Appendix A. R-Package Documentation

Package ‘ObsNetwork’
April 30, 2013

Version  0.1-6
Date     2013-04-30
Title    Optimization of Observation Networks
Author   Jason C. Fisher
Maintainer Fisher, Jason <jfisher@usgs.gov>
Depends  R (>= 3.0.0), tcltk, raster, sp, rgdal, gstat, GA
Description This package evaluates and optimizes long-term monitoring networks
using a kriging-based genetic algorithm methodology.
License  file LICENSE
Copyright This software is in the public domain because it contains
    materials that originally came from the United States
    Geological Survey, an agency of the United States Department of
    Interior. For more information, see the official USGS copyright policy at
    http://www.usgs.gov/visual-id/credit_usgs.html#copyright
URL     https://github.com/jfisher-usgs/ObsNetwork
BugReports https://github.com/jfisher-usgs/ObsNetwork/issues
ByteCompile yes

R topics documented:

ESRP_Boundary  ............................................................. 2
ESRP_Lakes ................................................................. 3
ESRP_NED ................................................................. 3
ESRP_NWIS ................................................................. 5
ESRP_Rivers ............................................................... 6
INL_Boundary ............................................................... 6
OpenGraphicsDevice ....................................................... 7
OptimizeNetwork .......................................................... 8
PlotBubble ................................................................. 11
PlotRaster ................................................................. 13
ReadNWISData ............................................................ 15
RunCrossValidation ....................................................... 17
WriteGAResults ............................................................ 19
ESRP_Boundary

Outlines of the ESRP and Buttes

Description
An outline of the generalized boundary of the eastern Snake River Plain (ESRP), East Butte, Middle Butte, and Big Southern Butte.

Usage
ESRP_Boundary

Format
An object of SpatialPolygonsDataFrame-class containing 4 Polygons. Three of the polygons represent buttes and are set as holes in the much larger aquifer polygon. Geographic coordinates are in units of decimal degrees, and in conformance with the North American Datum of 1983 (NAD 83).

Source
Idaho Department of Water Resources (IDWR).

Examples

data(ESRP_Boundary)
plot(ESRP_Boundary)
summary(ESRP_Boundary)

# Commands used to construct this data object:
dsn <- system.file("extdata/ESRP_Boundary", package = "ObsNetwork")
obj <- readOGR(dsn = dsn, layer = basename(dsn))
identical(obj, ESRP_Boundary)

# Polygon slots
p <- sapply(slot(obj, "polygons"), function(i) slot(i, "Polygons"))

# Generalized boundary of the ESRP aquifer
plot(SpatialPolygons(list(Polygons(list(p[[1]]), 1))))

# Generalized boundary of the East Butte
plot(SpatialPolygons(list(Polygons(list(p[[2]]), 1))), add = TRUE)

# Generalized boundary of the Middle Butte
plot(SpatialPolygons(list(Polygons(list(p[[3]]), 1))), add = TRUE)

# Generalized boundary of the Big Southern Butte
plot(SpatialPolygons(list(Polygons(list(p[[4]]), 1))), add = TRUE)
ESRP_Lakes

**Outlines of the ESRP Lakes**

**Description**

Major lakes of the eastern Snake River Plain (ESRP) and surrounding areas.

**Usage**

ESRP_Lakes

**Format**

An object of SpatialPolygonsDataFrame-class containing a set of Polygons. Geographic coordinates are in units of decimal degrees, and in conformance with the North American Datum of 1983 (NAD 83).

**Source**

Idaho Department of Water Resources (IDWR).

**Examples**

data(ESRP_Lakes)
plot(ESRP_Lakes)

ESRP_NED

**Topographic Information on the ESRP**

**Description**

The eastern Snake River Plain (ESRP) is a geologic feature located in the state of Idaho. This data set gives topographic information for the ESRP and vicinity on a 500 by 500 meter grid. The west, east, north, and south bounding coordinates are about -115.4, -111.4, 44.5, and 42.2 decimal degrees, respectively.

**Usage**

ESRP_NED

**Format**

An object of SpatialGridDataFrame-class with 323,724 points (coordinates) and a single data attribute, var2, the land-surface elevation. Geographic coordinates are in units of meters, in conformance with the North American Datum of 1927 (NAD 27), and placed in a Albers Equal-Area Conic projection; standard parallels 42.83, 44.16; central meridian -113.00, false easting 200,000 meters; false northing 0.
Source

The National Elevation Dataset (NED) 1-arc-second raster. Dataset resampled to 10 arc-seconds using a cubic convolution assignment. All other data processing is shown in ‘Examples’ section.

