Package ‘FIESTA’

August 10, 2022

Type Package

Title Forest Inventory Estimation and Analysis

Version 3.4.2

Date 2022-08-10

Description A research estimation tool for analysts that work with sample-based inventory data from the U.S. Department of Agriculture, Forest Service, Forest Inventory and Analysis (FIA) Program.

Depends R (>= 4.0)

Imports data.table, DBI, FIESTAutils (>= 1.1.3), grDevices, graphics, methods, rgdal, RSQLite, sf, sqldf, utils, largeList

Suggests knitr, raster, rmarkdown

VignetteBuilder knitr

License GPL-3

Copyright See file COPYRIGHTS for details.

LazyData true

SystemRequirements C++11

Encoding UTF-8

RoxygenNote 7.2.0

NeedsCompilation no

Author Tracey Frescino [aut], Gretchen Moisen [aut], Paul Patterson [aut], Chris Toney [aut], Grayson White [aut, cre] (<https://orcid.org/0000-0003-4993-2792>)

Maintainer Grayson White <graysonwhite13@gmail.com>

Repository CRAN

Date/Publication 2022-08-10 21:30:02 UTC
**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIESTA-package</td>
<td>4</td>
</tr>
<tr>
<td>datBarplot</td>
<td>4</td>
</tr>
<tr>
<td>datBarStacked</td>
<td>7</td>
</tr>
<tr>
<td>datFilter</td>
<td>11</td>
</tr>
<tr>
<td>datFreq</td>
<td>13</td>
</tr>
<tr>
<td>datLineplot</td>
<td>14</td>
</tr>
<tr>
<td>datLUTclass</td>
<td>17</td>
</tr>
<tr>
<td>datLUTnm</td>
<td>19</td>
</tr>
<tr>
<td>datPBplotchg</td>
<td>21</td>
</tr>
<tr>
<td>datPBpnt2pct</td>
<td>22</td>
</tr>
<tr>
<td>datPivot</td>
<td>23</td>
</tr>
<tr>
<td>datPlotcnt</td>
<td>24</td>
</tr>
<tr>
<td>datSumCond</td>
<td>26</td>
</tr>
<tr>
<td>datSumTree</td>
<td>28</td>
</tr>
<tr>
<td>datSumTreeDom</td>
<td>31</td>
</tr>
<tr>
<td>DBgetCSV</td>
<td>37</td>
</tr>
<tr>
<td>DBgetEvalid</td>
<td>38</td>
</tr>
<tr>
<td>DBgetPlots</td>
<td>41</td>
</tr>
<tr>
<td>DBgetSQLite</td>
<td>51</td>
</tr>
<tr>
<td>DBgetStrata</td>
<td>52</td>
</tr>
<tr>
<td>DBgetXY</td>
<td>55</td>
</tr>
<tr>
<td>DBqryCSV</td>
<td>58</td>
</tr>
<tr>
<td>DEFAULT_NODATA</td>
<td>59</td>
</tr>
<tr>
<td>ecomap</td>
<td>59</td>
</tr>
<tr>
<td>GDT_NAMES</td>
<td>60</td>
</tr>
<tr>
<td>kindcd3old</td>
<td>60</td>
</tr>
<tr>
<td>modGBarea</td>
<td>61</td>
</tr>
<tr>
<td>modGBp2veg</td>
<td>66</td>
</tr>
<tr>
<td>modGBpop</td>
<td>71</td>
</tr>
<tr>
<td>modGBratio</td>
<td>76</td>
</tr>
<tr>
<td>modGBtree</td>
<td>83</td>
</tr>
<tr>
<td>modMAarea</td>
<td>89</td>
</tr>
<tr>
<td>modMApop</td>
<td>94</td>
</tr>
<tr>
<td>modMAtree</td>
<td>99</td>
</tr>
<tr>
<td>modPB</td>
<td>105</td>
</tr>
<tr>
<td>modPBpop</td>
<td>111</td>
</tr>
<tr>
<td>modSAarea</td>
<td>117</td>
</tr>
<tr>
<td>modSApop</td>
<td>121</td>
</tr>
<tr>
<td>modSAtree</td>
<td>125</td>
</tr>
<tr>
<td>popFilters</td>
<td>130</td>
</tr>
<tr>
<td>popTableIDs</td>
<td>131</td>
</tr>
<tr>
<td>popTables</td>
<td>132</td>
</tr>
<tr>
<td>ref_ALP</td>
<td>134</td>
</tr>
<tr>
<td>ref_codes</td>
<td>135</td>
</tr>
<tr>
<td>ref_cond</td>
<td>135</td>
</tr>
<tr>
<td>ref_conversion</td>
<td>136</td>
</tr>
</tbody>
</table>
FIESTA-package

FIESTA - Forest Inventory Estimation for Analysis

Description

FIESTA is a research estimation tool for analysts that work with sample-based inventory data from the U.S. Department of Agriculture, Forest Service, Forest Inventory and Analysis (FIA) Program.

Details

FIESTA can generate FIA's traditional state-wide estimates while also accommodate: unique population boundaries, different evaluation time periods, customized stratification schemes, non-standard variance equations, integration of multi-scale remotely-sensed data and other auxiliary information, and interaction with other modeling and estimation tools from CRAN's library of packages.

FIESTA contains a collection of functions that can query FIA databases, summarize and compile plot and spatial data, and generate estimates with associated sampling errors.

Author(s)

Tracey S. Frescino Maintainer: Tracey S. Frescino

References


datBarplot

Data - Generates frequency barplot.

Description

Generate a barplot of from a frequency data frame.
datBarplot

Usage

datBarplot(
  x,
  xvar = NULL,
  yvar = "FREQ",
  grpvar = NULL,
  errbars = FALSE,
  x.order = NULL,
  sevar = NULL,
  psevar = NULL,
  device.type = "dev.new",
  jpeg.res = 300,
  device.height = 5,
  device.width = 8,
  horiz = FALSE,
  toplabelvar = NULL,
  ylim = NULL,
  divideby = NULL,
  ylabel = NULL,
  xlabel = NULL,
  mar = NULL,
  addlegend = FALSE,
  main = NULL,
  cex.main = 1,
  cex.label = 1,
  cex.names = 0.8,
  las.xnames = 0,
  las.ynames = 1,
  savedata = FALSE,
  outfolder = NULL,
  outfn = NULL,
  outfn.pre = NULL,
  outfn.date = TRUE,
  overwrite = FALSE,
  ...
)

Arguments

x          Data frame or comma-delimited file (*.csv) - a frequency table.
xvar       String. Name of X variable.
yvar       String. Name of the y variable (e.g., FREQ).
grpvar      String. Name of the variable for grouping.
errbars     Logical. If TRUE, error bars are added to bar plot (sevar or psevar must also be populated).
x.order     String or Vector. Define order of xvar based on y values: descending ("DESC") or ascending ("ASC") or vector of row numbers. If NULL, the order of the input
datBarplot

table is used.

sevar String. Name of the variable with standard error values.
psevar String. Name of the variable with percent standard error.
device.type String. Type(s) of device for plotting ("dev.new", "jpg", "pdf").
jpeg.res Integer. Resolution for jpeg image.
device.height Integer. Height (in inches) of barplot, if writing to file.
device.width Integer. Width (in inches) of barplot, if writing to file.
horz Logical. If TRUE, bars are drawn horizontally with first bar at the bottom. If FALSE, bars are drawn vertically with first bar to the left (barplot parameter).
toplabelvar String. Name of variable in x for adding labels to place above each bar (e.g., NBRPLOTS.gt0).
ylim Number. A vector of min and max values, c(min,max) for the y axis (or x axis if horiz=TRUE). If NULL, defaults to maximum y value. If errbars=TRUE, the ylim defaults to the maximum y value plus the standard error.
divideby String. Conversion number for output (‘hundred’, ‘thousand’, ‘million’).
ylabel String. Label for the y axis (same as ylab).
xlabel String. Label for the x axis (same as xlab).
mar See par.. A numerical vector representing number of lines for margins (c(bottom, left, top, right).
addlegend Logical. If TRUE, adds legend to bar plot (only applicable if grouping).
main String. Title for plot.
cex.main Number. Expansion factor for title.
cex.label Number. A number representing cex in barplot (size expansion of x and/or ylabels).
cex.names Number. Expansion factor for axis names (bar labels) (e.g., 0.5 represents half the size.
las.xnames Number. The direction of x variable names (0,1,3). 0:diagonal (Default), 1:horizontal; 3:vertical.
las.ynames Number. The direction of y variable names (0,1,3). 0:diagonal (Default), 1:horizontal; 3:vertical.
savedata Logical. If TRUE, writes output data to outfolder (jpg and pdf).
outfolder String. The name of the output folder. If savedata=TRUE, all output saved to the outfolder. If savedata=FALSE, only a text file of input parameters is saved.
outfn String. The name of the output file if savedata=TRUE (*.csv). Do not include extension. If NULL, the file will be named BARPLOT_’yvar_date’.csv
outfn.pre String. Add a prefix to output name (e.g., "01").
outfn.date Logical. If TRUE, add date to end of outfile (e.g., outfn_’date’.csv).
overwrite Logical. If TRUE and exportshp=TRUE, overwrite files in outfolder.

additional arguments to pass to barplot(), including a list of arguments for legend() arguments (e.g., args.legend=list(x="topleft", bty="n"), for moving legend to topleft and removing box around legend).
Details
If parameters = NULL, then it will prompt user for input.

Value
Outputs barplot to display window.

Note
If savedata = TRUE, writes a jpg and pdf of barplot to outfolder.
To add legend parameters, add a parameter named args.legend, defined as a list of specific legend parameters (see ?legend)... e.g., args.legend=list(x="topright"). If specifying x and y, x defines the lower right corner of legend box and y defines the upper right corner of box.

Author(s)
Tracey S. Frescino

Examples
# Set up data frame for example
ftyptab <- data.frame(cbind(FORTYPCD = c(182, 184, 201, 221, 265),
                        FREQ = c(110, 7, 900, 410, 155),
                        SE = c(10, 11, 18, 14, 22)))

# Create basic barplot
datBarplot(x = ftyptab, xvar = "FORTYPCD")

# Add standard errors to basic barplot
datBarplot(x = ftyptab, xvar = "FORTYPCD", errbars = TRUE, sevar = "SE")

---

datBarStacked  

---

Description
Generate a barplot of frequencies in order from most to least.

Usage
datBarStacked(
  x,  
  main.attribute,  
  sub.attribute,  
  response = "phat",  
  percent = FALSE,  
  LUT.color = NULL,  
  color = "rainbow",  
)
Arguments

x  Data frame or comma-delimited file (*.csv) - table of values to be plotted.

main.attribute  String. The column to be used for each bar.

sub.attribute  String. The column to be used to subdivide each bar.

response  String. The column of values to be plotted. Currently defaults to "phat".

percent  Logical. If TRUE, values cover values in a stack are converted to percent of stack.

LUT.color  Data frame or comma-delimited file (*.csv) - look up table for colors. Must contain column with same name as sub.attribute.

color  String. Automated color selection ("rainbow", "topo", "heat", "terrain", "cm").
device.type  String. The type(s) of device for plotting ("default", "jpg", "pdf", or "ps").
jpeg.res  Integer. The resolution for jpeg image.
device.width  Integer. The width of output device (in inches)
device.height  Integer. The height of output device (in inches)
mar  See par. A numerical vector representing number of lines for margins (c(bottom, left, top, right).
horz  Logical. See barplot. If FALSE, the bars are drawn vertically with the first bar to the left. If TRUE, the bars are drawn horizontally with the first bar at the bottom.
bar.lim  Number vector. Equivalent to xlim or ylim, for whichever is the lengthwise axis in barcharts (ex. c(0,10)). Warning: for lower limits other than zero (ex. c(20,100), will behave strangely because par(xpd) is set to NA.
bar.ratio  Proportion of figure area taken up by barplot vs taken by legend.
ylabel  String. A label for the y axis (same as ylab).
xlabel  String. A label for the x axis (same as xlabel).
las.xnames  Number. The direction of x variable names (0,1,2,3). 0:Default, parallel; 1:horizontal; 2:perpendicular; 3:vertical.
main.order  String vector. A vector of main.attribute names in desired order for bars. May also be 'DESC' or 'ASC'.
sub.order  String vector. A vector of sub.attribute names in desired order for stack, with the first name used as the column in each stack. If NULL, the order is based on the overall cover of each sub.attribute. May also be 'DESC' or 'ASC'.
legend.fit  Logical. Should bar.lim be changed to fit the legend of the plot. Will only be used if the legend is on the right side of a horizontal plot or the top of a vertical plot. (i.e. horiz=FALSE).
legend.cex  Number. Expansion factor for legend text.
legend.x  See legend. The x coordinate to be used to position the legend. If horiz=TRUE, suggested options include "topright" or "bottomright". If horiz=FALSE, suggested options include "topleft" or "topright".
legend.y  See legend. The y coordinate to be used to position the legend.
legend.title  See legend. A title for the legend.
legend.bty  See legend. The type of box to be drawn around the legend.
legend.bg  See legend. The background color for the legend box.
legend.inset  See legend. The distance from the margins as a fraction of the plot region.
legend.xpd  See legend.
main  String. Title for plot.
cex.main  Number. Expansion factor for title.
cex.label  Number. A number representing cex in barplot (size expansion of x and/or y labels).
cex.names  Number. Expansion factor for axis names (bar labels). Ex. 0.5 represents half the size.
sub.add0 Logical. If TRUE, adds categories with 0 values to sub.attribute legend.
savedata Logical. If TRUE, writes output data to outfolder (jpg and pdf).
outfolder String. The name of the output folder. If savedata=TRUE, all output saved to the outfolder. If savedata=FALSE, only a text file of input parameters is saved.
outfn String. The name of the output file if savedata=TRUE (*.csv). Do not include extension. If NULL, the file will be named BARPLOT_'yvar_date'.csv
outfn.pre String. Add a prefix to output name (e.g., "01").
outfn.date Logical. If TRUE, add date to end of outfile (e.g., outfn_'date'.csv).
overwrite Logical. If TRUE and exportshp=TRUE, overwrite files in outfolder.
... list of additional arguments to pass to barplot(); names of the list are used as argument names.

Details

# Note: This function uses a customized lengthwise axis (y axis if horiz=F, x axis if horiz=T) # Therefore to modify this axis by par commands (other than the # ones specifically included above) may require changing the function. For example par(yaxp) # may not work. # # # The arguments bar.lab and bar.lim are equivalent to xlab/ylab, xlim/ylim, controlling # whichever is the lengthwise axis in the bar charts (as determined by horiz). # # # bar.ratio is only used when bar.lim is specified, it controls what percentage of the # plot area will be occupied by the specified range of the chart itself. For example, setting # bar.lim=c(0,80) and bar.ratio=0.75 will result in the 80 region, leaving 25 the bars are taller than 80 bar.lim is not specified, the function will automatically scale the lengthwise axis so # that all the bars are as tall as possible, but none of the bars over lap the legend. # # # The main purpose of specifying bar.lim is if you want multiple forest types to be shown at # the same scale, sacrificing individual forest type details to make the graphs comparable # across forest types. Then you may need to fiddle with bar.ratio till the plots all look good.

Value

Outputs stacked barplot to display window.

Note

If savedata = TRUE, writes a jpg and pdf of barplot to outfolder.

Author(s)

Elizabeth Freeman, Tracey S. Frescino
**datFilter**

Data - Filters data table.

### Description

Subsets a data table by specified filter(s).

### Usage

```r
datFilter(
  x, 
  xfilter = NULL, 
  xfiltervar = NULL, 
  othertabnms = NULL, 
  uniqueid = "PLT_CN", 
  vardelete = NULL, 
  title.filter = NULL, 
  savedata = FALSE, 
  filternm = NULL, 
  stopifnull = FALSE, 
  returnDT = TRUE, 
  xnm = NULL, 
  savedata_opts = NULL, 
  gui = FALSE 
)
```

### Arguments

- **x**
  Data frame, sf data frame or comma-delimited file (*.csv). A data frame to filter.

- **xfilter**
  String. A filter expression. Must be R syntax. (e.g., "STATUSCD == 1", "INVYR within double quotes (e.g., "SPP == 'Lodgepole'")"). If NULL, a window pops up to select filter variable(s) and filter value(s).

- **xfiltervar**
  String. The filtervar if you know what it is. If NULL, a window will pop up to select filter value(s).

- **othertabnms**
  String vector. Name(s) of the objects or comma-delimited files to subset. Names must be in quotes (e.g., othertables=c('tree', 'cond')).

