

par(mfcol=c(1,2))
plotSoilRelationGraph(m, s='amatador', plot.style='dendrogram')
title('community matrix')
plotSoilRelationGraph(m.o, s='amatador', plot.style='dendrogram')
title('occurrence')

# investigate max spanning tree
plotSoilRelationGraph(m, spanning.tree='max')

# investigate max spanning tree + edges with weights > 75-th pctile
plotSoilRelationGraph(m, spanning.tree=0.75)

## Not run:
# get similar data from soilweb, for the Pardee series
s <- 'pardee'
u <- url(paste('http://casoilresource.lawr.ucdavis.edu/soil_web/reflector_api/soils.php?',
    'what=soil_series_component_query&q_string=', s, sep=''))

# fetch data
d <- read.table(u, sep='|', header=TRUE, stringsAsFactors=FALSE)

# normalize component names
d$compname <- tolower(d$compname)

# keep only major components
d <- subset(d, subset=compkind == 'Series')

# build adj. matrix and plot
m <- component.adj.matrix(d)
plotSoilRelationGraph(m, s=s, plot.style='dendrogram')

# alter plotting style, see ?plot.phylo
plotSoilRelationGraph(m, s=s, plot.style='dendrogram', type='fan')
plotSoilRelationGraph(m, s=s, plot.style='dendrogram', type='unrooted', use.edge.length=FALSE)
## plotTransect

*Plot a collection of Soil Profiles linked to their position along some gradient (e.g. transect).*

### Description

Plot a collection of Soil Profiles linked to their position along some gradient (e.g. transect).

### Usage

```r
plotTransect(s, grad.var.name, transect.col = "RoyalBlue",
             tick.number = 7, y.offset = 100,
             scaling.factor = 0.5,
             distance.axis.title = "Distance Along Transect (km)",
             crs = NULL, grad.axis.title = NULL, ...)
```

### Arguments

- `s`: a SoilProfileCollection object
- `grad.var.name`: the name of a site-level attribute containing gradient values
- `transect.col`: color used to plot gradient (transect) values
- `tick.number`: number of desired ticks and labels on the gradient axis
- `y.offset`: vertical offset used to position profile sketches
- `scaling.factor`: scaling factor used to adjust profile sketches
- `distance.axis.title`: a title for the along-transect distances
- `crs`: an optional CRS object used to convert coordinates into a planar system
- `grad.axis.title`: a title for the gradient axis
- `...`: further arguments passed to `plotSPC`

### Details

Depending on the nature of your SoilProfileCollection and associated gradient values, it may be necessary to tinker with figure margins, `y.offset` and `scaling.factor`. 
Value
An invisibly-returned data.frame object:

- **scaled.grad** scaled gradient values
- **scaled.distance** cumulative distance, scaled to the interval of \(0.5, nrow(coords) + 0.5\)
- **distance** cumulative distance computed along gradient, e.g. transect distance
- **variable** sorted gradient values
- **x** x coordinates, ordered by gradient values
- **y** y coordinate, ordered by gradient values
- **grad.order** a vector index describing the sort order defined by gradient values

Note
This function is very much a work in progress, ideas welcome!

Author(s)
D.E. Beaudette

Examples
```r
## Not run:
data(loafercreek, package='soildb')
x <- loafercreek[1:10,]
coordinates(x) <- ~ x_std + y_std
proj4string(x) <- '+proj=longlat +datum=NAD83'
par(mar=c(4,3,1,1))
plotTransect(x, 'elev_field', crs=CRS('+proj=utm +zone=10 +datum=NAD83'), max.depth=185)
## End(Not run)
```

describe polygonAdjacency Evaluate Spatial Adjacency of SpatialPolygonsDataFrame Objects

Description
This function utilizes the ‘spdep’ and ‘igraph’ packages to evaluate several measures of spatial connectivity.

Usage
```r
polygonAdjacency(x, v = "MUSYM")
```
polygonAdjacency

Arguments

x a SpatialPolygonsDataFrame object

v name of the field in the attribute table to use when searching for 'common lines', see details

Details

Pending.