References


Examples

data(ESRP_NED)
image(ESRP_NED)
summary(ESRP_NED)

# Commands used to construct this data object:
f <- system.file("extdata/ESRP_NED.tif", package = "ObsNetwork")
grd <- rgdal::readGDAL(f, band = 1)
names(grd) <- "var2"
summary(grd)
grd.attr <- as.data.frame(slot(grd, "grid"))
summary(grd.attr)

# Transform coordinates to meters; Albers Equal-Area Conic projection; NAD27
projargs <- paste("+proj=aea +lat_1=42.83333333333333 +lat_2=44.16666666666666", "+lat_0=41.5 +lon_0=-113 +x_0=200000 +y_0=0 +ellps=clrk66", "+datum=NAD27 +units=m +no_defs")
pts <- suppressWarnings(spTransform(grd, CRS(projargs)))
class(pts)

# Resample data using reduced grid size of 500 by 500 meters
dx <- 500
dy <- 500
xlim <- c(10000, 328000)
ylim <- c(81200, 335700)
cellcentre.offset <- c(min(xlim) + dx / 2, min(ylim) + dy / 2)
cellsizex <- c(dx, dy)
cellsidy <- c(diff(range(xlim)) / dx, diff(range(ylim)) / dy)
newdata <- GridTopology(cellcentre.offset = cellcentre.offset, 
cellsizex = cellsizex, cellsidy = cellsidy)
newdata <- SpatialGrid(newdata, proj4string = CRS(projargs))
coordnames(newdata) <- c("x", "y")

# Inverse distance weighting interpolation; computationally demanding
obj <- gstat::idw(var2 ~ 1, pts, newdata, idp = 2.0, maxdist = 250)
obj$var1.var <- NULL
names(obj) <- "var2"

# Compare with available data set
identical(obj, ESRP_NED)
Description

A summary of water-level elevation measurements from wells located in the eastern Snake River Plain (ESRP) aquifer, Idaho.

Usage

ESRP_NWIS

Format

An object of `SpatialPointsDataFrame-class` with 335 points (coordinates). Geographic coordinates are in units of decimal degrees, and in conformance with the North American Datum of 1983 (NAD 83). This data set has the following variables:

- **site.no** Unique numerical identifier for each well site, for example 435339112444601.
- **var1.acy** Mean measurement accuracy of water-level elevations, in meters, for calendar year 2008.
- **var1.sd** Standard deviation of water-level elevations, in meters, for entire period of record; duration varies for each well site.
- **var2** Land-surface reference point elevation, in meters above the NAVD 88.
- **map.no** Numeric identifier used to locate well sites on map.
- **network.nm** Identifier for water-level monitoring network. For this data set "State" is used to identify wells in the 2008 Federal-State Cooperative water-level monitoring network (166 wells), and "INL" for wells in the 2008 U.S. Geological Survey-Idaho National Laboratory water-level monitoring network (171 wells). Wells coded as "State, INL" belong to both monitoring networks.
- **nrec.por** Number of records in the entire period of record.
- **nrec** Number of records in calendar year 2008.
- **alt.acy.va** Accuracy of land-surface reference point elevation, in meters.
- **lev.acy.va** Mean water-level measurement accuracy, in meters, for calendar year 2008.
- **coord.acy.va** Latitude/longitude coordinate accuracy, in arc-seconds.
- **coord.meth.cd** Method used to determine horizontal datum.
- **alt.meth.cd** Method used to determine land-surface datum.
- **lev.meth.cd** All methods used to determine water levels; comma separated.
- **site.nm** Local well identifier, for example "07N 31E 34BDD1 USGS 25".

Source

The National Water Information System (NWIS); accessed January 2013 using RNWIS (version 0.1-8). The software VERTCON (version 2.1) was used to convert National Geodetic Vertical Datum of 1929 (NGVD 29) heights to NAVD 88. The final data table was constructed using the `ReadNWISData` pre-processing function.
Examples

```r
data(ESRP_NWIS)
plot(coordinates(ESRP_NWIS))
str(ESRP_NWIS)
```

---

**ESRP_Rivers**  
*Traces of the ESRP Rivers*

**Description**

Major rivers of the eastern Snake River Plain (ESRP) and surrounding areas.

**Usage**

ESRP_Rivers

**Format**

An object of `SpatialLinesDataFrame-class` containing a set of `Lines`. Geographic coordinates are in units of decimal degrees, and in conformance with the North American Datum of 1983 (NAD 83).

**Source**

Idaho Department of Water Resources (IDWR).

Examples

```r
data(ESRP_Rivers)
plot(ESRP_Rivers)
```

---

**INL_Boundary**  
*Outline of the Idaho National Laboratory*

**Description**

The political boundary of the Idaho National Laboratory (INL).