- **uniqueid**
  String. Unique identifier of x. Only needed if othertables != NULL. The uniqueid must be the same for all tables except if PLT_CN and CN.

- **vardelete**
  String vector. Vector of variables you would like deleted from filter list. Mostly used for internal queries.

- **title.filter**
  String. Title of the filter query window. Mostly used for internal queries.

- **savedata**
  Logical. If TRUE, writes output data to outfolder.

- **filternm**
  String. Optional. Name of filter, for feedback purposes.

- **stopifnull**
  Logical. If TRUE, stop if output is NULL.
returnDT Logical. If TRUE, returns a data table. If FALSE, returns a data frame.

xnm String. Name for filter attribute. Used for warning messages.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = 'datf'.

gui Logical. If TRUE, pop-up windows will appear for user-interface.

Details

If no parameters, then user is prompted for input. If partial parameters, default parameter values are used.

Value

A list of the following items:

xf A data frame of filtered x.

xfilter The xfilter.

If othertables != NULL, the other tables, named with 'in' prefix

Note

If message returned is 'filter removed all records', either the filter removed all records in x or the filter is incorrect.

Author(s)

Tracey S. Frescino

Examples

# Set up data for example
tab <- data.frame(cbind(CONDCLASS=c(1, 1, 2, 1, 3, 3, 3, 1, 1, 1, 2, 1),
    FORTYPCD = c(182, 184, 201, 221, 221, 184, 221, 182, 182, 201, 182, 221)))

# Filter for value not equal to 182
datFilter(x = tab, xfilter = "FORTYPCD != 182")

# Filter on two conditions, grab xf object from list
datFilter(x = WYcond, xfilter = "FORTYPCD == c(221) & STDSZCD == 3")$xf
**datFreq**

*Data - Get frequency table for specified variable(s).*

---

**Description**

Generates a frequency table from a data frame, including number of records by a specified variable or variables in the data frame with optional totals and/or subtotals.

**Usage**

```r
datFreq(
x,  
xvar = NULL,  
total = FALSE,  
subtotal = FALSE,  
subtotalcol = NULL,  
savedata = FALSE,  
savedata_opts = NULL,  
gui = FALSE
)
```

**Arguments**

- `x`  
  Data frame or comma-delimited file (*.csv). The table with the variable(s).

- `xvar`  
  String (vector).* The name of the variable(s) to summarize.

- `total`  
  Logical. If TRUE, a row is added to bottom of table with a total for the whole table.

- `subtotal`  
  Logical. If TRUE, a row is added to bottom of each section for subtotals.

- `subtotalcol`  
  Logical. If subtotal=TRUE, the column(s) to generate subtotals.

- `savedata`  
  Logical. If TRUE, writes output data to outfolder.

- `savedata_opts`  
  List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = ‘datfreq’.

- `gui`  
  Logical. If TRUE, pop-up windows will appear for user-interface.

**Details**

If no parameters, then user is prompted for input. If partial parameters, default parameter values are used.

**Value**

- `freqtable`  
  Data frame. The frequency table.

If savedata=TRUE, a comma-delimited file of the frequency table is written to the outfolder.
Author(s)

Tracey S. Frescino

Examples

# Set up data for example
tab <- data.frame(cbind(CONDCLASS = c(1, 1, 2, 1, 3, 3, 3, 1, 1, 2, 1, 2),
                      FORTYPCD = c(182, 184, 201, 221, 221, 184, 221, 182, 182, 182, 221, 221)))

# Frequency table with "FORTYPCD"
datFreq(x = tab, xvar = "FORTYPCD")

# Frequency table with "CONDCLASS" and "FORTYPCD"
datFreq(x = tab, xvar = c("CONDCLASS", "FORTYPCD"))

# Frequency table with "CONDCLASS" and "FORTYPCD", adding total and subtotal
# rows
datFreq(x = tab, xvar = c("CONDCLASS", "FORTYPCD"),
        total = TRUE,
        subtotal = TRUE)

# Frequency table for WYtree, multiple variables, subtotal options
datFreq(x = FIESTA::WYtree, xvar = c("SPGRPCD", "SPCD", "STATUSCD"),
        subtotal = TRUE, subtotalcol = "SPCD")

datLineplot

Data - Generates line graph.

Description

Generate a line plot of multiple estimates.

Usage

datLineplot(
  x, xvar, yvar,
  plotCI = FALSE, sevar = NULL,
  CIlst = c(68, 95),
  CIcolorlst = c("dark grey", "black"),
  addshade = FALSE,
  device.type = "dev.new",
  ...)
Arguments

x                  Data frame or comma-delimited file (*.csv) - a frequency table.
xvar               String. Name of X variable.
yvar               String. Name of the y variable (e.g., FREQ).
plotCI             Logical. If TRUE, adds confidence intervals to plot as dotted lines.
sevar              String. Name of the variable with standard error values.
CIlst              String. Numeric vector. If plotCI=TRUE, identifies percent confidence interval to add to plot.
CIcolorlst         String. Character vector. If plotCI=TRUE, identifies colors to plot confidence interval lines. Must be same length as CIlst and from colors() list.
addshade           Logical. If TRUE, adds a light grey shading between the large confidence interval lines.
device.type        String. Type(s) of device for plotting ("dev.new", "jpg", "pdf").
jpeg.res           Integer. Resolution for jpeg image.
device.height      Integer. Height (in inches) of barplot, if writing to file.
device.width       Integer. Width (in inches) of barplot, if writing to file.
ylim               Number. A vector of min and max values, c(min,max) for the y axis (or x axis if horiz=TRUE). If NULL, defaults to maximum y value. If errbars=TRUE, the ylim defaults to the maximum y value plus the standard error.
divideby  String. Conversion number for output ('hundred', 'thousand', 'million').
ylabel    String. Label for the y axis (same as ylab).
xlabel    String. Label for the x axis (same as xlab).
xticks    Numeric vector. Vector of tick marks for x axis.
mar        See par.. A numerical vector representing number of lines for margins (c(bottom, left, top, right).
addlegend Logical. If TRUE, adds legend to bar plot (only applicable if grouping).
main       String. Title for plot.
cex.main   Number. Expansion factor for title.
cex.label  Number. A number representing cex in barplot (size expansion of x and/or yla-
            bels).
cex.names  Number. Expansion factor for axis names (bar labels) (e.g., 0.5 represents half
            the size.
las.xnames Number. The direction of x variable names (0,1,3). 0:diagonal (Default), 1:hor-
            izontal; 3:vertical.
las.ynames Number. The direction of y variable names (0,1,3). 0:diagonal (Default), 1:hor-
            izontal; 3:vertical.
savedata  Logical. If TRUE, writes output data to outfolder (jpg and pdf).
outfolder  String. The name of the output folder. If savedata=TRUE, all output saved to
            the outfolder. If savedata=FALSE, only a text file of input parameters is saved.
outfn      String. The name of the output file if savedata=TRUE (*.csv). Do not include
            extension. If NULL, the file will be named BARPLOT_'yvar_date'.csv
outfn.pre  String. Add a prefix to output name (e.g., "01").
outfn.date Logical. If TRUE, add date to end of outfile (e.g., outfn_'date'.csv).
overwrite  Logical. If TRUE and exportshp=TRUE, overwrite files in outfolder.
            ...
            additional arguments to pass to barplot(), including a list of arguments for leg-
            end() arguments (e.g., args.legend=list(x="topleft", "bty="n"), for moving leg-
            end to topleft and removing box around legend).

Details

If parameters = NULL, then it will prompt user for input.

Value

Outputs barplot to display window.

Note

If savedata = TRUE, writes a jpg and pdf of barplot to outfolder.

To add legend parameters, add a parameter named args.legend, defined as a list of specific legend
parameters (see ?legend)... ex. ... args.legend=list(x="topleft", "bty="n")). If specifying x and y, x defines
the lower right corner of legend box and y defines the upper right corner of box.
Author(s)

Tracey S. Frescino

Examples

# Lineplot of cubic foot volume by above-ground biomass, Wyoming tree data
# datLineplot(x = WYtree, xvar = "VOLCFNET", yvar = "DRYBIO_AG") # needs work

datLUTclass

Data - Create a variable with classified values.

Description

Merge a look-up table to define categories of continuous data in x (e.g., DIA). Adds a variable (LUTclassnm) to x, defining as: xvar >= MIN (and xvar <= MAX).

Usage

datLUTclass(
  x,
  xvar = NULL,
  LUT = NULL,
  minvar = NULL,
  maxvar = NULL,
  cutbreaks = NULL,
  cutlabels = NULL,
  LUTclassnm = NULL,
  label.dec = 1,
  NAto0 = FALSE,
  vars2keep = NULL,
  keepcutbreaks = FALSE,
  savedata = FALSE,
  savedata_opts = NULL,
  gui = FALSE
)

Arguments

x Data frame or comma-delimited file (*.csv). The data table with variable to classify.

xvar String. Name of variable in the data table to join to.

LUT Data frame or comma-delimited file (*.csv). Name of the look-up table with collapsed classes. Lookup table should include minimum values for classes, maximum values for classes, and a name of class (i.e., LUTclassnm). Maximum values and names are optional.
minvar String. If LUT is not null, name of variable with minimum class value (>= minvar).

maxvar String. Optional. If LUT is not null, name of variable with maximum class value (<= maxvar).

cutbreaks Numeric vector. Vector of numbers for minimum class values.

cutlabels String vector. Optional. Vector of names for classes. If NULL, class names are generated from cutbreaks.

LUTclassnm String. Optional. Name of classified variable in x. If LUT is not null and class names are included, this is the name of variable with class names. If NULL, a class names are generated from minvar or minvar and maxvar with default name equal to ‘xvar’CL.

label.dec Integer. Number of decimals to include in label.

NAto0 Logical. If TRUE, converts NA values to 0 before classification.

vars2keep String vector. Variable names from LUT to keep (append) to x.

keepcutbreaks Logical. If TRUE, the cutbreaks used for creating classes are appended to dataset.

savedata Logical. If TRUE, saves data to outfolder.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = ‘datlutcl’.

gui Logical. If gui, user is prompted for parameters.

Details
Use datLUTclass() to prompt for input.

Value

xLUT Input data table with look-up table variable(s).

LUTclassnm Name of the classified variable.

LUT Look-up table with categories.

If savedata = TRUE, a comma-delimited file is output to the outfolder as outfn. If outfn = NULL, the name of the file will be datlut_‘date’.csv.

Note
The look-up table format must be similar to the following table Set LUTclassnm = VARCLNM. MAX and VALCLNM columns are optional.

<table>
<thead>
<tr>
<th>MIN</th>
<th>MAX</th>
<th>VARCLNM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.9</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>9.9</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>15.0</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>19.9</td>
<td>4</td>
</tr>
<tr>
<td>20</td>
<td>24.9</td>
<td>5</td>
</tr>
<tr>
<td>25</td>
<td>49.9</td>
<td>6</td>
</tr>
</tbody>
</table>
**datLUTnm**

**Author(s)**

Tracey S. Frescino

**Examples**

```r
head(FIESTA::ref_diacl2in)
WYtreelut <- datLUTclass(FIESTA::WYtree,
  xvar = "DIA",
  LUT = FIESTA::ref_diacl2in,
  LUTclassnm = "DIACL2IN")
names(WYtreelut)
head(WYtreelut$xLUT)
table(WYtreelut$xLUT$DIACL2IN)

WYtreelut2 <- datLUTclass(FIESTA::WYtree,
  xvar = "DIA",
  cutbreaks = c(1, 5, 25, 50, 100),
  LUTclassnm = "DIACL2IN")
names(WYtreelut2)
head(WYtreelut2$xLUT)
table(WYtreelut2$xLUT$DIACL2IN)

# Create look-up table of stand age classes
MIN <- c(0, 20, 40, 60, 80, 101)
STDAGENM <- c("0-20", "21-40", "41-60", "61-80", "81-100", "101+")
stdagelut <- data.frame(MIN = MIN, STDAGENM = STDAGENM)
stdagelut

WYcondlut <- datLUTclass(FIESTA::WYcond,
  xvar = "STDAGE",
  LUT = stdagelut,
  LUTclassnm = "STDAGENM")
names(WYcondlut)
head(WYcondlut$xLUT)
table(WYcondlut$xLUT$STDAGENM)
```

---

**datLUTnm**  
Data - Gets variable description or class.

**Description**

Merge a look-up table to append new variables, names, or categories to x.

**Usage**

```r
datLUTnm(
  x,
  xvar = NULL,
  LUT = NULL,
  ...)
```

LUTvar = NULL,
LUTnewvar = NULL,
LUTnewvarnm = NULL,
FIAname = FALSE,
NAclass = "Other",
group = FALSE,
add0 = FALSE,
stopifmiss = FALSE,
xtxt = NULL,
savedata = FALSE,
savedata_opts = NULL,
 gui = FALSE
)

Arguments

x  Data frame or comma-delimited file (*.csv). The data table with variable to classify.
xvar  String. Name of variable in the data table to join to.
LUT  Data frame or comma-delimited file (*.csv). Name of the file with collapsed classes (If FIAname=FALSE).
LUTvar  String. Name of variable in LUT with values matching that xvar. If LUTvar=NULL, LUTvar=xvar.
LUTnewvar  String. Name(s) of other variable(s) in the look-up table to include in join. If NULL, all other variables in table will be included.
LUTnewvarnm  String. Different name(s) for LUTnewvar. If NULL, names will default to LUTnewvar. The length of LUTnewvarnm must equal the length for LUTnewvar.
FIAname  Logical. If TRUE, get FIA reference name based on (ref_codes) within FIESTA.
NAclass  String. NA values in xvar will be changed to NAclass.
group  Logical. If TRUE, the group variables in reference table (ref_codes) are merged to data table (GROUPCD, GROUPNM).
add0  Logical. If TRUE, keep all codes in look up table. If FALSE, only include codes that are in x.
stopifmiss  Logical. If TRUE, stops function if missing codes in LUTx.
xtxt  String. Name of x table for more useful information in warnings.
savedata  Logical. If TRUE, saves data to outfolder.
savedata_opts  List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = 'datlut'.
 gui  Logical. If gui, user is prompted for parameters.

Value

xLUT  The input data table with look-up table variable(s).
xLUTnm  Name of the new variable(s).
datPBplotchg

LUT Look up table with categories.
If savedata = TRUE, a comma-delimited file is output to the outfolder as outfn. If outfn = NULL, the name of the file will be datlut_’date’.csv.

Note
For available reference tables: sort(unique(ref_codes$VARIABLE))

Author(s)
Tracey S. Frescino

Examples

# Append forest type names using the reference table above.
ref_fortypcd <- ref_codes[ref_codes$VARIABLE == "FORTYPCD",]
WYcondlut <- datLUTnm(WYcond,
  xvar = "FORTYPCD",
  LUT = ref_fortypcd,
  LUTvar = "VALUE",
  LUTnewvar = "MEANING",
  LUTnewvarnm = "FORTYPNM")
names(WYcondlut)
WYcond2 <- WYcondlut$xLUT
head(WYcond2[WYcond2$FORTYPCD > 0, ])

# Append forest type names the FIAname parameter. If the xvar is in the stored
# reference table, the name and values will automatically be appended.
WYcondlut2 <- datLUTnm(WYcond,
  xvar = "FORTYPCD",
  FIAname = TRUE)
names(WYcondlut2)
WYcond3 <- WYcondlut2$xLUT
head(WYcond3[WYcond3$FORTYPCD > 0, ])

---

datPBplotchg Data - Generates barplot of photo-based change estimates.

Description
Generate a bar plot of net change for photo-based estimates of land use / land cover change.

Usage
datPBplotchg(gainloss, CI = 95, figTitle = "")
Arguments

**gainloss**  
Data frame or comma-delimited file (*.csv) - table with gain loss estimates.

**CI**  
Number. Confidence Interval to include on plot.

**figTitle**  
String. Title of figure.

Value

Outputs barplot to display window.

Note

If savedata = TRUE, writes a jpg and pdf of barplot to outfolder.

Author(s)

Tracey S. Frescino

---

**datPBpnt2pct**  
*Data - Transpose point data to plot-level percent by domain.*

Description

Calculates percent point by plot and domain and transpose to plot level.

Usage

datPBpnt2pct(pnt, uniqueid, tvar, othervars = NULL)

Arguments

**pnt**  
DF/DT or comma-delimited file (*.csv). Point-level table with one record per point. If NULL, aggregated point counts must be in pntcnt.