Value

A list object containing:

commonLines An integer vector of feature IDs, that share a common boundary and attribute v. Sometimes referred to as "common soil lines".

adjMat A weighted adjacency matrix

Author(s)

D.E. Beaudette

Examples

## Not run:
library(spdep)
library(rgdal)
library(igraph)
library(sharpsht)

# load some data
x <- readOGR(dsn='FG_CA630_OFFICIAL.gdb', layer='ca630_a', stringsAsFactors=FALSE)

# compute spatial adjacency summary
res <- polygonAdjacency(x)

# graphical check: slow for large number of features
plot(x)
plot(x[res$commonLines, ], col='red', add=TRUE)

# save to SHP
writeOGR(x[res$commonLines, ], dsn='.',
layer='common-soil-lines', driver='ESRI Shapefile',
overwrite_layer=TRUE)

# plot spatial adjacency information
par(mar=c(0,0,0,0))
plotSoilRelationGraph(res$adjMat,
vertex.scaling.factor = 1)
plotSoilRelationGraph(res$adjMat, spanning.tree='max',
edge.scaling.factor=0.1, vertex.scaling.factor=1)
Description
Sample by polygon, according to a specified sampling density

Usage
```r
sample.by.poly(p, n.pts.per.ac = 1, min.samples = 5,
sampling.type = "hexagonal", iterations = 10, ...)
```

Arguments
- `p` a Polygon object, with coordinates from a projected CRS
- `n.pts.per.ac` requested sampling density in points per acre (results will be close)
- `min.samples` minimum requested samples / polygon
- `sampling.type` sampling type, see `spsample`
- `iterations` number of tries that `spsample` will attempt
- `...` additional arguments to `spsample`

Details
This function is typically used within the context of a loop or similar function that iterates over a list of Polygons associated with a `SpatialPolygons` object. See examples.

Value
A `SpatialPoints` object.

Note
This function expects that the Polygon object has coordinates associated with a projected CRS—e.g. units of meters.

Author(s)
D.E. Beaudette
Soil Taxonomy Dendrogram

Soil Taxonomy Dendrogram

Description

Plot a dendrogram based on the first 4 levels of Soil Taxonomy, with soil profiles hanging below. A dissimilarity matrix is computed using Gower’s distance metric for nominal-scale variables, based on order, sub order, great group, and subgroup level taxa. See the Details and Examples sections below for more information.

Usage

SoilTaxonomyDendrogram(spc, name = "hzname", max.depth = 150,
  n.depth.ticks = 6, scaling.factor = 0.015, cex.names = 0.75,
  cex.id = 0.75, axis.line.offset = -4, width = 0.1, y.offset = 0.5,
  cex.taxon.labels = 0.66)

Arguments

  spc         a SoilProfileCollection object, see details
  name        column name containing horizon names
  max.depth   depth at which profiles are truncated for plotting
  n.depth.ticks suggested number of ticks on the depth axis
  scaling.factor scaling factor used to convert depth units into plotting units
  cex.names   character scaling for horizon names
  cex.id      character scaling for profile IDs
  axis.line.offset horizontal offset for depth axis
  width       width of profiles
  y.offset    vertical offset between dendrogram and profiles
  cex.taxon.labels character scaling for taxonomic information

Details

This function looks for specific site-level attributes named: soilorder, suborder, greatgroup, and subgroup.

Value

An invisibly-returned list containing:

  dist  pair-wise dissimilarity matrix
Author(s)

D.E. Beaudette

References


Examples

```r
# Not run:
library(soilDB)

# soils of interest
s.list <- c('musick', 'cecil', 'drummer', 'amador', 'pentz', 'reiff',
'san joaquin','montpellier','grangeville','pollasky','ramona')

# fetch and convert data into an SPC
h <- fetchOSD(s.list)

# plot dendrogram + profiles
SoilTaxonomyDendrogram(h)

# again, this time save the pair-wise dissimilarity matrix
# note that there isn’t a lot of discrimination between soils
(d <- SoilTaxonomyDendrogram(h))

# End(Not run)
```
Index

*Topic datasets
    amador, 2
    CDEC.snow.courses, 4
*Topic hplots
    aspect.plot, 3
    diagnosticPropertyPlot, 10
    dueling.dendrograms, 13
    plotAvailWater, 15
    plotTransect, 19
    SoilTaxonomyDendrogram, 23
*Topic hplot
    plotSoilRelationGraph, 16
*Topic manip
    component.adj.matrix, 9
    dist.along.grad, 12
    hillslope.probability, 14
    polygonAdjacency, 20
    sample.by.poly, 22

amador, 2
aspect.plot, 3
CDEC.snow.courses, 4
CDECquery, 5
CDECSnowQuery, 6, 8
component.adj.matrix, 9
diagnosticPropertyPlot, 10
diagnosticPropertyPlot2
    (diagnosticPropertyPlot), 10
dist.along.grad, 12
dueling.dendrograms, 13
hillslope.probability, 14
plotAvailWater, 15
plotSoilRelationGraph, 16
plotTransect, 12, 19
polygonAdjacency, 20
sample.by.poly, 22

sharpshootR (sharpshootR-package), 2
sharpshootR-package, 2
SoilTaxonomyDendrogram, 23