**Usage**

INL_Boundary

**Format**

An object of `SpatialPolygonsDataFrame-class` containing 1 `Polygon`. Geographic coordinates are in units of decimal degrees, and in conformance with the North American Datum of 1983 (NAD 83).

**Source**

OpenGraphicsDevice

Examples

data(INL_Boundary)
plot(INL_Boundary)

OpenGraphicsDevice  Open Graphics Device

Description

This function starts the device driver for producing graphics.

Usage

OpenGraphicsDevice(file, type = "windows", w = 7, h = 7, p = 12,
       res = 300, win.title = "Save As")

Arguments

file character; the path of a file for writing (optional).
type character; the name of a graphics driver, either "postscript", "pdf", "png", or the
default "windows".
w, h numeric; the (nominal) width and height of the canvas of the plotting window in
inches. Default is 7 and 7, respectively.
p numeric; the default point size of plotted text, its default is 12.
res numeric; the nominal resolution in points per inch (ppi) which will be recorded
in the Portable Network Graphics (PNG) file, its default is 300 ppi to set the size
of text and line widths.
win.title character; a string to display as the title of the dialog box, only used if file is
not specified.

Details

A "postscript" graphics device produces an Encapsulated PostScript (eps) file and is recom-
mended for figures of publication quality.

Value

A plot device is opened.

Author(s)

J.C. Fisher

See Also

postscript, pdf, png, windows
OptimizeNetwork

Examples

OpenGraphicsDevice(type = "windows")
plot(1:3)
dev.off()

OpenGraphicsDevice(type = "pdf")
plot(1:3)
dev.off()

OptimizeNetwork

Description

Determine sites to exclude from an existing observation network because they provide little or no beneficial added information. A kriging-based genetic algorithm (GA) is used to solve the multi-objective optimization problem.

Usage

OptimizeNetwork(pts, grd, ply, network.nm, nsites, model, formula, nmax = Inf, xlim = bbox(grd)[1, ], ylim = bbox(grd)[2, ], grd.fact = 1, obj.weights = c(1, 1, 1, 1), penalty.constant = 1E6, maxabort = 10, popSize = 50, pcrossover = 0.8, pmutation = 0.1, elitism = base::max(1, round(popSize * 0.05)), maxiter = 100, run = maxiter, suggestions = NULL, ...)

Arguments

pts SpatialPointsDataFrame; data at observation sites. Required data frame variables include: site.no, a unique site number; var1, the dependent variable (such as, the water-level elevation); var1.acy, the mean measurement error of the dependent variable; var1.sd, the standard deviation of the dependent variable. An optional network.nm variable may be included to identify a sites observation network(s). Sites belonging to multiple networks are specified using comma separation. Note that duplicate site numbers and (or) spatial coordinates are not permitted.

grd SpatialGridDataFrame; interpolation grid. For kriging with external drift (KED) a data frame variable var2, the independent variable (such as, land-surface elevation), is required.

ply SpatialPolygonsDataFrame; a polygon used to crop the raster grid (optional).

network.nm character; vector of observation network names. Only sites belonging to this network will be included in the analysis; this argument is optional, in its absence all sites are assumed to belong to a single network (that is, all sites are used).

nsites integer; number of sites to remove from the observation network.

model variogramModel; variogram model of dependent variable defined by a call to vgm.

formula formula; defines the dependent variable as a linear model of the independent variables.
nmax numeric; for local kriging, the number of nearest sites that should be used for a kriging prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, all sites are used.

xlim numeric; vector of length 2 giving left and right limits for the x-axis, used to crop the interpolation grid.

ylim numeric; vector of length 2 giving lower and upper limits for the y-axis, used to crop the interpolation grid.

grd.fact integer; aggregation factor for the interpolation grid, grd.

obj.weights numeric; vector of length 4 giving the weights for each objective in the multi-objective optimization problem, see ‘details’ section below.

penalty.constant numeric; constant in the penalty function, its value needs to be greater than the largest possible fitness value.

maxabort integer; maximum number of times an invalid child chromosome can be aborted during crossover.

popSize integer; population size.

crossover numeric; probability of crossover between pairs of chromosomes.

pmutation numeric; probability of mutation in a parent chromosome.

elitism integer; number of chromosomes to survive into the next generation. By default is about 5 percent of the population size.

maxiter integer; maximum number of iterations to run before the GA search is halted.

run integer; number of consecutive generations without any improvement in the “best” fitness value before the GA is stopped.

suggestions matrix; initial population.

... additional arguments to be passed to ga.