**uniqueid**  
String. Name of unique identifier of plot in pnt.

**tvar**  
String. Name of variable to transpose.

**othervars**  
String vector. Name(s) of plot-level variables to merge with transposed data.

Value

**pltdom.pct**  
Data frame with transposed data.

Author(s)

Tracey S. Frescino
Examples

# Load necessary data from FIESTA
## Point data
icepntfn <- system.file("extdata", 
    "PB_data/icepnt_utco1135.csv", 
    package = "FIESTA")
icepnt <- read.csv(icepntfn)

# Convert to percent
percent <- datPBpnt2pct(pnt = icepnt, 
    uniqueid = "plot_id", 
    tvar = "cover_1")

head(percent)

---

datPivot

Data - Generates a pivot table.

Description

Generates a pivot table of values by x row and y column.

Usage

datPivot(
    x, 
    pvar, 
    xvar, 
    yvar, 
    pfun = sum, 
    xfilter = NULL, 
    NAto0 = TRUE, 
    dropNAxvar = TRUE, 
    dropNAyvar = TRUE, 
    pvar.round = 2, 
    savedata = FALSE, 
    savedata_opts = NULL, 
    gui = FALSE) 

Arguments

- **x**: Dataframe. Table with pivot variables.
- **pvar**: String. The name of the variable for pivot table values.
- **xvar**: String. The name of the variable for rows.
- **yvar**: String. The name of the variable for columns.
- **pfun**: Function. The name of the function to use for pivot table values (ex. sum, mean, max).
datPlotcnt

**Description**

Extract plot counts by inventory year and state.

**Value**

- `ptab`: Matrix. The pivot table.

**Author(s)**

Tracey S. Frescino

**Examples**

```r
# Pivot WYcond table
datPivot(x = FIESTA::WYcond,
    pvar = "CONDPROP_UNADJ",
    xvar = "FORTYPCD",
    yvar = "STDSZCD")

# Pivot WYtree table
datPivot(x = FIESTA::WYtree,
    pvar = "TPA_UNADJ",
    xvar = "SPCD",
    yvar = "STATUSCD",
    pfun = mean,
    NAto0 = TRUE)
```

---

**datPlotcnt**

*Database - Get plot counts.*

**Description**

Extract plot counts by inventory year and state.
datPlotcnt

Usage

datPlotcnt(
    plt,
    yrtype = "INVYR",
    states = NULL,
    designcd = FALSE,
    forsamp = TRUE,
    total = TRUE,
    subtotal = TRUE,
    savedata = FALSE,
    outfolder = NULL,
    outfn = NULL,
    gui = FALSE
)

Arguments

plt Data frame. Table of plot-level variables to count plots. If using this table, it must include INVYR.

yrtype String. Type of year to categorize data ("INVYR", "MEASYEAR").

states String vector. The states in plt table.

designcd Logical. If TRUE, includes FIA design codes in the table.

forsamp Logical. If TRUE, includes forest/nonforest/nonsampled codes in the table.

total Logical. If TRUE, a row is added to bottom of table with a total for the whole table.

subtotal Logical. If TRUE, a row is added to bottom of each section for subtotals.

savedata Logical. If TRUE, saves data to outfolder as comma-delimited file (*.csv). No objects are returned. If FALSE, the data are saved as R objects and returned to user. See details for caveats.

outfolder String. The output folder path. If NULL and savedata=TRUE or parameters=TRUE, outfolder is the working directory.

outfn String. The name of the output file. If NULL, defaults to pltcnt_`date`.csv

gui Logical. If TRUE, gui windows pop up for parameter selection.

Value

pltcnt - a dataframe of counts (YEAR, STABBR, STCD, PLOTS, NONSAMPLED, FOREST, NONFOREST)

Author(s)

Tracey S. Frescino
datSumCond

**Data - Aggregates numeric condition data to plot level.**

### Description

Aggregates CONDPROP_UNADJ variable or other continuous condition variables to plot level with option to apply condition filters. If condition variable is not CONDPROP_UNADJ the variable is multiplied by CONDPROP_UNADJ for weighted sum.

### Usage

```r
datSumCond(
  cond = NULL,
  datsource = "obj",
  data_dsn = NULL,
  plt = NULL,
  plt_dsn = NULL,
  cuniqueid = "PLT_CN",
  puniqueid = "CN",
  csumvar = NULL,
  csumvarnm = NULL,
  cfilter = NULL,
  getadjplot = FALSE,
  adjcond = FALSE,
  NAto0 = FALSE,
  returnDT = TRUE,
  savedata = FALSE,
  savedata_opts = NULL,
  gui = FALSE
)
```

### Arguments

- **cond**: Data frame or comma-delimited file (*.csv). Condition-level table with aggregate variable and CONDPROP_UNADJ.
- **data_dsn**: String. If datsource=’sqlite’, the name of SQLite database (*.sqlite).
- **plt**: Data frame, comma-delimited file (*.csv), shapefile (*.shp), or database file. Plot-level table to join the aggregated tree data to (if bycond=FALSE). Nonsampled plots (PLOT_STATUS_CD = 3) are removed. Optional.
- **plt_dsn**: String. The data source name (dsn; i.e., folder or database name) of plt. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional.
- **cuniqueid**: String. Unique identifier of cond (default = "PLT_CN").
- **puniqueid**: String. Unique identifier of plt (default = "CN").
**datSumCond**

- **csumvar** String. One or more variable names to sum to plot level.
- **csumvarnm** String. Name of the resulting aggregated plot-level variable(s). Default = csumvar + '_PLT'.
- **cfilter** String. A filter to subset the cond data before aggregating (e.g., "COND_STATUS_CD == 1"). Must be R syntax.
- **getadjplot** Logical. If TRUE, adjustments are calculated for nonsampled conditions on plot.
- **adjcond** Logical. If TRUE, csumvar condition variables are adjusted for nonsampled conditions by plot.
- **NAto0** Logical. If TRUE, convert NA values to 0.
- **returnDT** Logical. If TRUE, returns data.table object(s). If FALSE, returns data.frame object(s).
- **savedata** Logical. If TRUE, saves data to outfolder.
- **savedata_opts** List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = 'condsum'.
- **gui** Logical. If gui, user is prompted for parameters.

**Details**

If variable = NULL, then it will prompt user for input.

**Value**

A list of the following items:

- **condsum** Data frame. Plot-level table with aggregated condition attribute.
- **cfilter** Condition filter.

If savedata=TRUE, condsum is saved to the outfolder.

**Note**

Nonsampled plots are removed from table.

**Author(s)**

Tracey S. Frescino

**Examples**

```r
# Aggregate LIVE_CANOPY_CVR_PCT to plot, weighted by CONDPROP_UNADJ
condsum <- datSumCond(cond = FIESTA::WYcond,
                       csumvar = "LIVE_CANOPY_CVR_PCT")$condsum

# Check results
condsum[condsum$PLT_CN == 40404737010690,]
FIESTA::WYcond[FIESTA::WYcond$PLT_CN == 40404737010690,]
```
datSumTree  

**Data** - Aggregates numeric tree data to the plot or condition-level.

**Description**

Aggregates numeric tree-level data (e.g., VOLCFNET) to plot or condition, including options for filtering tree data or extrapolating to plot acre by multiplying by TPA.

**Usage**

```r
datSumTree(
  tree = NULL,
  seed = NULL,
  cond = NULL,
  datsource = "obj",
  data_dsn = NULL,
  plt = NULL,
  plt_dsn = NULL,
  tuniqueid = "PLT_CN",
  cuniqueid = "PLT_CN",
  puniqueid = "CN",
  bycond = FALSE,
  condid = "CONDID",
  bysubp = FALSE,
  subpid = "SUBP",
  tsumvarlst = NULL,
  tsumvarnmlst = NULL,
  TPA = TRUE,
  tfun = sum,
  ACI = FALSE,
  tfilter = NULL,
  addseed = FALSE,
  lbs2tons = TRUE,
  metric = FALSE,
  getadjplot = FALSE,
  adjtree = FALSE,
  adjvar = "tadjfac",
  adjTPA = 1,
  NAto0 = FALSE,
  tround = 5,
  checkNA = FALSE,
  returnDT = TRUE,
  savedata = FALSE,
  savedata_opts = NULL,
  gui = FALSE
)
```
Arguments

tree  Dataframe or comma-delimited file (*.csv). The tree-level table.

seed Dataframe or comma-delimited file (*.csv). The seedling table.

cond Dataframe or comma-delimited file (*.csv). Condition-level table to join the aggregated tree data to, if bycond=TRUE. This table also may be used for condition proportion or strata variables used if adjcond or adjstrata = TRUE (See details below). This table is optional.

datasource String. Source of data ('obj', 'csv', 'sqlite', 'gdb').

data_dsn String. If datasource='sqlite', the name of SQLite database (*.sqlite).

plt Dataframe, comma-delimited file (*.csv), or shapefile (*.shp). Plot-level table to join the aggregated tree data to, if bycond=FALSE. This table is optional.

plt_dsn String. The data source name (dsn; i.e., folder or database name) of plt. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional.

tuniqueid String. Unique identifier of plt in tree table.

cuniqueid String. Unique identifier of plt in cond table if cond is NOT NULL.

puniqueid String. Unique identifier of plt table if plt is NOT NULL.

bycond Logical. If TRUE, the data are aggregated to the condition level (by: cuniqueid, condid). If FALSE, the data are aggregated to the plot level (by: puniqueid).

condid String. Unique identifier for conditions.

bysubp Logical. If TRUE, data are aggregated to the subplot level.

subpid String. Unique identifier of each subplot.

tsumvarlst String (vector). Tree-level variable(s) to aggregate (e.g., "TPA_UNADJ", "BA"). Use tsumvar="PLT_CN" (tfun=sum) for summed tree count.

tsumvarnmlst String (vector). Name of the tree-level variable(s) to aggregate (e.g., "TPALIVE", "BALIVE"). This list must have the same number of variables as tsumvarlst and be in respective order. If NULL, the default names will be tsumvar'_tfun' (e.g., "TPA_UNADJ_SUM", "BA_SUM").

TPA Logical. If TRUE, tsumvarlst variable(s) are multiplied by the respective trees-per-acre variable (see details) to get per-acre measurements.

tfun Function. The name of the function to use to aggregate the data (e.g., sum, mean, max).

ACI Logical. If TRUE, if ACI (All Condition Inventory) plots exist, any trees on these plots will be included in summary. If FALSE, you must include condition table.

tfilter String. Filter to subset the tree data before aggregating (e.g., "STATUSCD == 1"). This must be in R syntax. If tfilter=NULL, user is prompted. Use tfilter="NONE" if no filters.

addseed Logical. If TRUE, add seedling data to tree counts (if TPA variable in tsumvarlst).

lbs2tons Logical. If TRUE, converts biomass or carbon variables from pounds to tons.
metric Logical. If TRUE, converts response to metric units based on FIESTA::ref_conversion, if any variable in tsumvarlst is in FIESTAUtils::ref_estvar. Note: if TPA, TPA is converted to trees per hectare (TPH: 1 / (tpavar * 0.4046860)).

getadjplot Logical. If TRUE, adjustments are calculated for nonsampled conditions on plot.

adjtree Logical. If TRUE, trees are individually adjusted by adjustment factors. Adjustment factors must be included in tree table (see adjvar).

adjvar String. If adjtree=TRUE, the name of the variable to use for multiplying by adjustment (e.g., tadjfac).

adjTPA Numeric. A tree-per-acre adjustment. Use for DESIGNCD=1 (annual inventory), if using less than 4 subplots. If using only 1 subplot for estimate, adjTPA=4. The default is 1.

NAt0 Logical. If TRUE, convert NA values to 0.

tround Number. The number of digits to round to. If NULL, default=6.

checkNA Logical. If TRUE, checks if NA values exist in necessary variables.

returnDT Logical. If TRUE, returns data.table object(s). If FALSE, returns data.frame object(s).

savedata Logical. If TRUE, saves data to outfolder.
savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = 'treesum'.

gui Logical. If gui, user is prompted for parameters.

Details

If variable = NULL, then it will prompt user for input.

Dependent external functions: datFilter Dependent internal functions: addcommas, fileexistsnm, getadjfactor

For adjcond (bycond=FALSE):
If you want to get trees-per-acre information aggregated to plot or condition level, you need to include a TPA variable in tree table.

For tsumvars = GROWCFGS, GROWBFSL, GROWCFAL, FGROWCFGS, FGROWBFSL, or FGROWCFAL, you must have TPAGROW_UNADJ

For tsumvars = MORTCFGS, MORTBFSL, MORTCFAL, FMORTCFGS, FMORTBFSL, or FMORTCFAL, you must have TPAMORT_UNADJ

For tsumvars = REMVCFGS, REMVBFSL, REMVCFAL, FREMVCFGS, FREMVBFSL, or FREMVCFA, you must have TPAREMV_UNADJ

If you want to adjust plot-level information by condition proportions (adjplot), you need to include CONDID & CONDPROP_UNADJ in cond or tree table and COND_STATUS_CD and FORTYPCD if adjplot="FVS". For adjplot="WEIGHTED" - each tree is weighted (or multiplied) by the condition proportions (CONDPROP_UNADJ). For adjplot="FVS", each tree is weighted (or multiplied) by the sum of the condition proportions (CONDPROP_UNADJ) that are forested and not nonstocked (COND_STATUS_CD = 1 and FORTYPCD != 999).
Value

A list of the following items:

- **treedat**: Data frame. Plot or condition-level table with aggregated tree attributes.
- **sumvars**: String vector. Name(s) of the output aggregated tree attributes.

If `savedata=TRUE`
- `treedat` will be saved to the outfolder.
- A text file of input parameters is saved to outfolder (`'outfn'_parameters_'date'.txt).

Note

If a dat table is provided, the aggregated tree data will be merged to table and NULL values will be output as 0.

Author(s)

Tracey S. Frescino

Examples

```r
# Aggregate LIVE_CANOPY_CVR_PCT to plot, weighted by CONDPROP_UNADJ
treesum <- datSumTree(tree = FIESTA::WYtree,
                      tsumvarlst = "TPA_UNADJ")$treedat

# Check results
treesum[treesum$PLT_CN == 40404737010690,]
FIESTA::WYtree[FIESTA::WYtree$PLT_CN == 40404737010690,]
```

Description

Aggregates numeric tree domain data (e.g., SPCD) to plot or condition, including options for filtering tree data or extrapolating to plot acre by multiplying by TPA. Includes options for generating barplots, proportion data, and cover data.

Usage

datSumTreeDom(
  tree = NULL,
  seed = NULL,
  cond = NULL,
  datsource = "obj",
  data_dsn = NULL,
  plt = NULL,
  ...)
plt_dsn = NULL,
tuniqueid = "PLT_CN",
cuniqueid = "PLT_CN",
puniqueid = "CN",
bycond = FALSE,
condid = "CONDID",
bysubp = FALSE,
subpid = "SUBP",
tsumvar = NULL,
TPA = TRUE,
tfun = sum,
ACI = FALSE,
tfilter = NULL,
lbs2tons = TRUE,
metric = FALSE,
tdomvar = "SPCD",
tdomvar1st = NULL,
tdomvar2 = NULL,
tdomvar21st = NULL,
tdomprefix = NULL,
tdombarplot = FALSE,
tdomtot = FALSE,
tdomtotnm = NULL,
FIAname = FALSE,
addseed = FALSE,
pivot = TRUE,
presence = FALSE,
proportion = FALSE,
cover = FALSE,
getadjplot = FALSE,
adjtree = FALSE,
adjvar = "tadjfac",
NAto0 = FALSE,
adjTPA = 1,
tround = 5,
checkNA = FALSE,
returnDT = TRUE,
savedata = FALSE,
savedata_opts = NULL,
gui = FALSE
)

Arguments

  tree       Data frame or comma-delimited file (*.csv). The tree-level table with tree domain data.
  seed       Data frame or comma-delimited file (*.csv). The seedling table with tree seedling counts. Only applicable for counts (tsumvar="PLT_CN").
cond
Data frame or comma-delimited file (*.csv). Condition-level table to join the aggregated tree data to, if bycond=TRUE. This table also may be used for condition proportion or strata variables used if adjcond or adjstrata = TRUE (See details below). This table is optional. If included, CONDID must be present in table.

datasource

data_dsn
String. If datasource=’sqlite’, the name of SQLite database (*.sqlite).

plt
Data frame, comma-delimited file (*.csv), shapefile (*.shp), or database file. Plot-level table to join the aggregated tree data to (if bycond=FALSE). Nonsampled plots (PLOT_STATUS_CD = 3) are removed. Optional.

plt_dsn
String. The data source name (dsn; i.e., folder or database name) of plt. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional.

tuniqueid
String. Unique identifier of the tree table. If including seedling table, this should be the same for seed.

cuniqueid
String. Unique identifier of the cond table if cond is NOT NULL.

puniqueid
String. Unique identifier of the plt table if plt is NOT NULL.

bycond
Logical. If TRUE, data are aggregated to the condition level (by: uniqueid, CONDID). If FALSE, data are aggregated to the plot level (by: uniqueid).

condid
String. Unique identifier for conditions.

bysubp
Logical. If TRUE, data are aggregated to the subplot level.

subpid
String. Unique identifier of each subplot.

tsumvar
String. Name of the variable to aggregate (e.g., "BA"). For summing number of trees, use tsumvar="PLT_CN" with tfun=sum.