Details

A solution to the multi-objective optimization problem is found by minimizing the aggregate objective function, the weighted linear sum of 4 objectives. The objectives are given as:

1. Mean standard error at points in interpolation grid.
2. Root-mean-square error, difference between predicted and measured values, at removed sites.
3. Mean standard deviation, variability of measurement over time, at removed sites.
4. Mean measurement error, at remaining sites.

The “best” solution found will depend on the relative values of the weights specified in obj.weights. For example, if a higher weight is specified for the mean standard error, the solution will be one that favors a smaller mean standard error over a small root-mean squared error, mean standard deviation, and mean measurement error. Setting a weight equal to zero will remove an objective from the multi-objective function.

Spatial data is transformed to the map projection and datum of the raster data set in grd. The optimization problem is solved using a GA with integer chromosomes; site identifiers are represented as binary strings using Gray encoding. The initial population is randomly generated with valid chromosomes; that is, sites are not repeated within a single chromosome. The GA uses linear-rank selection, single-point crossover, and uniform random mutation.
Value

Returns a list with components:

- **call**: character; function call with all specified arguments
- **pts.rm**: SpatialPointsDataFrame; a subset of pts with row length equal to nsites. Includes data records for those sites identified by the GA for removal from the observation network(s).
- **is.net**: logical; vector of length equal to the number of rows in pts specifying sites belonging to the reduced network.
- **is.rm**: logical; vector of length equal to the number of rows in pts specifying sites to remove from the network.
- **obj.values**: matrix; objective values at each iteration of the GA. This matrix has maxiter rows and 5 columns (that is, the 4 objective values and their sum).
- **niter**: integer; number of completed iterations.
- **nrep.ans**: integer; number of iterations the “best” solution was repeated.
- **proc.time**: proc_time; CPU time for running the GA, in seconds.
- **ncalls.penalty**: integer; vector giving the number of calls to the penalty function at each iteration of the GA.
- **kr**: SpatialGridDataFrame; a data frame containing the coordinates of grd cropped to the axis limits and polygon. Data attributes based on block kriging of the reduced network include: predictions, `var1.pred`; prediction variances, `var1.var`; and prediction standard errors, `var1.se`. Differences between the original and reduced network predictions are specified in the `var1.diff` attribute.
- **rmsd**: numeric; root-mean-square-deviation between the kriged surfaces using the original and reduced networks.
- **local.error**: numeric; percent local error between the kriged surfaces using the original and reduced networks.
- **obj.space**: matrix; range of objective values in solution space.
- **ga.ans**: ga; returned value of ga.
- **start.time**: POSIXct; system time at start of GA run.
- **diff.time**: numeric; vector of time differences since start of GA run, specified for each iteration in the GA, in hours.

The status of the objective values after each iteration of the GA is plotted.

Author(s)

J.C. Fisher

References


See Also

WriteGAResults, krige, ga
Examples

```r
data(ESRP_NED)
data(ESRP_NWIS)
data(ESRP_Boundary)

# Formula and variogram
fo <- var1 ~ x + y
model <- vgm(psill = 1948.533, model = "Sph", nugget = 0, range = 153891.038)

# Optimize combined "State" and "INL" networks
xlim <- c(10000, 328000)
ylim <- c(812, 3357)
ans <- OptimizeNetwork(ESRP_NWIS, ESRP_NED, ESRP_Boundary,
                       network.nm = c("State", "INL"), nsites = 20,
                       model = model, formula = fo, grd.fact = 5,
                       obj.weights = c(100, 1, 1, 1), popSize = 20,
                       maxiter = 3)
PlotRaster(ans$kr, "var1.pred", ESRP_NWIS,
           pal = colorRampPalette(c("#F2311", "#FFDFA", "#17FC9")),
           main = "Predictions", net.idxs = which(ans$is.net),
           rm.idxs = which(ans$is.rm), xlim = xlim, ylim = ylim)
PlotRaster(ans$kr, "var1.se", ESRP_NWIS,
           pal = terrain.colors, main = "Standard errors",
           net.idxs = which(ans$is.net), rm.idxs = which(ans$is.rm),
           xlim = xlim, ylim = ylim)

# Optimize "INL" network
xlim <- c(178000, 257500)
ylim <- c(2, 272)
ans <- OptimizeNetwork(ESRP_NWIS, ESRP_NED, ESRP_Boundary,
                       network.nm = "INL", nsites = 20, model = model,
                       formula = fo, xlim = xlim, ylim = ylim, grd.fact = 5,
                       obj.weights = c(100, 1, 1, 1), maxiter = 3)
PlotRaster(ans$kr, "var1.diff", ESRP_NWIS, pal = jet.colors,
           main = "Prediction Differences", net.idxs = which(ans$is.net),
           rm.idxs = which(ans$is.rm), xlim = xlim, ylim = ylim)

# Restart GA using previous "best" solution
ans <- OptimizeNetwork(ESRP_NWIS, ESRP_NED, ESRP_Boundary,
                       network.nm = "INL", nsites = 20, model = model,
                       formula = fo, xlim = xlim, ylim = ylim, grd.fact = 5,
                       obj.weights = c(100, 1, 1, 1), maxiter = 3, suggestions = ans$ga.ans@population)
```

---

**PlotBubble**

Create a Bubble Plot of Spatial Data

**Description**

Bubble plot for spatial data with attributes. This is a wrapper around `bubble` in the `sp` package.

**Usage**

`PlotBubble(pts, zcol, ply, xlim = bbox(ply)[1, ], ylim = bbox(ply)[2, ],`
main = "", gr.type = "windows", gr.file = NULL,
projargs = proj4string(pts))

Arguments

- **pts**  
  SpatialPointsDataFrame; data at observation sites.

- **zcol**  
  character; z-variable column name in attribute table of pts.

- **ply**  
  SpatialPolygonsDataFrame; polygon to include in layout.

- **xlim**  
  numeric; vector of length 2 giving left and right limits for the x-axis.

- **ylim**  
  numeric; vector of length 2 giving lower and upper limits for the y-axis.

- **main**  
  character; main plot title to be placed on top.

- **gr.type**  
  character; the name of a graphics driver, either "postscript", "pdf", "png", or the default "windows".

- **gr.file**  
  character; the path of a file for writing the graphics, only used if gr.type is not "windows".

- **projargs**  
  character; projection arguments; the arguments must be entered exactly as in the PROJ.4 documentation.

Value

Returns (or plots) the bubble plot. Values plotted in the key include the five quantiles: minimum, 25th percentile, median, 75th percentile, and maximum.

Author(s)

J.C. Fisher

See Also

bubble, OpenGraphicsDevice

Examples

data(ESRP_NWIS)
data(ESRP_Boundary)
data(ESRP_NED)

PlotBubble(ESRP_NWIS, "var1.acy", ESRP_Boundary,
main = "Measurment error")

PlotBubble(ESRP_NWIS, "var1.sd", ESRP_Boundary,
xlim = c(178000, 257500), ylim = c(202000, 272000),
main = "Standard deviation", projargs = proj4string(ESRP_NED))
Create a Filled Contour Plot of Spatial Data

Description

Raster plot for spatial data with attributes.

Usage

PlotRaster(grd, zcol, pts, ply, net.idxs, rm.idxs, xlim, ylim, at,
           pal = heat.colors, contour = FALSE, label.contours = FALSE,
           label.pts = FALSE, main = "", gr.type = "windows",
           gr.file = NULL, width = 7, height = NA, lo = list(),
           ll.lines = FALSE)

Arguments

- **grd**: SpatialGridDataFrame; raster data.
- **zcol**: character; z-variable column name or number in attribute table of `grd`.
- **pts**: SpatialPointsDataFrame; point coordinates to include in layout.
- **ply**: SpatialPolygonsDataFrame; a polygon defining the spatial domain of the raster data; data outside this domain is excluded. Polygon is included in the layout.
- **net.idxs**: integer; vector of row indexes in `pts`, point coordinates at these sites are drawn as circle symbols with white backgrounds. All other circle symbols are drawn with gray backgrounds.
- **rm.idxs**: integer; vector of row indexes in `pts`, point symbols at these sites are drawn as crosses.
- **xlim**: numeric; vector of length 2 giving left and right limits for the x-axis.
- **ylim**: numeric; vector of length 2 giving lower and upper limits for the y-axis.
- **at**: numeric; vector giving breakpoints along the range of z (including upper and lower limits).
- **pal**: function; a color palette to be used to assign colors in the plot.
- **contour**: logical; add contour lines to plot, default is FALSE.
- **label.contours**: logical; label contour lines, default is FALSE.
- **label.pts**: character or logical; attribute in `pts` specifying the labels to place at point coordinates. If TRUE, points are labeled with index numbers.
- **main**: character; main plot title to be placed on top.
- **gr.type**: character; the name of a graphics driver, either "postscript", "pdf", "png", or the default "windows".
- **gr.file**: character; the path of a file for writing the graphics, only used if `gr.type` is not "windows".
- **width, height**: numeric; the (nominal) width and height of the canvas of the plotting window in inches.
- **lo**: list; a list with more layout items, see `sp.layout` argument in `spplot`.
- **ll.lines**: logical; plot long-lat grid over projected data, default is FALSE.
Details

Spatial data is transformed to the map projection and datum of the raster data set, grd.

Value

Returns (or plots) the raster plot.