TPA
Logical. If TRUE, tsumvarlst variable(s) are multiplied by the respective trees-per-acre variable (see details) to get per-acre measurements.

tfun
Function. Name of the function to use to aggregate the data (e.g., sum, mean, max).

ACI
Logical. If TRUE, if ACI (All Condition Inventory) plots exist, any trees on these plots will be included in summary. If FALSE, you must include condition table.

tfilter
String. A filter to subset the tree data before aggregating (e.g., "STATUSCD == 1"). This must be in R syntax. If tfilter=NULL, user is prompted. Use tfilter="NONE" if no filters.

lbs2tons
Logical. If TRUE, converts biomass or carbon variables from pounds to tons.

metric
Logical. If TRUE, converts response to metric units based on FIESTA::ref_conversion, if tsumvar is in FIESTAutils::ref_estvar. Note: if TPA, TPA is converted to trees per hectare (TPH: 1 / (1/ tpavar * 0.4046860)).

tdomvar
String. The tree domain (tdom) variable used to aggregate by (e.g., "SPCD", "SPGRPCD").

tdomvarlst
String (vector). List of specific tree domains of tdomvar to aggregate (ex. c(108, 202)). If NULL, all domains of tdomvar are used.
datSumTreeDom

- **tdomvar2**: String. A second tree domain variable to use to aggregate by (ex. "DIACL"). The variables, tdomvar and tdomvar2 will be concatenated before summed.
- **tdomvar2lst**: String (vector). List of specific tree domains of tdomvar2 to aggregate. If NULL, all domains of tdomvar2 are used.
- **tdomprefix**: String. The prefix used for naming the aggregated tree data, before numeric codes (e.g., "SP" = SP102, SP746).
- **tdombarplot**: Logical. If TRUE and pivot=TRUE, calls datBarplot() and outputs a barplot of tdom distributions. If savedata=TRUE, barplots are written to outfolder.
- **tdomtottot**: Logical. If TRUE and pivot=TRUE a total of all tree domains in tdomvarlst is calculated and added to output data frame.
- **tdomtottotnm**: String. If tdomtottot=TRUE, the variable name for the total column in output data frame. If NULL, the default will be tdomvar + 'TOT'.
- **FIAname**: Logical. If TRUE, changes names of columns for SPCD and SPGRPCD from code to FIA names.
- **addseed**: Logical. If TRUE, add seedling counts to tree counts. Note: tdomvar must be 'SPCD' or 'SPGRPCD'.
- **pivot**: Logical. If TRUE, tdomvar data are transposed (pivoted) to separate columns.
- **presence**: Logical. If TRUE, an additional table is output with tree domain values as presence/absence (1/0).
- **proportion**: Logical. If TRUE and pivot=TRUE, an additional table will be output with tree domain data as proportions of total tsumvar.
- **cover**: Logical. If TRUE and pivot=TRUE, an additional table will be output with tree domain data as percent cover, based on proportions of tsumvar (see proportion) and tree canopy cover variable in cond (LIVE_CANOPY_CVR_PCT) or in plt (CCLIVEPLT). Does not include seedlings.
- **getadjplot**: Logical. If TRUE, adjustments are calculated for nonsampled conditions on plot.
- **adjtree**: Logical. If TRUE, trees are individually adjusted by adjustment factors. Adjustment factors must be included in tree table (see adjvar).
- **adjvar**: String. If adjtree=TRUE, the name of the variable to use for multiplying by adjustment (e.g., tadjfac).
- **NAto0**: Logical. If TRUE, convert NA values to 0.
- **adjTPA**: Numeric. A tree-per-acre adjustment. Use for DESIGNCD=1 (annual inventory), if using less than 4 subplots. If using only 1 subplot for estimate, adjTPA=4. The default is 1.
- **tround**: Number. The number of digits to round to. If NULL, default=6.
- **checkNA**: Logical. If TRUE, checks if NA values exist in necessary variables.
- **returnDT**: Logical. If TRUE, returns data.table object(s). If FALSE, returns data.frame object(s).
- **savedata**: Logical. If TRUE, saves data to outfolder.
- **savedata_opts**: List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = 'tdomsum'.
- **gui**: Logical. If gui, user is prompted for parameters.
Details

If variable = NULL, then it will prompt user for input.

If you want to get trees-per-acre information aggregated to plot or condition level, you need to include a TPA variable in tree table.

For tsumvars = GROWCFGS, GROWBFSL, GROWCFAL, FGROWCFGS, FGROWBFSL, or FGROWCFAL, you must have TPAGROW_UNADJ
For tsumvars = MORTCFGS, MORTBFSL, MORTCFAL, FMORTCFGS, FMORTBFSL, or FMORTCFAL, you must have TPAMORT_UNADJ
For tsumvars = REMVCFGS, REMVBFSL, REMVCFAL, FREMVCFGS, FREMVBFSL, or FREMVCFAL, you must have TPAREMV_UNADJ

If you want to adjust plot-level information by condition proportions, you need to include CONDID & CONDPROP_UNADJ in cond or tree table.

If you want to adjust the aggregated tree data by the area of the strata (estimation unit), you need to either have a variable in your tree data named adjfact or you need to included the following variables in your datasets:
Condition table: STATECD, CONDID, STRATA, ESTUNIT, SUBPPROP_UNADJ, MICRPROP_UNADJ (if microplot trees) MACRPROP_UNADJ (if macroplot trees).
Tree table: TPA_UNADJ

All trees where DIA=NA are removed from analysis. These are trees that were remeasured but are no longer in inventory (ex. a tree that is dead and not standing in the current inventory).

Value

tdomdata - a list of the following objects:

  tdomdat   Data frame. Plot or condition-level table with aggregated tree domain (tdom) attributes (filtered).
  tdomsum   Data frame. The tdom look-up table with data aggregated by species.
  tdomvar   String. Name of the tdom variable used to aggregate by.
  tsumvar   String. Name of the aggregated output variable.
  tdomlst   Vector. List of the aggregated tree data in tdomdat.
  tdomdat.pres Data frame. Plot or condition-level table with aggregated tree domain attributes represented as presence/absence (1/0).
  tdomdat.prop Data frame. Plot or condition-level table with aggregated tree domain attributes represented as proportion of total by plot.
  tdomdat.cov Data frame. Plot or condition-level table with aggregated tree domain attributes represented as percent cover, multiplying cover attribute by tdom proportion by plot.

If savedata=TRUE
- tdomdat will be saved to the outfolder (‘tdomprefix’_DAT.csv).
- a text file of input parameters is saved to outfolder (‘outfn’_parameters_‘date’.txt).
- if presence=TRUE, tdomdat.prop is saved to outfolder (‘tdomprefix’_PRESDAT.csv) - if proportion=TRUE, tdomdat.prop is saved to outfolder (‘tdomprefix’_PROPDAT.csv) - if cover=TRUE, tdomdat.prop is saved to outfolder (‘tdomprefix’_COVDAT.csv)
Note

This function can be used to get tree domain data. This data can be used for mapping tree domain distributions.

Author(s)

Tracey S. Frescino

Examples

# Sum of Live Basal Area Per Acre by Species
datSumTreeDom(tree = FIESTA::WYtree,
    cond = FIESTA::WYcond,
    plt = FIESTA::WYplt,
    puniqueid = "CN",
    bycond = FALSE,
    tsumvar = "BA",
    TPA = TRUE,
    tdomtot = TRUE,
    tdomtotnm = "BA_LIVE",
    tdomprefix = "BA_LIVE",
    tround = 2,
    tfilter = "STATUSCD==1")

# Sum of Number of Live Trees by Species

datSumTreeDom(tree = FIESTA::WYtree,
    cond = FIESTA::WYcond,
    plt = FIESTA::WYplt,
    puniqueid = "CN",
    bycond = FALSE,
    tsumvar = "PLT_CN",
    TPA = TRUE,
    tdomtot = TRUE,
    tdomtotnm = "CNT",
    tdomprefix = "CNT",
    tround = 0,
    tfilter = "STATUSCD==1")

# Sum of Number of Live Trees by Species, Including Seedlings

datSumTreeDom(cond = WYcond,
    plt = WYplt,
    seed = WYseed,
    puniqueid = "CN",
    bycond = FALSE,
    tsumvar = "PLT_CN",
    TPA = TRUE,
    tdomtot = TRUE,
    tdomprefix = "CNT",
    tround = 0)
DBgetCSV

Database - Extracts data table(s) from FIA DataMart.

Description

Downloads and extracts compressed comma-delimited file(s) (*.zip) from FIA DataMart (https://apps.fs.usda.gov/fia/datamart/CSV/datamart_csv.html). Only 1 table can be specified, but multiple states may be included.

Usage

```
DBgetCSV(
  DBtable, 
  states = NULL, 
  returnDT = FALSE, 
  stopifnull = TRUE, 
  noIDate = TRUE
)
```

Arguments

- **DBtable**: String. Name of table to download. Only 1 table allowed.
- **states**: String or numeric vector. Name (e.g., "Arizona", "New Mexico") or code (e.g., 4, 35) of states to download data. If NULL, tables that are not state-level are downloaded.
- **returnDT**: Logical. If TRUE, a data table is returned, else, a data frame.
- **stopifnull**: Logical. If TRUE, stop if table is NULL.
- **noIDate**: Logical. If TRUE, do not include columns with type IDate.

Details

The compressed data files are downloaded from FIA DataMart; saved to a temporary space; extracted and imported; and deleted from temporary space. Accessibility and download time depends on access and speed of internet connection.

Value

Returns a data table (returnDT=TRUE), or data.frame (returnDT=FALSE) of downloaded table(s). If more than one state, returned as one table.

Author(s)

Tracey S. Frescino
Examples

```r
## Not run:
# Get reference table for FIA research stations
DBgetCSV(DBtable="ref_research_station")

# Get plot data for multiple states
FIAplots <- DBgetCSV("PLOT", c("Georgia", "Utah"))
table(FIAplots$STATECD)

## End(Not run)
```

---

**DBgetEvalid**

Database - Gets or checks FIA EVALIDs and/or gets inventory years from FIA’s online publicly-available DataMart (https://apps.fs.usda.gov/fia/datamart/CSV/datamart_csv.html).

**Description**

Extracts FIA EVALIDs for identifying an estimation group of plots. EVALIDs may be extracted by most current evaluation (evalCur=TRUE) or by the end year of an evaluation (evalEndyr) or all evaluations in the database for one or more states. See details for more information.

**Usage**

```r
DBgetEvalid(
  states = NULL,
  RS = NULL,
  datasource = "datamart",
  data_dsn = NULL,
  invtype = "ANNUAL",
  evalCur = TRUE,
  evalEndyr = NULL,
  evalid = NULL,
  evalAll = FALSE,
  evalType = "VOL",
  invyrtab = NULL,
  ppsanm = "pop_plot_stratum_assgn",
  gui = FALSE
)
```

**Arguments**

- **states** String or numeric vector. Name (e.g., 'Arizona', 'New Mexico') or code (e.g., 4, 35) of state(s) for evalid. If all states in one or more FIA Research Station is desired, set states=NULL and use RS argument to define RS.
- **RS** String vector. Name of research station(s) ('RMRS', 'SRS', 'NCRS', 'NERS', 'PNWRS'). Do not use if states is populated.
**DBgetValid**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>datasource</td>
<td>Source of data (’datamart’, ’sqlite’).</td>
</tr>
<tr>
<td>data_dsn</td>
<td>If datasource=’sqlite’, the file name (data source name) of the sqlite database (*.sqlite).</td>
</tr>
<tr>
<td>invtype</td>
<td>String. The type of FIA data to extract (’PERIODIC’, ’ANNUAL’). See further details below.</td>
</tr>
<tr>
<td>evalCur</td>
<td>Logical. If TRUE, the most current evaluation is extracted for state(s).</td>
</tr>
<tr>
<td>evalEndyr</td>
<td>Number. The end year of the evaluation period of interest. Selects only sampled plots and conditions for the evaluation period. If more than one state, create a named list object with evalEndyr labeled for each state (e.g., list(Utah=2014, Colorado=2013).</td>
</tr>
<tr>
<td>evalid</td>
<td>Integer. One or more EVALID to check if exists.</td>
</tr>
<tr>
<td>evalAll</td>
<td>Logical. If TRUE, gets all EVALIDs for invtype.</td>
</tr>
<tr>
<td>evalType</td>
<td>String vector. The type(s) of evaluation of interest (’ALL’, ’CURR’, ’VOL’, ’GRM’, ’P2VEG’, ’DWM’, ’INV’, ’REGEN’, ’CRWN’). The evalType ’ALL’ includes nonsampled plots; ’CURR’ includes plots used for area estimates; ’VOL’ includes plots used for area and/or tree estimates; The evalType ’GRM’ includes plots used for growth, removals, mortality, and change estimates (eval_typ are accepted. See details below and FIA database manual for regional availability and/or differences.</td>
</tr>
<tr>
<td>invyrtab</td>
<td>Data frame. A data frame including inventory years by state. If NULL, it is generated from SURVEY table from FIA database based on states and invtype.</td>
</tr>
<tr>
<td>ppsanm</td>
<td>String. Name of pop_plot_assgn_layer in database.</td>
</tr>
<tr>
<td>gui</td>
<td>Logical. If TRUE, gui windows pop up for parameter selection.</td>
</tr>
</tbody>
</table>

**Details**

FIA Evaluation

An Evaluation defines a group of plots in the FIA Database used for state-level estimates, representing different spans of data and different stratification and area adjustments. An Evaluation Type (evalType) is used to identify a specific set of plots for a particular response to be able to ensure a sample-based estimate for a population. See FIA’s Database documentation for current available Evaluation Types and descriptions (https://www.fia.fs.fed.us/library/database-documentation/index.php).

**EVALID**

An EVALID is a unique code defining an Evaluation, generally in the format of a 2-digit State code, a 2-digit year code, and a 2-digit Evaluation Type code.

**EVAL_TYP**

<table>
<thead>
<tr>
<th>EVALIDCD</th>
<th>EVAL_TYP</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>EXPALL</td>
<td>All area</td>
</tr>
<tr>
<td>01</td>
<td>EXPVOL/EXPCURR</td>
<td>Area/Volume</td>
</tr>
<tr>
<td>03</td>
<td>EXPCHNG/EXPGROW/EXPMORT/EXPREMV</td>
<td>Area Change/GRM</td>
</tr>
<tr>
<td>07</td>
<td>EXPDWM</td>
<td>DWM</td>
</tr>
<tr>
<td>08</td>
<td>EXPREGEN</td>
<td>Regeneration</td>
</tr>
<tr>
<td>09</td>
<td>EXPINV</td>
<td>Invasive</td>
</tr>
</tbody>
</table>
### Value

A list of the following objects:

- **states**: String vector. State names.
- **rslst**: String vector. FIA research station names included in output.
- **evalidlist**: Named list. `evalid` by state.
- **invtype**: String. Inventory type for states(s) (ANNUAL/PERIODIC).
- **invyrtab**: Data frame. Inventory years by state for `evalidlist`.
- **evalTypelist**: Named list. Evaluation type(s) by state.
- **invyrs**: Named list. Inventory years by state for `evalidlist`.
- **SURVEY**: Data frame. If `returnPOP=TRUE`, the SURVEY table from FIADB.