Author(s)

J.C. Fisher

See Also

spplot

Examples

data(ESRP_NED)
data(ESRP_NWIS)
data(ESRP_Boundary)
data(ESRP_Lakes)
data(ESRP_Rivers)
data(INL_Boundary)

# Plot topography
PlotRaster(ESRP_NED, "var2")

# Set axis limits, add points and long-lat grid to plot layout
pts <- ESRP_NWIS[ESRP_NWIS$network.nm == "State", ]
xlim <- c(100000, 3280000)
ylim <- c(812000, 3357000)
PlotRaster(ESRP_NED, "var2", pts, ESRP_Boundary, xlim = xlim, ylim = ylim,
pal = terrain.colors, label.pts = "map.no", ll.lines = TRUE)

# Plot hill shade, add lakes, river, and INL boundary to the plot layout
slp <- terrain(raster(ESRP_NED), opt = "slope")
asp <- terrain(raster(ESRP_NED), opt = "aspect")
grd <- as(hillShade(slp, asp, 40, 270), "SpatialGridDataFrame")
zlim <- range(grd[[1]], na.rm = TRUE)
at <- seq(zlim[1], zlim[2], length.out = 50)
pal <- function(n) grey(0:5/5)[1:n]
lo <- list()
lo[[1]] <- list("sp.polygons", ESRP_Lakes, col = "#1B70E0", fill = "#BAE4E5",
              first = FALSE)
lo[[2]] <- list("sp.lines", ESRP_Rivers, col = "#1B70E0", first = FALSE)
lo[[3]] <- list("sp.polygons", INL_Boundary, col = "#000000", first = FALSE)
PlotRaster(grd, 1, xlim = xlim, ylim = ylim, at = at, pal = pal, lo = lo)

# Set new axis limits
pts <- ESRP_NWIS[ESRP_NWIS$network.nm == "INL", ]
xlim <- c(178000, 2575000)
ylim <- c(202000, 2720000)
PlotRaster(ESRP_NED, "var2", pts, ESRP_Boundary, xlim = xlim, ylim = ylim,
pal = terrain.colors, ll.lines = TRUE, contour = TRUE)
ReadNWISData

Description

This function reads water-level measurements exported from the U.S. Geological Survey (USGS) National Water Information System (NWIS) database and summarizes this data for a specified duration of time.

Usage

ReadNWISData(file, dt.lim, dt.fmt = "%Y-%m-%d %H:%M", sep = "\t")

Arguments

- **file**: character; a path to the file to be opened.
- **dt.lim**: character; vector of length 2 giving the starting and ending date-time limits. Measurements recorded during this time period are used to calculate the median water-level elevation and mean measurement accuracy. Defaults to the entire period of record.
- **dt.fmt**: character; a date-time format as used by `strptime`. Date-time values specified in `dt.lim` must be provided in this format.
- **sep**: character; the field separator string. Values on each line of the file are separated by this string, its default is the tab separator.

Format

The input text file contains the names of the variables in its first line. Each subsequent line corresponds to a single water-level measurement. Measurement variables (columns in the data table) include:

- **SITE_NO**: Site number.
- **DEC_LONG_VA**: Decimal longitude.
- **DEC_LAT_VA**: Decimal latitude.
- **COORD_ACY_CD**: Latitude/longitude coordinate accuracy code (optional).
- **COORD_DATUM_CD**: Latitude/longitude (horizontal) coordinate datum code (optional).
- **COORD_METH_CD**: Code indicating the method used to determine horizontal datum (optional).
- **ALT_VA**: Gage or land-surface datum, in feet.
- **ALT_ACY_VA**: Accuracy of land-surface datum, in feet.
- **ALT_DATUM_CD**: Code indicating the geodetic or local vertical datum of the elevation datum elevation component (optional); for example, "NGVD29" or "NAVD88".
- **ALT_METH_CD**: Code indicating the method used to determine the elevation of land-surface datum (optional).
- **HOLE_DEPTH_VA**: Borehole depth, in feet (optional).
- **WELL_DEPTH_VA**: Well depth, in feet (optional).
- **LEV_DT**: Date/time when water-level measurement was recorded. The expected conversion specification format is "%Y-%m-%d %H:%M:%S"; see `strftime` for details.
**LEV VA** Water-level measurement referenced to land-surface datum (that is, depth below land surface), in feet.

**LEV ACY CD** Water-level measurement accuracy code.

**LEV METH CD** Code indicating how the water level was measured (optional).

**STATION NM** Site name (optional).

**NETWORK NM** Water-level monitoring network name (optional). Sites belonging to multiple networks should be comma separated. This variable is not included in the NWIS database and it is tasked to the user to populate this field.

**Details**

This function is a pre-processor for `ObsNetwork`. Groundwater data can be downloaded at NWISWeb.

**Value**

Returns an object of `SpatialPointsDataFrame-class` with components:

- `site.no` numeric; unique identifier for site.
- `var1` numeric; the median value of the water-level elevation, in meters, for the time period defined by `dt.lim`.
- `var1.acy` numeric; mean measurement accuracy of the water-level elevation, in meters, for the time period defined by `dt.lim`. This variable is calculated from the summation of `alt.acy.va` and `lev.acy.va` values.
- `var1.sd` numeric; standard deviation of the water-level elevation, in meters, for the entire period of record.
- `var2` numeric; land-surface datum, in meters.
- `map.no` integer; identifier (row index number) used to locate sites on map.
- `network.nm` character; monitoring network name.
- `nrec.por` integer; number of records in the entire period of record.
- `nrec` integer; number of records in the time period defined by `dt.lim`.
- `alt.acy.va` numeric; accuracy of land-surface datum, in meters.
- `lev.acy.va` numeric; mean water-level measurement accuracy, in meters, for the time period defined by `dt.lim`.
- `coord.acy.va` numeric; latitude/longitude coordinate accuracy, in arc-seconds.
- `coord.meth.cd` character; method used to determine horizontal datum.
- `alt.meth.cd` character; method used to determine elevation at land-surface datum.
- `lev.meth.cd` character; all methods used to determine water levels, comma separated.
- `site.nm` character; site name.

**Author(s)**

J.C. Fisher

**See Also**

`read.table`, `as.POSIXct`, `CRS`
Examples

# Read data from text file
file <- system.file("extdata/ESRP_NWIS.txt.gz", package = "ObsNetwork")
dt.lim <- c("2008-01-01 00:00", "2008-12-31 23:59") # 2008 calendar year
obj <- ReadNWISData(file, dt.lim)
str(obj)

# Compare with available data set
data(ESRP_NWIS)
identical(obj, ESRP_NWIS)

RunCrossValidation

Description

Cross validation function for kriging; a wrapper around the \texttt{krige.cv} function in the \texttt{gstat} package. This function may be used to identify negative aspects of a kriging model, such as the worst errors or the areas with consistent bias.

Usage

\begin{verbatim}
RunCrossValidation(formula, pts, model, nfold = nrow(pts), ..., 
    projargs = proj4string(pts))
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{formula} \hspace{1cm} \texttt{formula}; defines the dependent variable as a linear model of independent variables.
  \item \texttt{pts} \hspace{1cm} \texttt{SpatialPointsDataFrame}; data at observation sites.
  \item \texttt{model} \hspace{1cm} \texttt{variogramModel}; variogram model of dependent variable defined by a call to \texttt{vgm}.
  \item \texttt{nfold} \hspace{1cm} \texttt{numeric}; for local kriging, the number of nearest observations that should be used for a kriging prediction or simulation, where nearest is defined in terms of the space of the spatial locations. By default, all observations are used.
  \item \texttt{...} \hspace{1cm} \texttt{other arguments that will be passed to \texttt{predict.gstat}.}
  \item \texttt{projargs} \hspace{1cm} \texttt{character}; projection arguments; the arguments must be entered exactly as in the \texttt{PROJ.4} documentation.
\end{itemize}

Value

Returns a list with components:

\begin{itemize}
  \item \texttt{cv} \hspace{1cm} \texttt{SpatialPointsDataFrame}; attributes of prediction and prediction variance of cross validated data points, observed values, residuals, \texttt{zscore} (residual divided by kriging standard error), and fold.
  \item \texttt{me} \hspace{1cm} \texttt{numeric}; mean error, ideally 0.
  \item \texttt{mspe} \hspace{1cm} \texttt{numeric}; mean squared prediction error, ideally small.
  \item \texttt{cor.op} \hspace{1cm} \texttt{numeric}; correlation of observed and predicted, ideally 1.
  \item \texttt{cor.pr} \hspace{1cm} \texttt{numeric}; correlation of predicted and residual, ideally 0.
\end{itemize}
18

RunCrossValidation

Author(s)
J.C. Fisher

See Also

krige.cv

Examples

data(ESRP_NWIS)
data(ESRP_Boundary)
data(ESRP_NED)

# Set datum and projection
crs <- CRS(proj4string(ESRP_NED))
pts <- spTransform(ESRP_NWIS, crs)
ply <- spTransform(ESRP_Boundary, crs)