### Note

FIA database tables used:

1. **SURVEY** - To get latest inventory year, `invyrtab = NULL`
2. **POP_EVAL** - To get `EVALID` and `EVALID` years

### Author(s)

Tracey S. Frescino

### Examples

```r
## Not run:
# Get `evalid` and inventory years for Wyoming
WYeval <- DBgetEval( states = "Wyoming" )
names(WYeval)

WYeval$evalidlist
WYeval$invtype
WYeval$invyrtab
WYeval$evalType
WYeval$invyrs

# Get `evalid` for Utah and Wyoming
DBgetEval( states = c("Wyoming", "Utah") )

# Get `evalid` for an FIA Research Station
RSvalid <- DBgetEval( RS = "NERS" )
names(RSvalid)
RSvalid$evalidlist

## End(Not run)
```
DBgetPlots

Description

Extracts data from FIA’s online publicly-available DataMart (https://apps.fs.usda.gov/fia/datamart/CSV/datamart_csv.html).

Usage

```r
DBgetPlots(
  states = NULL,
  datasource = "datamart",
  data_dsn = NULL,
  RS = NULL,
  invtype = "ANNUAL",
  evalid = NULL,
  evalCur = FALSE,
  evalEndyr = NULL,
  evalAll = FALSE,
  evalType = "VOL",
  measCur = FALSE,
  measEndyr = NULL,
  allyrs = FALSE,
  invyrs = NULL,
  measyrs = NULL,
  getxy = TRUE,
  xymeascUr = FALSE,
  istree = FALSE,
  isseed = FALSE,
  isveg = FALSE,
  issubp = FALSE,
  islulc = FALSE,
  isdwm = FALSE,
  plotgeom = FALSE,
  othertables = NULL,
  issp = FALSE,
  scond = FALSE,
  scondid1 = FALSE,
  defaultVars = TRUE,
  regionVars = FALSE,
  regionVarsRS = "RMRS",
  ACI = FALSE,
  subcycle99 = FALSE,
  intensity1 = FALSE,
  stateFilter = NULL,
  allFilter = NULL,
  allltFilter = NULL,
)```
DBgetPlots

  returndata = TRUE,
  savedata = FALSE,
  saveqry = FALSE,
  savePOP = FALSE,
  savedata_opts = NULL
)

Arguments

states String or numeric vector. Name (e.g., ‘Arizona’, ‘New Mexico’) or code (e.g., 4, 35) of state(s) for evalid. If all states in one or more FIA Research Station is desired, set states=NULL and use RS argument to define RS.
datasource String. Source of data (‘datamart’, ‘sqlite’).
data_dsn String. If datasource=’sqlite’, the name of SQLite database (*.sqlite).
RS String vector. Name of research station(s) to get public XY coordinates for (‘RMRS’, ‘SRS’, ‘NCRS’, ‘NERS’, ‘PNWRS’). Do not use if states is populated. See FIESTA::ref_statecd for reference to RS and states.
invtype String. Type of FIA inventory to extract (‘PERIODIC’, ‘ANNUAL’). Only one inventory type (PERIODIC/ANNUAL) at a time.
evalid Integer. Inventory span defining variable. Extract data for a specific FIA Evaluation (See details for more information about FIA Evaluations).
evalCur Logical. Inventory span defining variable. If TRUE, extract data for the most current FIA Evaluation for each state.
evalEndyr YYYY. Inventory span defining variable. Extract data for the FIA Evaluation(s) ending in the specified evalEndyr(s). If more than one state and different Evaluations by state are desired, input a named list.
evalAll Logical. Inventory span defining variable. If TRUE, extract data for all FIA Evaluations for each state.
evalType String vector. The type(s) of FIA Evaluation of interest (‘ALL’, ‘CURR’, ‘VOL’, ‘GRM’, ‘P2VEG’, ‘DWM’, ‘INV’, ‘REGEN’, ‘CRWN’). The evalType ‘ALL’ includes nonsampled plots; ‘CURR’ includes plots used for area estimates; ‘VOL’ includes plots used for area and/or tree estimates; ‘GRM’ includes plots used for growth, removals, mortality, and change estimates (evalType are accepted. See details below and FIA database manual for regional availability and/or differences. Note: do not use if EVAlID is specified.
measCur Logical. Inventory span defining variable. If TRUE, extract plots with most current sampled measurement for state(s).
measEndyr Logical. Inventory span defining variable. If TRUE, extract plots with most current sampled measurement for state(s) for years measured in or before measEndyr.
allyrs Logical. Inventory span defining variable. If TRUE, extract all annual inventory years in database for each state.
invyrs YYYY vector. Inventory span defining variable. Extract data by state for the specified inventory year(s) (e.g., c(2000, 2001, 2002)). If more than one state and different inventory years are desired, input a named list object with years labeled by state (e.g., list(Utah=2000:2009, Colorado=c(2002,2003,2005)).
measyrs  YYYY vector. Measurement year span defining variable. Extract data by state for the specified measurement year(s) (e.g., c(2000, 2001, 2002)). If more than one state and different measurement years are desired, input a named list object with years labeled by state (e.g., list(Utah=2000:2009, Colorado=c(2002,2003,2005)).

getxy   Logical. If TRUE, gets separate XY table.

xymeasCur Logical. If TRUE, include XY coordinates from the most current sampled measurement of each plot.

istree  Logical. If TRUE, tree data are extracted from TREE table in database.

isseed  Logical. If TRUE, seedling data are extracted from SEEDLING table in database.

isveg   Logical. If TRUE, understory vegetation tables are extracted from FIA database (P2VEG_SUBPLOT_SPP, P2VEG_SUBP_STRUCTURE, INVASIVE_SUBPLOT_SPP).

issubp  Logical. If TRUE, subplot tables are extracted from FIA database (SUBPLOT, SUBP_COND).

islulc  Logical. If TRUE, condition-level land use/land cover data are for current and previous years are extracted from FIA database.

isdwm   Logical. If TRUE, summarized condition-level down woody debris data are extracted from FIA database (COND_DWM_CALC).

plotgeom Logical. If TRUE, variables from the PLOTGEOM table are appended to the plot table.

otherTables String Vector. Name of other table(s) in FIADB to include in output. The table must have PLT_CN as unique identifier of a plot.

issp    Logical. If TRUE, an sf spatial object is generated from the public X/Y coordinates in the plot table.

spcond  Logical. If TRUE, a set of condition-level attributes (e.g., FORTYPCD) represented at the plot-level are extracted from FIA DataMart COND table. (See Notes for more info on how condition attributes were added).

spcondid1 Logical. If TRUE and issp=TRUE and spcond=TRUE, condition variables are determined by condition 1 attributes. If FALSE, an algorithm is used to select the condition to use (See details for algorithm used).

defaultVars Logical. If TRUE, a set of default variables are selected in query. See notes for variable descriptions.

regionVars Logical. If TRUE, regional variables are included in query (e.g., SDI_RMRS, SDIPCT_RMRS, SDIMAX_RMRS, QMD_RMRS).

regionVarsRS String. Region for regionVars ('RMRS', 'SRS', 'NCRS', 'NERS', 'PNWRS').

ACI Logical. If TRUE, the data from All Condition Inventories (ACI) are included in dataset (NF_SAMPLING_STATUS_CD = 1). See below for more details.

subcycle99 Logical. If TRUE, excludes plots with SUBCYCLE = 99. These plots are plots that are measured more than once and are not included in the estimation process.

intensity1 Logical. If TRUE, includes only plots where INTENSITY = 1.

stateFilter Character string or Named list. Logical statement to use as plot and filter in sql query. Must include plot alias (‘p.’) and be sql syntax (e.g., ‘p.COUNTYCD = 1’). If more than 1 state, stateFilter must be a named list with names as state(s) (e.g., list(Utah='p.COUNTYCD = 1').
allFilter String. An overall filter for plot or condition data in all states in query. The expression must be R syntax (e.g., 'PLOT_STATUS_CD == 1').

alltFilter String. If istree=TRUE, an overall filter for tree data in all states (e.g., only Whitebark pine trees - 'SPCD == 101'). Note: returns only plots with trees included in filter.

returndata Logical. If TRUE, returns data objects.

savedata Logical. If TRUE, saves data to outfolder as comma-delimited file (*.csv). No objects are returned. If FALSE, the data are saved as R objects and returned to user. See details for caveats.

saveqry Logical. If TRUE, saves queries to outfolder (by state).

savePOP Logical. If TRUE, save and return the POP_PLOT_STRATUM_ASSGN table.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL,

Details

FIA forest land definition

Current
Forested plots include plots with >= 10 percent cover (or equivalent stocking) by live trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. To qualify, the area must be >= 1.0 acre in size and 120.0 feet wide (See Burrill et al. 2018).

*ACI (All Condition Inventory)*
RMRS National Forest plots. For nonforest conditions that have been visited in the field (NF_SAMPLING_STATUS_CD = if trees exist on the condition, the data exist in the tree table. If you do not want these trees included, ACI=FALSE. This will filter the data to only forested conditions (COND_STATUS_CD = 1)

*Nevada*
In 2016, the population area of Nevada changed to exclude the large restricted area owned by Department of Defense (Area 51) from the sample. Prior to 2016, the plots within this area were observed using aerial photos and if they were definitely nonforest the plots were entered in the database with nonforest information. If they were observed as forested or potentially forested, they were given a PLOT_STATUS_CD=3 because they were Denied Access. From 2016 on, all plots within this area are removed from the sample, and thus, removed from database.

FIA DataMart Data
FIA data available on FIA DataMart include the following information.

- the PLOT variable is renumbered.
- the LON/LAT coordinates are fuzzed & swapped.
- the OWNERCD variable is based on fuzzed & swapped locations.
- ECOSUBCD, CONGCD, ELEV, and EMAP_HEX are GIS-extracted values based on fuzzed & swapped locations.
- For annual data, forested plots represent the current definition of >= 10 percent cover...
- For periodic data, forested plots are defined by a definition of Other Wooded Land (OWL), including >= 5 percent cover...
*FIADB Table Extraction*

**Argument Table Name(s)**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Table Name(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>istree</td>
<td>TREE</td>
</tr>
<tr>
<td>isveg</td>
<td>P2VEG_SUBPLOT_SPP, P2VEG_SUBP_STRUCTURE, INVASIVE_SUBPLOT_SPP</td>
</tr>
<tr>
<td>issubp</td>
<td>SUBPLOT, SUBP_COND</td>
</tr>
<tr>
<td>isdwm</td>
<td>COND_DWM_CALC</td>
</tr>
<tr>
<td>isgrm</td>
<td>TREE_GRM_COMPONENT</td>
</tr>
<tr>
<td>issccm</td>
<td>SUBP_COND_CHNG_MTRX</td>
</tr>
</tbody>
</table>

**FIA Evaluations**

An evaluation is a group of plots within the FIA database that is used for generating population estimates, representing different inventory spans of data with different stratification or area adjustments. Each evaluation is determined by the type of estimation (evalType) including: area and tree estimates; growth, removal, and mortality estimates; and area change estimates (EVAL_TYPE). These plots are identified by an evalid, which is a unique identifier in the format of a 2-digit State code, a 2-digit year code, and a 2-digit evaluation type code. For example, EVALID '491601' represents the Utah 2016 evaluation for current area estimates.

**FIA Evaluation Types**

Define one or more Evaluation Type for evalCur=TRUE or evalEndyr=YYYY. An Evaluation type is used to identify a specific set of plots for a particular response that can be used to make a statistically valid sample-based estimate. If evalType="ALL", the evaluation includes all sampled and nonsampled plots or plots that were missed in an inventory year.

Regional differences may occur on how missed plots are represented in a FIA Evaluation. For example, RMRS Evaluations are static; missed plots are included in an Evaluation as nonsampled, and when measured, are included in a following Evaluation. Therefore, the number of nonsampled plots in previous Evaluations may change, depending on when missed plot are measured. In the PNW Research Station, plots are brought forward to replace missed plots in an evaluation, depending on the evalType.

**EVAL_TYP**

<table>
<thead>
<tr>
<th>EVALIDCD</th>
<th>EVAL_TYP</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>EXPALL</td>
<td>All area</td>
</tr>
<tr>
<td>01</td>
<td>EXPVOL/EXPCURR</td>
<td>Area/Volume</td>
</tr>
<tr>
<td>03</td>
<td>EXPCHNG/EXPGROW/EXPMORT/EXPREMV</td>
<td>Area Change/GRM</td>
</tr>
<tr>
<td>07</td>
<td>EXPDWM</td>
<td>DWM</td>
</tr>
<tr>
<td>08</td>
<td>EXPREGEN</td>
<td>Regeneration</td>
</tr>
<tr>
<td>09</td>
<td>EXPINV</td>
<td>Invasive</td>
</tr>
<tr>
<td>10</td>
<td>EXPP2VEG</td>
<td>Veg profile</td>
</tr>
<tr>
<td>12</td>
<td>EXPCRWN</td>
<td>Crown</td>
</tr>
</tbody>
</table>

**Inventory span defining variables**
Data can be extracted using FIA Evaluations or a custom-defined Evaluation for one or more states, one or more FIA Research Stations (RS), or all available states in database (states=NULL, RS=NULL).

*FIA Evaluation*

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVALID</td>
<td>Specified FIA EVALID (e.g., 491801)</td>
</tr>
<tr>
<td>evalCur</td>
<td>Most current FIA Evaluation</td>
</tr>
<tr>
<td>evalEndyr</td>
<td>End year of an FIA Evaluation (e.g., 2018)</td>
</tr>
<tr>
<td>evalAll</td>
<td>All evaluations in database</td>
</tr>
<tr>
<td>evalType</td>
<td>Type of FIA Evaluation (response)</td>
</tr>
</tbody>
</table>

*Custom evaluation*

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>measCur</td>
<td>Most current measurement of plot in database</td>
</tr>
<tr>
<td>measEndyr</td>
<td>Most current measurement of plot in database in or before year</td>
</tr>
<tr>
<td>allyrs</td>
<td>All years for invtype (ANNUAL/PERIODIC)</td>
</tr>
<tr>
<td>invyrs</td>
<td>Specified inventory years (e.g., 2015:2018)</td>
</tr>
</tbody>
</table>

**Spatial data**

If issp=TRUE, an sf spatial object of plot-level attributes is generated from public coordinates, with NAD83 Geographic Coordinate Reference System.

*Exporting*

If savedata=TRUE and out_fmt="shp", the spatial object is exported to the outfolder using the ESRI Shapefile driver. The driver truncates variable names to 10 characters or less. Variable names are changed using an internal function. The name changes are written to a csv file and saved to the outfolder (shpfile_newnames.csv).

*spcond*

Only one condition per plot is used for spatial representation of condition attributes. IF CONDID1=TRUE, condition 1 is selected. If CONDID1=FALSE, the condition is selected based on the following criteria. A column named CONDMETHOD is added to the attribute table to show the method and steps used, identified by the abbreviation in parentheses.

1. minimum COND_STATUS_CD (_ST)
2. maximum condition proportion (_CP)
3. maximum live_canopy_cvr_pct (_CC)
4. minimum STDSZCD (_SZ)
5. minimum CONDID (_C1)

**Derived Variables**
If defaultVars=TRUE, the following derived variables are calculated after extracting data from the FIA database.

Plot-level variables:

- **NBRCND** - Number of conditions on plot, including nonsampled conditions (COND_STATUS_CD = 5)
- **NBRCNDSAMP** - Number of sampled conditions on plot.
- **NBRCNDFOR** - Number of sampled forested conditions on plot.
- **NBRCNDFTP** - Number of sampled forested conditions with different forest types on plot.
- **NBRCNDFGRP** - Number of sampled forested conditions with different forest type groups on plot.
- **CCLIVEPLT** - Percent live canopy cover of condition aggregated to plot-level (LIVE_CANOPY_CVR_PCT * CONDPROP_UNADJ).
- **PLOT_ID** - Unique Identifier for a plot (‘ID’ + STATECD(2) + UNITCD(2) + COUNTYCD(3) + PLOT(5)). This variable can be used to identify multiple records for each measurement of plot.

Condition-level variables:

- **FORTYPGRP** - TYPGRPCD merged to FORTYPCD
- **FLDTYPGRP** - TYPGRPCD merged to FLDTYPCD
- **FORTYPESAMP** - Combination of PLOT_STATUS_CD and PLOT_NONSAMPLE_REASN_CD
- **QMD** - Quadratic Mean Diameter

Tree-level variables:

- **BA** - the basal area of a tree ($BA = DIA \times DIA \times 0.005454$)

TREE AGE Notes:
- Available for live timber and woodland trees in the following states: AZ, CO, ID, MT, NV, UT, OR, WA.
- **BHAGE** - Breast height age (4.5' above ground) of timber trees.
- **PNW** - one tree is sampled for each species, within each crown class, and for each condition class present on plot. Age of live trees from 1.0” DIA upwards (Above ground age). No affiliation with diameter class.
- **RMRS** - one tree is sampled for each species and broad diameter class present on plot.

DRYBIO Notes:
- **DRYBIO_AG** - Aboveground oven-dry biomass, in pounds ($DRYBIO_AG = (DRYBIO_BOLE + DRYBIO_STUMP + DRYBIO_TOP + DRYBIO_SAPLING + DRYBIO_WDLD_SPP$).
- Available for both timber and woodland species, live trees ≥ 1.0” DIA and dead trees ≥ 5.0” DIA. Summed dry biomass of overstory trees above 1.5” DIA.
- **DRYBIO_BOLE** - dry biomass of sound wood in live and dead trees, including bark, from a 1-foot stump to a min 4-inch top DIA.
- **DRYBIO_STUMP** - dry biomass in the tree stump, including the portion of the tree from the ground to the bottom of merchantable bole, 1-foot.
- **DRYBIO_TOP** - dry biomass in the top of the tree, including the portion of the tree above merchantable bole, 4-inch top, and all branches, excludes foliage.
- **DRYBIO_SAPLING** - dry biomass of saplings, including aboveground portion, excluding foliage, of live timber trees ≥1.0” and <5.0” DIA.
- **DRYBIO_WDLD_SPP** - dry biomass of woodland trees, live or dead, including the aboveground portion, excluding foliage.
ABOVEGROUND CARBON ESTIMATES (IN POUNDS)
Available for both timber and woodland species, live trees >= 1.0" DIA and dead trees >= 5.0" DIA.
Calculated as 1/2 of the aboveground estimates of biomass:

\[
\text{CARBON}_{\text{AG}} = 0.5 \times (\text{DRYBIO}_{\text{AG}})
\]

TREE AGE DATA ONLY IN FOR ("AZ", "CO", "ID", "MT", "NV", "UT")
FMORTCFAL includes trees >= 5.0" DIA and greater and is not populated for states ("CA", "OR", "WA", "OK")

Mortality variables only available in: AZ, CO, ID, MT, NV, NM, UT, WY, ND, SD, NE, KS, OK.

**TPA** If TPA=TRUE and istree=TRUE or isseed=TRUE, the following tree/seedling variables are multiplied by trees-per-acre (TPA\_UNADJ). TPA\_UNADJ is set to a constant derived from the plot size and equals 6.018046 for trees sampled on subplots, 74.965282 for trees sampled on microplots, and 0.999188 for trees sampled on macroplots. Variable-radius plots were often used in earlier inventories, so the value in TPA\_UNADJ decreases as the tree diameter increases (FIADB User Guide)

Variables: VOLCFNET, VOLCFGRS, GROWCFGS, GROWCFAL, FGROWCFGS, FGROWCFAL, MORTCFGS, MORTCFAL, REMCFGS, REMCFAL, REMCFVL, REMVCFGS, REMVCFAL, DRYBIO\_BOLE, DRYBIO\_STUMP, DRYBIO\_TOP, DRYBIO\_SAPLING, DRYBIO\_WDLD\_SPP, DRYBIO\_BG, CARBON\_BG, CARBON\_AG

**MISC**
For regions outside RMRS, there is no OWNCD attached to nonforest lands.

**Value**

if returndata=TRUE, a list of the following objects:

- **states** Vector. Input state(s) (full state names: Arizona).
- **tabs** List. A list of data frames from FIA database, including plt and cond; and tree (if istree=TRUE); seed (if isseed=TRUE). See below 'Output Tables - FIA Table Names' for reference to FIA database tables. See FIESTA::ref_* for variable descriptions (e.g., FIESTA::ref_tree). If istree and the number of states > 3, tree data are saved to outfolder and not returned to accommodate R memory issues.
- **xy\_*\_PUBLIC** Data frame. XY data from FIA's public database. If measCur=TRUE, named xy\_Cur\_PUBLIC, else named xy\_PUBLIC. The data frame has 10 columns ("PLT\_CN", "LON\_PUBLIC", "LAT\_PUBLIC", "STATECD", "UNITCD", "COUNTYCD", "PLOT", "INTENSITY", "PLOT\_ID" (ID+STATECD+UNITCD+COUNTRYCD+PLOT), "COUNTYFIPS"). If issp=TRUE, returns an sf object.
- **spconddat** If spcond=TRUE, the condition variables representing each plot for spatial display. For plots with multiple conditions, the selected condition is based on CONDID=1 (if spcondid1=TRUE) or a set if criteria defined in Details - spcond (if spcondid1=FALSE).
- **evalid** Number. If evalCur=TRUE or evalEndyr is not NULL, the Evaluation ID from the FIA database used to define the output data.
- **pltcnt** Data frame. Number of plots (NBRPLOTS) by state, cycle, inventory year, and plot status.
pop_plot_stratum_assgn

Data frame. If savePOP=TRUE, and FIA Evaluations are used to extract data from database, return the POP_PLOT_STRATUM_ASSGN table or, if more than one evalType and savePOP=FALSE. If more than one evalType, only the records for the evalTypes are returned, otherwise all evalTypes for the state evaluation are returned.

*Output Tables - FIA Table Names*

<table>
<thead>
<tr>
<th>tab</th>
<th>FIA Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>plt</td>
<td>plot</td>
</tr>
<tr>
<td>cond</td>
<td>cond</td>
</tr>
<tr>
<td>tree</td>
<td>tree</td>
</tr>
<tr>
<td>p2veg_subplot_spp</td>
<td>P2VEG_SUBPLOT_SPP</td>
</tr>
<tr>
<td>p2veg_subp_structure</td>
<td>P2VEG_SUBP_STRUCTURE</td>
</tr>
<tr>
<td>invsubp</td>
<td>INVASIVE_SUBPLOT_SPP</td>
</tr>
<tr>
<td>subplot</td>
<td>SUBPLOT</td>
</tr>
<tr>
<td>subp_cond</td>
<td>SUBP_COND</td>
</tr>
<tr>
<td>dwm</td>
<td>COND_DWM_CALC</td>
</tr>
<tr>
<td>grm</td>
<td>TREE_GRM_COMPONENT</td>
</tr>
<tr>
<td>issccm</td>
<td>SUBP_COND_CHNG_MTRX</td>
</tr>
</tbody>
</table>

# Outputs to outfolder (if savedata=TRUE):

- If saveqry=TRUE, text file(s) of SQL queries used to extract data from database (_txt). Note: one query is used for extracting both plt and cond (pltcondqry*.txt).
- CSV file of plot and condition counts (pltcnt*.txt).
- Layers in a database or CSV files of output tables.
- If issp=TRUE, a feature class or ESRI shapefile of plot-level level attributes. If shapefile (.shp), variable names are truncated to 10 characters or less. See notes for more info.
- If issp=TRUE and out_fmt='sqlite', the SQLite data is SpatiaLite.

To deal with limitations of R object size and/or computer memory issues, if istree=TRUE and more than three states are desired, the tree data are saved to a CSV file, with no tree data object returned.

**Note**

If no parameters are included, the user is prompted for input. If partial parameters, the default parameter values are used for those not specified.

**Data Access** All data are downloaded from FIA’s publicly-available online Datamart (https://apps.fs.usda.gov/fia/datamart/CSV/datamart_csv.html).

Because of FIA’s confidentiality agreement to protect the privacy of landowners as well as protecting the scientific integrity of FIA’s sample design, the exact coordinates of the sample plot locations are not included. The X/Y coordinates (LON_PUBLIC/LAT_PUBLIC) for download are perturbed up to a mile from the original location (https://www.fia.fs.fed.us/tools-data/spatial/Policy/index.php). If the exact location of the plots are necessary for your analysis, contact FIA’s Spatial Data Services (https://www.fia.fs.fed.us/tools-data/spatial/index.php).
Author(s)
Tracey S. Frescino

References


Examples
```r
## Not run:
# Extract the most current evaluation of data for Utah
UTdat <- DBgetPlots(states = "Utah", evalCur = TRUE)

names(UTdat)
head(UTdat$plt)
UTdat$pltcnt

# Look at number of plots by inventory year
table(UTdat$plt$INVYR)

# Note: see FIESTA::ref_plt and FIESTA::ref_cond for variable descriptions
# Or consult FIA Database documentation
# \link{https://www.fia.fs.fed.us/library/database-documentation/index.php}

# Extract specified inventory years 2012:2014 and spatial information
UTdat2 <- DBgetPlots(states = "Utah", invyrs = 2012:2014, issp = TRUE)

names(UTdat2)
UTdat2$pltcnt
UTdat2$spxy_PUBLIC

# Extract and display plots with aspen forest type
UTdat3 <- DBgetPlots(states = "Utah", invyrs = 2012:2014, issp = TRUE, allFilter = "FORTYPCD == 901")

names(UTdat3)
UTdat3$pltcnt
```
DBgetSQLite

Database - Queries a SQLite database table.

Description

Extracts and queries data from a SQLite (*.sqlite) database (Note: must use SQL syntax).

Usage

DBgetSQLite(states = NULL, outfolder = NULL)

Arguments

states String. Vector of one or more state names.
outfolder String. The output folder path. If NULL, outfolder is the working directory.

Value

Returns nothing.

Author(s)

Tracey S. Frescino

Examples

## Not run:
# Extract data from Washington state
DBgetSQLite(states = "WA")

# Extract data from Utah and California, save to an outfolder
DBgetSQLite(states = c("UT", "CA"),
            outfolder = tempdir())

## End(Not run)
Description

Gets strata information from FIA’s Oracle database or FIA DataMart, including: (1) strata and estimation unit assignment per plot; (2) total area by estimation unit; (3) pixel counts and number plots by strata/estimation unit. Include a data frame of plots, states, or evaluation information.

Usage

```r
DBgetStrata(
  dat = NULL,
  uniqueid = "CN",
  datsource = "datamart",
  data_dsn = NULL,
  states = NULL,
  evalid = NULL,
  evalCur = TRUE,
  evalEndyr = NULL,
  evalAll = FALSE,
  evalType = "VOL",
  savedata = FALSE,
  getassgn = TRUE,
  pop_plot_stratum_assgn = NULL,
  savedata_opts = NULL
)
```

Arguments

- **dat**: Data frame, comma-delimited file (*.csv), or shapefile (*.shp). The strata value is merged to this table and returned as a data frame. See details for necessary variables.
- **uniqueid**: String. The unique plot identifier of dat (e.g., ’CN’).
- **datsource**: String. Source of data (‘datamart’, ‘sqlite’).
- **data_dsn**: String. If datsource=’sqlite’, the name of SQLite database (*.sqlite).
- **states**: String or numeric vector. Name(s) (e.g., ’Arizona’, ’New Mexico’) or code(s) (e.g., 4, 35) of states for strata if dat=NULL.
- **evalid**: Integer. The evaluation if dat=NULL.
- **evalCur**: Logical. If TRUE, the most current evaluation for state(s).
- **evalEndyr**: Number. The end year of evaluation time period for state(s).
- **evalAll**: Logical. If TRUE, gets all EVALIDs for invtype.
evalType  
String vector. The type(s) of evaluation of interest ('ALL', 'CURR', 'VOL', 'GRM', 'P2VEG', 'DWM', 'INV', 'REGEN', 'CRWN'). The evalType 'ALL' includes nonsampled plots; 'CURR' includes plots used for area estimates; 'VOL' includes plots used for area and/or tree estimates; The evalType 'GRM' includes plots used for growth, removals, mortality, and change estimates (eval_typ are accepted. See details below and FIA database manual for regional availability and/or differences. Note: do not use if EVALID is specified. See notes for more information on evalType.

savedata  
Logical. If TRUE, writes output to outfolder.

getassgn  
Logical. If TRUE, extracts plot assignments from pop_plot_stratum_assgn table in database.

pop_plot_stratum_assgn  
Data frame. The pop_plot_stratum_assgn for state(s).

savedata_opts  
List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

Details

Dependent packages: DBI, rgdal

The following variables must be present in dat: STATECD, UNITCD, INVYR, a uniqueid (ex. "PLT_CN"), and PLOT_STATUS_CD (if nonsampled plots in dataset).

FIADB TABLES USED:

FS_FIADB.SURVEY To get latest inventory year.
FS_FIADB.POP_EVAL To get EVALID and EVALID years.
FS_FIADB.POP_ESTN_UNIT To get total area by estimation unit (AREATOT_EU-includes water).
FS_FIADB.POP_STRATUM To get pixel counts by stratum and estimation unit.
FS_FIADB.POP_PLOT_STRATUM_ASSGN To get estimation unit & stratum assignment for each plot.

Area by estimation unit includes total area for all plots (evaltype="all").

Value

FIAstrata - a list of the following objects:

pltassgn  
Data frame. Plot-level strata/estimation unit assignment. If dat is not NULL, strata/estimation unit variables are appended to dat.

pltassgnid  
String. Name of unique identifier of plot in pltassgn.

unitarea  
Data frame. Total acres by estimation unit.

unitvar  
String. Name of the estimation unit variable (ESTN_UNIT).

areavar  
String. Name of the acre variable (ACRES).

stratalut  
Data frame. Strata look-up table with summarized pixel counts (P1POINTCNT) by strata/estimation unit.

strvar  
String. Name of the strata variable (STRATA).
strwtvar  String. Name of the strata weight variable (PIPOINTCNT).
evalid   List. evalid by state.

Outputs to outfolder (if savedata=TRUE):
- CSV file of pltassgn (*'date'.csv).
- CSV file of unitarea (*'date'.csv).
- CSV file of stratalut (*'date'.csv).
- If collapsed, a CSV file of original classes and new collapsed classes.

Note

Steps used in data extraction:

1. Get EVALID and EVALID years by state - DBgetEvalid().
2. unitarea: get total area by estimation unit for EVALID (POP_ESTN_UNIT).
3. stratalut: get pixel counts by estimation unit and stratum for EVALID (POP_STRATUM).
4. pltassgn: get estimation unit and stratum assignment for each plot for EVALID. (POP_PLOT_STRATUM_ASSGN).
5. If dat is not NULL, merge pltassgn assignment to dat.
6. Merge number of plots to stratalut
7. Check for only 1 MEASYEAR or 1 INVYR and number of plots by strata/estimation unit. If less than minimumnum plots per strata/estimation unit collapse using the following algorithm.

Strata collapsing:
If there are less than minplotnum (10) plots in the smallest strata of the estimation unit, these plots are grouped with the larger strata in the same estimation unit and defined as the highest strata value. If, after grouping, there are still less than minplotnum, all of these plots are combined with the corresponding strata of the estimation unit above. If there are no records above, then they are combined with the estimation unit below. The process repeats, grouping the strata to the highest strata value if necessary. All grouping is restrained within survey units (UNITCD).

More than one evaluation:
If attributing a table of plots and there are plots that have been visited more than once, all plots are assigned an estimation unit and strata value, but the area and strata proportions are from the most current evaluation for the dataset. The plots outside the most current evaluation are attributes with values from the next most current evaluation occurring in the database.

Author(s)

Tracey S. Frescino

Examples

```r
## Not run:
# Get strata for the most current evaluation of a state (ex. Wyoming)
WYstrat1 <- DBgetStrata(states = "Wyoming",
                         evalCur = TRUE)
names(WYstrat1)
```
head(WYstrat1$pltassgn)
WYstrat1$unitarea
WYstrat1$unitvar
WYstrat1$areavar
WYstrat1$strvar
WYstrat1$evalid

# Get strata information for a specific set of plots
WYstrat4 <- DBgetStrata(dat = WYplt)
names(WYstrat4)

head(WYstrat4$pltassgn)
WYstrat4$unitarea
WYstrat4$evalid

## End(Not run)

---

**DBgetXY**

*Database - Extracts actual plot coordinates from the FIA Oracle database.*

**Description**

Extracts actual plot coordinates for the most current measurement of each plot in the database.

**Usage**