# Exclude grid points outside polygon
grd <- ESRP_NED
grd$var2 <- grd$var2 * sp::overlay(grd, ply)

# Set axis limits
xlim <- c(10000, 328000)
ylim <- c(812000, 335700)

# Kriging function with plotting routine
Krige <- function(fo, model, xlim, ylim, ...) {
  kr <- gstat::krige(formula = fo, locations = pts, newdata = grd,
                     model = model, ...)
  kr$var1.se <- sqrt(kr$var1.var)
  pal1 <- colorRampPalette(c("#F02311", "#F7FDFA", "#17FFC9"))
  pal2 <- terrain.colors
  PlotRaster(kr, "var1.pred", pts, xlim = xlim, ylim = ylim, pal = pal1,
            main = "Predictions", contour = TRUE)
  PlotRaster(kr, "var1.se", pts, xlim = xlim, ylim = ylim, pal = pal2,
            main = "Standard errors")
}

### Ordinary Kriging (OK):
fo <- var1 ~ 1
vg <- variogram(fo, pts)
plot(variogram(fo, pts, cloud = TRUE), pch = 3)
model <- fit.variogram(vg, vgm(model = "Lin", nugget = 0))
plot(vg, model, main = "Linear variogram model (var1 ~ 1)"

### Universal Kriging (UK), accounts for linear spatial trend:
fo <- var1 ~ x + y
lm.trend <- lm(fo, data = pts)
summary(lm.trend)

vg <- variogram(fo, pts, cutoff = 150000)
model <- fit.variogram(vg, vgm(psill = 1500, model = "Sph", range = 100000,
nugget = 0), fit.method = 1)
plot(vg, model, main = "Residual variogram model (var1 ~ x + y)"

Krige(fo, model, xlim, ylim)
cv <- RunCrossValidation(fo, pts, model = model)
cat(cv$me, cv$mspe, cv$cor.op, cv$cor.pr, "\n")

### Kriging with External Drift (KED); assumes var1 forms a subdued replica of var2:
fo <- var1 ~ var2
lm.trend <- lm(fo, data = pts)
summary(lm.trend)
plot(fo, pts, main = "Correlation plot and fitted regression model")
abline(lm.trend)

vg <- variogram(fo, pts)
model <- fit.variogram(vg, vgm(psill = 4500, model = "Sph", range = 10000,
nugget = 0), fit.method = 1)
plot(vg, model, main = "Residual variogram model (var1 ~ var2)"
Krige(fo, model, xlim, ylim)
cv <- RunCrossValidation(fo, pts, model = model)
cat(cv$me, cv$mspe, cv$cor.op, cv$cor.pr, "\n")

# KED in a local neighborhood:
cv <- RunCrossValidation(fo, pts, model = model, nmax = 50)
cat(cv$me, cv$mspe, cv$cor.op, cv$cor.pr, "\n")

---

**WriteGAResults**

**Write Results of Genetic Algorithm**

**Description**

Prints list components returned from the `OptimizeNetwork` function to a file.

**Usage**

`WriteGAResults(x, file)`

**Arguments**

- **x**
  - list; object returned from `OptimizeNetwork`.
- **file**
  - character; path of a file for writing (optional).

**Details**

Printed components of `ga` include: `call`; `obj.values`; `rmsd`; `local.error`; `nitors`; `nrep.ans`; `proc.time`; `start.time`; `obj.space`; `ncalls.penalty`; `pts.rm`; and `ga.ans@suggestion`.

**Value**

None (invisible NULL).

**Author(s)**

J.C. Fisher
See Also

OptimizeNetwork

Examples

## Not run: WriteGAResults(ans)
Index

*Topic IO
  WriteGAResults, 19

*Topic datasets
  ESRP_Boundary, 2
  ESRP_Lakes, 3
  ESRP_NED, 3
  ESRP_NWIS, 5
  ESRP_Rivers, 6
  INL_Boundary, 6

*Topic device
  OpenGraphicsDevice, 7

*Topic hplot
  PlotBubble, 11
  PlotRaster, 13

*Topic manip
  ReadNWISData, 15

*Topic misc
  RunCrossValidation, 17

*Topic optimize
  OptimizeNetwork, 8

as.POSIXct, 16

bubble, 11, 12

CRS, 16

ESRP_Boundary, 2
ESRP_Lakes, 3
ESRP_NED, 3
ESRP_NWIS, 5
ESRP_Rivers, 6

ga, 9, 10

INL_Boundary, 6

krige, 10
krige.cv, 17, 18

Lines, 6

OpenGraphicsDevice, 7, 12
OptimizeNetwork, 8, 19, 20

pdf, 7

PlotBubble, 11
PlotRaster, 13
png, 7
Polygon, 6
Polygons, 2, 3
postscript, 7
predict.gstat, 17

read.table, 16
ReadNWISData, 5, 15
RunCrossValidation, 17

spplot, 13, 14
strftime, 15
strptime, 15

vgm, 8, 17

windows, 7
WriteGAResults, 10, 19