```r
DBgetXY(
  states = NULL,
  RS = NULL,
  invtype = "ANNUAL",
  evalid = NULL,
  evalCur = FALSE,
  evalEndyr = NULL,
  evalAll = FALSE,
  evalType = "ALL",
  measCur = FALSE,
  measEndyr = NULL,
  allyrs = FALSE,
  invyrs = NULL,
  measyrs = NULL,
  intensity1 = FALSE,
  issp = FALSE,
  returndata = TRUE,
  savedata = FALSE,
  exportsp = FALSE,
  savedata_opts = NULL
)
```

Arguments

**states**
String or numeric vector. Name (e.g., 'Arizona', 'New Mexico') or code (e.g., 4, 35) of state(s) for eval. If all states in one or more FIA Research Station is desired, set states=NULL and use RS argument to define RS.

**RS**
String vector. Name of research station(s) to get public XY coordinates for ('RMRS', 'SRS', 'NCRS', 'NERS', 'PNWRS'). Do not use if states is populated. See FIESTA::ref_statecd for reference to RS and states.

**invtype**
String. Type of FIA inventory to extract ('PERIODIC', 'ANNUAL'). Only one inventory type (PERIODIC/ANNUAL) at a time.

**evalid**
Integer. Inventory span defining variable. Extract public XY coordinates for a specific FIA Evaluation (See details for more information about FIA Evaluations).

**evalCur**
Logical. Inventory span defining variable. If TRUE, extract public XY coordinates for the most current FIA Evaluation for each state.

**evalEndyr**
YYYY. Inventory span defining variable. Extract public XY coordinates for the FIA Evaluation(s) ending in the specified evalEndyr(s). If more than one state and different Evaluations by state are desired, input a named list object with evalEndyr by state (e.g., list(Utah=2014, Colorado=2013).

**evalAll**
Logical. Inventory span defining variable. If TRUE, extract public XY coordinates for all FIA Evaluations for each state.

**evalType**
String vector. The type(s) of FIA Evaluation of interest ('ALL', 'CURR', 'VOL', 'GRM', 'P2VEG', 'DWM', 'INV', 'REGEN', 'CRWN'). The evalType 'ALL' includes nonsampled plot coordinates; 'CURR' includes plot coordinates used for area estimates; 'VOL' includes plot coordinates used for area and/or tree estimates; 'GRM' includes plot coordinates used for growth, removals, mortality, and change estimates (evalType c(GROW, MORT, REMV, CHNG)). Multiple types are accepted. See details below and FIA database manual for regional availability and/or differences.

**measCur**
Logical. Inventory span defining variable. If TRUE, extract public XY coordinates with most current sampled measurement for state(s).

**measEndyr**
Logical. Inventory span defining variable. If TRUE, extract XY public coordinates with most current sampled measurement for state(s) for years measured in or before measEndyr.

**allyrs**
Logical. Inventory span defining variable. If TRUE, extract all annual inventory years in database for each state.

**invyrs**
YYYY vector. Inventory span defining variable. Extract public XY coordinates by state for the specified inventory year(s) (e.g., c(2000, 2001, 2002)). If more than one state and different inventory years are desired, input a named list object with years labeled by state (e.g., list(Utah=2000:2009, Colorado=2002,2003,2005)).

**measyrs**
YYYY vector. Measurement year span defining variable. Extract public XY coordinates by state for the specified measurement year(s) (e.g., c(2000, 2001, 2002)). If more than one state and different measurement years are desired, input a named list object with years labeled by state (e.g., list(Utah=2000:2009, Colorado=2002,2003,2005)).
DBgetXY

intensity1 Logical. If TRUE, includes only XY coordinates where INTENSITY = 1 (FIA base grid).

issp Logical. If TRUE, returns spatial XY data as a list object with query.

returndata Logical. If TRUE, returns XY data as a list object with query.

savedata Logical. If TRUE, saves XY data to outfolder as comma-delimited file (*.csv).

exportsp Logical. If TRUE, exports data as spatial.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

Value

if returndata=TRUE, a list of the following objects:


xyqry String. Query to extract coordinates.

xvar String. Name of X variable in xy*_PUBLIC.

yvar String. Name of Y variable in xy*_PUBLIC.

If savedata=TRUE, outputs the xy*_PUBLIC as out_fmt to outfolder. If exportsp=TRUE, the output xy data are saved as spatial layer.

Note

If no parameters are included, the user is prompted for input. If partial parameters, the default parameter values are used for those not specified.

Author(s)

Tracey S. Frescino

Examples

## Not run:
# Most current evaluation and shapefile with public coordinates
COxylst <- DBgetXY(states = "Colorado",
                   measCur = TRUE)
names(COxylst)

head(COxylst$xyCur_ACTUAL)
COxylst$xyqry

## End(Not run)
DBqryCSV

Description

Downloads, extracts, and queries compressed comma-delimited file(s) (*.zip) from FIA DataMart (https://apps.fs.usda.gov/fia/datamart/CSV/datamart_csv.html). (Note: must use SQL syntax).

Usage

DBqryCSV(sql, states = NULL, sqltables = NULL)

Arguments

- sql: String. A sql query. Must be appropriate sql syntax.
- states: String vector. Name of state(s) in query. If not by state, set to NULL.
- sqltables: String vector. Name of table(s) in sql statement to download. The sqltables must match tables in the sql statement (i.e., case-sensitive).

Details

The compressed data files are downloaded from FIA DataMart; saved to a temporary space; extracted and imported; and deleted from temporary space. Accessibility and download time depends on access and speed of internet connection.

Value

Returns a data frame from resulting query.

Author(s)

Tracey S. Frescino

Examples

```r
## Not run:
# Number of plots by inventory year for the state of Wyoming
sql <- "select INVYR, count(*) AS NBRPLOTS
        from plot
        where statecd=56 group by INVYR"
DBqryCSV(sql = sql,
        states = "Wyoming",
        sqltables = "plot")

## End(Not run)
```
Description

Reference tables - NODATA Values.

Usage

DEFAULT_NODATA

Format

An object of class list of length 7.

Source

gdal values.

ecomap

EcoMap SpatialPolygonsDataFrame

Description

Contains regional geographic delineations for analysis of ecological relationships across ecological units. ECOMAP is the term used for a USDA Forest Service initiative to map ecological units and encourage their use in ecosystem-based approaches to forest land conservation and management. It is coordinated at the national and regional levels by USDA Forest Service staff and implemented in cooperation with State forestry agencies and others. ECOMAP mapping criteria are outlined in the National Hierarchical Framework of Ecological Units (https://www.ncrs.fs.fed.us/gla/reports/hierarchy.htm). The framework systematically divides the country into progressively smaller areas of land and water that have similar physical and biological characteristics and ecological processes.

Format

A SpatialPolygonsDataFrame with 1233 features and 3 attributes PROVINCE - Ecomap Province SECTION - EcoMap Section SUBSECTION - Ecomap Subsection

Details

The EcoMap Provinces feature class contains ecological province polygons attributed with names and descriptions. The EcomapSections 2007 data set describes the ecological sections within the conterminous United States. The EcomapSubsections 2007 data set describes the ecological subsections within the conterminous United States.

Converted to simple feature
Transformed CRS from longlat(EPSG:4269) to Albers (EPSG:5070)
Saved to R object, with compression='xz'
Source

Downloaded from the FSGeodata Clearinghouse on 2019 October 30, format ESRI geodatabase (https://data.fs.usda.gov/geodata/edw/datasets.php)

References


---

**GDT_NAMES**

Reference tables - gdal data types.

**Description**

Reference tables - gdal data types.

**Usage**

GDT_NAMES

**Format**

An object of class character of length 12.

**Source**

gdal values.

---

**kindcd3old**

Reference table - List of RMRS plots that have fallen out of inventory because they were not found or they were in the wrong place.

**Description**

Reference table - List of RMRS plots that have fallen out of inventory because they were not found or they were in the wrong place.

**Usage**

kindcd3old

**Format**

An object of class data.frame with 38 rows and 8 columns.
modGBarea

Source
FIA query. SELECT bp.STATECD, bp.COUNTYCD, bp.PLOT_FIADB NEW_PLOT, bp.START_DATE NEW_START_DATE, bp_old.COUNTYCD OLD_COUNTYCD, bp_old.PLOT_FIADB OLD_PLOT, bp_old.END_DATE OLD_END_DATE, p.CN FROM fs_nims_rmrs.NIMS_BASE_PLOT bp JOIN fs_nims_rmrs.NIMS_BASE_PLOT bp_old on (bp.PREV_NBP_CN=bp_old.CN) JOIN fs_nims_rmrs.NIMS_PLOT_RMRS_VW p on(p.NBP_CN=bp_old.CN) WHERE p.KINDCD = 1 ORDER BY bp.STATECD, bp.COUNTYCD, bp_old.PLOT_FIADB

modGBarea

Green-Book module - Generate area estimates.

Description
Generates area estimates by domain (and estimation unit). Calculations are based on Scott et al. 2005 (‘the green-book’) for mapped forest inventory plots. The non-ratio estimator for estimating area by stratum and domain is used. Plots that are totally nonsampled are excluded from estimation dataset. Next, an adjustment factor is calculated by strata to adjust for nonsampled (nonresponse) conditions that have proportion less than 1. The attribute is the proportion of the plot which is divided by the adjustment factor, and averaged by stratum. Strata means are combined using the strata weights and then expanded to area using the total land area in the population.

Usage
modGBarea(
  GBpopdat,
  landarea = "FOREST",
  pcfilter = NULL,
  rowvar = NULL,
  colvar = NULL,
  sumunits = TRUE,
  returntitle = FALSE,
  savedata = FALSE,
  table_opts = NULL,
  title_opts = NULL,
  savedata_opts = NULL,
  gui = FALSE,
  ...
)

Arguments
GBpopdat List. Population data objects returned from modGBpop().
landarea String. The sample area filter for estimates ("ALL", "FOREST", "TIMBER-LAND"). If landarea=FOREST, filtered to COND_STATUS_CD = 1; If landarea=TIMBERLAND, filtered to SITECLCD in(1:6) and RESERVCD = 0.
pcfilter String. A filter for plot or cond attributes (including pltassgn). Must be R logical syntax.
rowvar  String. Name of row domain variable in cond. If only one domain, rowvar =
domain variable. If more than one domain, include colvar. If no domain, rowvar
= NULL.
colvar  String. Name of column domain variable in cond.
sumunits Logical. If TRUE, estimation units are summed and returned in one table.
returntitle Logical. If TRUE, returns title(s) of the estimation table(s).
savedata Logical. If TRUE, saves table(s) to outfolder.
table_opts List. See help(table_options()) for a list of options.
title_opts List. See help(title_options()) for a list of options.
savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata
= TRUE.
gui Logical. If gui, user is prompted for parameters.
... Parameters for modGBpop() if GBpopdat is NULL.

Details

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.)</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest, non-sampled nonforest).</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td>pltassgn</td>
<td>puniqueid</td>
<td>Unique identifier for each plot, to link to cond (ex. CN).</td>
</tr>
<tr>
<td></td>
<td>STATECD</td>
<td>Identifies state each plot is located in.</td>
</tr>
<tr>
<td></td>
<td>INVYR</td>
<td>Identifies inventory year of each plot.</td>
</tr>
<tr>
<td></td>
<td>PLOT_STATUS_CD</td>
<td>Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed to be sampled.</td>
</tr>
</tbody>
</table>

For available reference tables: sort(unique(FIESTAutils::ref_codes$VARIABLE))

Value

A list with estimates with percent sampling error for rowvar (and colvar). If sumunits=TRUE or
unitvar=NULL and colvar=NULL, one data frame is returned. Otherwise, a list object is returned
with the following information. If savedata=TRUE, all data frames are written to outfolder.
est  Data frame. Area estimates, in area units (e.g., acres), by rowvar, colvar (and estimation unit). If sumunits=TRUE or one estimation unit and colvar=NULL, or allin1=TRUE, estimates and percent sampling error are in one data frame.
Data frame. Percent sampling errors (Confidence level 68 for estimates by rowvar and colvar (and estimation unit)).

List. If returntitle=TRUE a list with table title(s). The list contains one title if est and pse are in the same table and two titles if est and pse are in separate tables. Row and column tables are also included in list.

List. If rawdata=TRUE, a list including the processing data used for estimation including: number of plots and conditions; stratification information; and 1 to 8 tables with calculated values for table cells and totals (See processing data below).

Table. Number of plots by plot status (e.g., sampled forest on plot, sampled nonforest, nonsampled).

DF. Number of conditions by condition status (forest land, nonforest land, non-census water, census water, nonsampled).

DF. Area by estimation unit.

DF. Condition-level area expansion factors.

DF. Final data table used for estimation.

DF. Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>stratum value</td>
</tr>
<tr>
<td>strwtvar</td>
<td>number of pixels by strata and estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or plots) by strata and estimation unit (strata weight)</td>
</tr>
<tr>
<td>CONDPROP_UNADJ_SUM</td>
<td>summed condition proportion by strata and estimation unit</td>
</tr>
<tr>
<td>CONDPROP_ADJFAC</td>
<td>adjusted condition proportion by strata after nonresponse plots removed</td>
</tr>
<tr>
<td>AREA</td>
<td>total area for estimation unit</td>
</tr>
<tr>
<td>CONDPROP_ADJFAC</td>
<td>average area</td>
</tr>
</tbody>
</table>

Data frames. Separate data frames containing calculated variables used in estimation process. The number of processing tables depends on the input parameters. The tables include: total by estimation unit (unit.totest); rowvar totals (unit.rowest), colvar totals, if not NULL (unit.colvar); and a combination of rowvar and colvar, if colvar is not NULL (unit.grpvar). If sumunits=TRUE, the raw data for the summed estimation units are also included (totest, rowest, colest, grpest, respectively). These tables do not included estimate proportions (nhat and nhat.var).

The data frames include the following information:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
</table>
nhat  estimate proportion of land
nhat.var  variance estimate of estimated proportion of land
NBRPLT.gt0  Number of non-zero plots used in estimates
AREA  total area for estimation unit
est  estimated area of land nhat*areavar
est.var  variance estimate of estimate acres of land nhat.var*areavar^2
est.se  standard error of estimated area of land sqrt(est.var)
est.cv  coefficient of variation of estimated area of land est.se/est
pse  percent sampling error of estimate est.cv*100
CI99left  left tail of 99 percent confidence interval for estimated area
CI99right  right tail of 99 percent confidence interval for estimated area
CI95left  left tail of 95 percent confidence interval for estimated area
CI95right  right tail of 95 percent confidence interval for estimated area
CI67left  left tail of 67 percent confidence interval for estimated area
CI67right  right tail of 67 percent confidence interval for estimated area

savedata
if savedata=TRUE...
tables with estimate and percent standard error will be written as *csv files to outfolder. if raw-
data=TRUE, the rawdata will be output to the outfolder in a folder named rawdata (if raw_fmt="csv")
or a database in the outfolder, if (raw_fmt != "csv").
if outfn.pre is not null...
a prefix is added to output files if raw_fmt = 'csv', prefix is added to file names in rawdata folder if
raw_fmt != 'csv', prefix is added to dsn name

Note

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each
estimation unit by strata by summing the unadjusted condition proportions (CONDPROP_UNADJ)
and dividing by the number of plots in the strata/estimation unit.
If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor.
If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in
adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the
All-Condition-Inventory.

STRATA:
Stratification is used to reduce variance in population estimates by partitioning the population into
homogenous classes (strata), such as forest and nonforest. For stratified sampling methods, the
strata sizes (weights) must be either known or estimated. Remotely-sensed data is often used to gen-
erate strata weights with proportion of pixels by strata. If stratification is desired (strata=TRUE),
the required data include: stratum assignment for the center location of each plot, stored in either
pltassgn or cond; and a look-up table with the area or proportion of the total area of each strata
value by estimation unit, making sure the name of the strata (and estimation unit) variable and
values match the plot assignment name(s) and value(s).

sumunits:
An estimation unit is a population, or area of interest, with known area and number of plots. Indi-
nual counties or combined Super-counties are common estimation units for FIA. An estimation
unit may also be a subpopulation of a larger population (e.g., Counties within a State). Subpopulations are mutually exclusive and independent within a population, therefore estimated totals and variances are additive. For example, State-level estimates are generated by summing estimates from all subpopulations within the State (Bechtold and Patterson. 2005. Chapter 2). Each plot must be assigned to only one estimation unit.

If sumunits=TRUE, estimates are generated by estimation unit, summed together, and returned as one estimate. If rawdata=TRUE, estimates by individual estimation unit are also returned.

If sumunits=FALSE, estimates are generated and returned by estimation unit as one data frame. If savedata=TRUE, a separate file is written for each estimation unit.

stratcombine:
If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.

rroll/collut:
There are several objectives for including rroll/collut look-up tables: 1) to include descriptive names that match row/column codes in the input table; 2) to use number codes that match row/column names in the input table for ordering rows; 3) to add rows and/or columns with 0 values for consistency. No duplicate names are allowed.

Include 2 columns in the table:
1-the merging variable with same name as the variable in the input merge table;
2-the ordering or descriptive variable.
If the ordering variable is the rowvar/colvar in the input table and the descriptive variable is in rowlut/collut, set row.orderby/col.orderby equal to rowvar/colvar. If the descriptive variable is the rowvar/colvar in the input table, and the ordering code variable is in rowlut/collut, set row.orderby/col.orderby equal to the variable name of the code variable in rowlut/collut.

Author(s)
Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

References

Examples
GBpopdat <- modGBpop(
popTabs = list(cond = FIESTA::WYcond,
               tree = FIESTA::WYtree,
               seed = FIESTA::WYseed),
popTabIDs = list(cond = "PLT_CN"),
pltassgn = FIESTA::WYpltassgn,
modGBp2veg

Green-Book module - Generate area estimates.

Description

Generates area estimates by domain (and estimation unit). Calculations are based on Scott et al. 2005 ('the green-book') for mapped forest inventory plots. The non-ratio estimator for estimating area by stratum and domain is used. Plots that are totally nonsampled are excluded from estimation dataset. Next, an adjustment factor is calculated by strata to adjust for nonsampled (nonresponse) conditions that have proportion less than 1. The attribute is the proportion of the plot which is divided by the adjustment factor, and averaged by stratum. Strata means are combined using the strata weights and then expanded to area using the total land area in the population.

Usage

modGBp2veg(
  GBpopdat = NULL,
  p2vegtype = "str",
  landarea = "FOREST",
  pcfilter = NULL,
  vfilter = NULL,
  rowvar = NULL,
  colvar = NULL,
  sumunits = TRUE,
  returntitle = FALSE,
modGBp2veg

```
savedata = FALSE,
table_opts = NULL,
title_opts = NULL,
savedata_opts = NULL,
gui = FALSE,
...
```

### Arguments

- **GBpopdat**: List. Population data objects returned from `modGBpop()`.
- **p2vegtype**: String. Type of p2veg estimate (‘spp’, ‘str’).
- **landarea**: String. The sample area filter for estimates (“ALL”, “FOREST”, ”TIMBERLAND”). If landarea=FOREST, filtered to COND_STATUS_CD = 1; If landarea=TIMBERLAND, filtered to SITECLCD in(1:6) and RESERVCD = 0.
- **pcfilter**: String. A filter for plot or cond attributes (including pltassgn). Must be R logical syntax.
- **vfilter**: String. A filter for the P2 vegetation table used for estimate. Must be R logical syntax.
- **rowvar**: String. Name of row domain variable in cond. If only one domain, rowvar = domain variable. If more than one domain, include colvar. If no domain, rowvar = NULL.
- **colvar**: String. Name of column domain variable in cond.
- **sumunits**: Logical. If TRUE, estimation units are summed and returned in one table.
- **returntitle**: Logical. If TRUE, returns title(s) of the estimation table(s).
- **savedata**: Logical. If TRUE, saves table(s) to outfolder.
- **table_opts**: List. See help(table_options()) for a list of options.
- **title_opts**: List. See help(title_options()) for a list of options.
- **savedata_opts**: List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.
- **gui**: Logical. If gui, user is prompted for parameters.

### Details

If variables are NULL, then it will prompt user to input variables.

#### Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.)</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest)</td>
</tr>
</tbody>
</table>
SITECLCD If landarea=TIMBERLAND. Measure of site productivity.
RESERVCD If landarea=TIMBERLAND. Reserved status.
pltassgn puniqueid Unique identifier for each plot, to link to cond (ex. CN).
STATECD Identifies state each plot is located in.
INVYR Identifies inventory year of each plot.
PLOT_STATUS_CD Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed sampled.

For available reference tables: sort(unique(FIESTAutils::ref_codes$VARIABLE))

Value

A list with estimates with percent sampling error for rowvar (and colvar). If sumunits=TRUE or unitvar=NULL and colvar=NULL, one data frame is returned. Otherwise, a list object is returned with the following information. If savedata=TRUE, all data frames are written to outfolder.

- est Data frame. Area estimates, in area units (e.g., acres), by rowvar, colvar (and estimation unit). If sumunits=TRUE or one estimation unit and colvar=NULL, or allin1=TRUE, estimates and percent sampling error are in one data frame.
- pse Data frame. Percent sampling errors (Confidence level 68 unit).
- titlelst List. If returntitle=TRUE a list with table title(s). The list contains one title if est and pse are in the same table and two titles if est and pse are in separate tables. Row and column tables are also included in list.
- raw List. If rawdata=TRUE, a list including the processing data used for estimation including: number of plots and conditions; stratification information; and 1 to 8 tables with calculated values for table cells and totals (See processing data below).

Raw data

- plotsampcnt Table. Number of plots by plot status (e.g., sampled forest on plot, sampled nonforest, nonsampled).
- condsampcnt DF. Number of conditions by condition status (forest land, nonforest land, non-census water, census water, nonsampled).
- unitarea DF. Area by estimation unit.
- expcondtab DF. Condition-level area expansion factors.
- domdat DF. Final data table used for estimation.
- stratdat Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>stratum value</td>
</tr>
<tr>
<td>strwtvar</td>
<td>number of pixels by strata and estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
</tbody>
</table>
strwt proportion of area (or plots) by strata and estimation unit (strata weight)
CONDPROP_UNADJ_SUM summed condition proportion by strata and estimation unit
CONDPROP_ADJFAC adjusted condition proportion by strata after nonresponse plots removed
AREA total area for estimation unit
CONDPROP_ADJFAC average area

processing data

Data frames. Separate data frames containing calculated variables used in estimation process. The number of processing tables depends on the input parameters. The tables include: total by estimation unit (unit.totest); rowvar totals (unit.rowest), colvar totals, if not NULL (unit.colvar); and a combination of rowvar and colvar, if colvar is not NULL (unit.grpvar). If sumunits=TRUE, the raw data for the summed estimation units are also included (totest, rowest, colest, grpest, respectively). These tables do not include estimate proportions (nhat and nhat.var).

The data frames include the following information:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nhat</td>
<td>estimate proportion of land</td>
</tr>
<tr>
<td>nhat.var</td>
<td>variance estimate of estimated proportion of land</td>
</tr>
<tr>
<td>NBRPLT.gt0</td>
<td>Number of non-zero plots used in estimates</td>
</tr>
<tr>
<td>AREA</td>
<td>total area for estimation unit</td>
</tr>
<tr>
<td>est</td>
<td>estimated area of land nhat*areavar</td>
</tr>
<tr>
<td>est.var</td>
<td>variance estimate of estimate acres of land nhat.var*areavar^2</td>
</tr>
<tr>
<td>est.se</td>
<td>standard error of estimated area of land sqrt(est.var)</td>
</tr>
<tr>
<td>est.cv</td>
<td>coefficient of variation of estimated area of land est.se/est</td>
</tr>
<tr>
<td>pse</td>
<td>percent sampling error of estimate est.cv*100</td>
</tr>
<tr>
<td>CI99left</td>
<td>left tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI99right</td>
<td>right tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95left</td>
<td>left tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95right</td>
<td>right tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67left</td>
<td>left tail of 67 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67right</td>
<td>right tail of 67 percent confidence interval for estimated area</td>
</tr>
</tbody>
</table>

savedata

if savedata=TRUE...
tables with estimate and percent standard error will be written as *.csv files to outfolder. if raw-data=TRUE, the rawdata will be output to the outfolder in a folder named rawdata (if raw_fmt="csv") or a database in the outfolder, if (raw_fmt != "csv").

if outfn.pre is not null...
a prefix is added to output files if raw_fmt = `csv`, prefix is added to file names in rawdata folder if raw_fmt != `csv`, prefix is added to dsn name
Note

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata by summing the unadjusted condition proportions (CONDPROP_UNADJ) and dividing by the number of plots in the strata/estimation unit.

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor. If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

STRATA:
Stratification is used to reduce variance in population estimates by partitioning the population into homogenous classes (strata), such as forest and nonforest. For stratified sampling methods, the strata sizes (weights) must be either known or estimated. Remotely-sensed data is often used to generate strata weights with proportion of pixels by strata. If stratification is desired (strata=TRUE), the required data include: stratum assignment for the center location of each plot, stored in either pltassgn or cond; and a look-up table with the area or proportion of the total area of each strata value by estimation unit, making sure the name of the strata (and estimation unit) variable and values match the plot assignment name(s) and value(s).

sumunits:
An estimation unit is a population, or area of interest, with known area and number of plots. Individual counties or combined Super-counties are common estimation units for FIA. An estimation unit may also be a subpopulation of a larger population (e.g., Counties within a State). Subpopulations are mutually exclusive and independent within a population, therefore estimated totals and variances are additive. For example, State-level estimates are generated by summing estimates from all subpopulations within the State (Bechtold and Patterson. 2005. Chapter 2). Each plot must be assigned to only one estimation unit.

If sumunits=TRUE, estimates are generated by estimation unit, summed together, and returned as one estimate. If rawdata=TRUE, estimates by individual estimation unit are also returned.

If sumunits=FALSE, estimates are generated and returned by estimation unit as one data frame. If savedata=TRUE, a separate file is written for each estimation unit.

stratcombine:
If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.

rowlut/collut:
There are several objectives for including rowlut/collut look-up tables: 1) to include descriptive names that match row/column codes in the input table; 2) to use number codes that match row/column names in the input table for ordering rows; 3) to add rows and/or columns with 0 values for consistency. No duplicate names are allowed.

Include 2 columns in the table:
1-the merging variable with same name as the variable in the input merge table;
2-the ordering or descriptive variable.
If the ordering variable is the rowvar/colvar in the input table and the descriptive variable is in rowlut/collut, set row.orderby/col.orderby equal to rowvar/colvar. If the descriptive variable is the rowvar/colvar in the input table, and the ordering code variable is in rowlut/collut, set row.orderby/col.orderby equal to the variable name of the code variable in rowlut/collut.

**Author(s)**

Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

**References**


---

**modGBpop**  
*Green-Book module - Generate population data for GB module.*

**Description**

Generates population data for generating ‘green-book’ estimates (Scott et al. 2005). Plots that are totally nonsampled are excluded from estimation dataset. Next, an adjustment factor is calculated by strata to adjust for nonsampled (nonresponse) conditions that have proportion less than 1. Attributes adjusted to a per-acre value are summed by plot, divided by the adjustment factor, and averaged by stratum. Strata means are combined using the strata weights and then expanded to using the total land area in the population.

**Usage**

```r
modGBpop(
  popType = "VOL",
  popTabs = popTables(),
  popTabIDs = popTableIDs(),
  popFilter = popFilters(),
  pltassgn = NULL,
  pltassgnid = "PLT_CN",
  dsn = NULL,
  pjoinid = "CN",
  areawt = "CONDPROP_UNADJ",
  nonsamp.vfilter.fixed = TRUE,
  adj = "samp",
  unitvar = NULL,
  unitarea = NULL,
  areavar = "ACRES",
  strata = TRUE,
  stratalut = NULL,
)```
strvar = "STRATUMCD",
savedata = FALSE,
saveobj = FALSE,
objnm = "GBpopdat",
unit_opts = NULL,
strata_opts = NULL,
savedata_opts = NULL,
GBdata = NULL,
pltdat = NULL,
stratdat = NULL,
auxdat = NULL,
gui = FALSE,
...
)

Arguments

popType String. Type of evaluation(s) to include in population data. Note: currently only c('CURR', 'VOL', 'LULC', 'DWM') are available. See details below for descriptions of each.

popTabs List of population tables the user would like returned. See help(popTables) for a list of options.

popTabIDs List of unique IDs corresponding to the population tables that the user has requested. See help(popTableIDs) for a list of options.

popFilter List of population filters. See help(popFilters) for a list of options.

pltassgn DF/DT, Optional. R object, sf R object, comma-delimited file(.csv), layer or spatial layer in dsn, or shapefile(.shp). Plot-level assignment of estimation unit and/or strata, with one record for each plot.

pltassgnid String.


pjoinid String. Join variable in plot to match pltassgnid. Does not need to be uniqueid. If using most current XY coordinates for plot assignments, use identifier for plot (e.g., PLOT_ID).

areawt String. Name of variable for summarizing area weights (e.g., CONDPROP_UNADJ).

nonsamp.vfilter.fixed Logical. Temporary parameter for vegetation data fix.

adj String. How to calculate adjustment factors for nonsampled (nonresponse) conditions based on summed proportions for by plot ('samp', 'plot'). 'samp' - adjustments are calculated at strata/estimation unit level; 'plot' - adjustments are calculated at plot-level. Adjustments are only calculated for annual inventory plots (DESIGNCD=1).

unitvar String. Name of the estimation unit variable in unitarea and cond or pltassgn data frame with estimation unit assignment for each plot (e.g., 'ESTN_UNIT'). Optional if only one estimation unit.
**modGBpop**

**unitarea**
Numeric or DF. Total area by estimation unit. If only 1 estimation unit, include number of total acreage for the area of interest or a data frame with area and estimation unit. If more than one estimation unit, provide a data frame of total area by estimation unit, including unitvar and areavar.

**areavar**
String. Name of area variable in unitarea. Default="ACRES".

**strata**
Logical. If TRUE, include information for post-stratification.

**stratalut**
DF/DT. If strata=TRUE, look-up table with pixel counts or area by strata or proportion or area ('strwt') by strata (and estimation unit). If 'strwt' is not included, set getwt=TRUE and getwtvar as the name of variable to calculate weights from (e.g., pixel counts).

**strvar**
String. If strata=TRUE, name of the strata variable in stratalut and cond or pltassgn data frame with stratum assignment for each plot (Default = 'STRATUMCD').

**savedata**
Logical. If TRUE, saves table(s) to outfolder.

**saveobj**
Logical. If TRUE, saves returned list object to outfolder.

**objnm**
String. Name of *.rds object.

**unit_opts**
List. See help(unit_options()) for a list of options.

**strata_opts**
List. See help(strata_options()) for a list of options. Only used when strata = TRUE.

**savedata_opts**
List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

**GBdata**
R List object. Output data list components from FIESTA::anGBdata().

**pltdat**
R List object. Output data list components from FIESTA::spGetPlots().

**stratdat**
R List object. Output data list components from FIESTA::spGetStrata().

**auxdat**
R List object. Output data list components from FIESTA::spGetAuxiliary().

**gui**
Logical. If gui, user is prompted for parameters.

... For extendibility.

**Details**

Population types

**popType Description**

<table>
<thead>
<tr>
<th>popType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>Population data, including nonsampled plots.</td>
</tr>
<tr>
<td>CURR</td>
<td>Population data for area estimates, excluding nonsampled plots.</td>
</tr>
<tr>
<td>VOL</td>
<td>Population data for area/tree estimates, excluding nonsampled plots.</td>
</tr>
<tr>
<td>LULC</td>
<td>Population data for land use/land cover transitional estimates, including only plots with previous measurement.</td>
</tr>
</tbody>
</table>

If variables are NULL, then it will prompt user to input variables.

Necessary variables:
Data Variable Description

**tree**
- **tuniqueid**: Unique identifier for each plot in tree table.
- **CONDID**: Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.
- **TPA_UNADJ**: Number of trees per acre each sample tree represents (e.g., DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on microplot).

**cond**
- **cuniqueid**: Unique identifier for each plot in cond table.
- **CONDID**: Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.
- **CONDPROP_UNADJ**: Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.
- **COND_STATUS_CD**: Status of each forested condition on plot (i.e., accessible forest, nonforest, water, etc.).
- **NF_COND_STATUS_CD**: If ACI=TRUE. Status of each nonforest condition on plot (i.e., accessible nonforest, nonsampled nonforest).
- **SITECLCD**: If landarea=TIMBERLAND. Measure of site productivity.
- **RESERVCD**: If landarea=TIMBERLAND. Reserved status.
- **SUBPROP_UNADJ**: Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.
- **MACRPROP_UNADJ**: If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot. Set MACRPROP_UNADJ=1, if only 1 condition per macroplot.

**pltassgn**
- **pltassgnid**: Unique identifier for each plot in pltassgn.
- **STATECD**: Identifies state each plot is located in.
- **INVYR**: Identifies inventory year of each plot.
- **PLOT_STATUS_CD**: Status of each plot (i.e., sampled, nonsampled). If not included, all plots are assumed as sampled.

For available reference tables: sort(unique(FIESTAutils::ref_codes$VARIABLE))

**Value**

A list with population data for Green-Book estimates.

- **condx**: Data frame. Condition-level data including plot-level assignment of estimation unit and stratum (if strata=TRUE), condition proportion adjustment factor (cadjfac), and adjusted condition proportions (CONDPROP_ADJ).
- **cuniqueid**: String. Unique identifier of plot in condx and pltcondx.
- **condid**: String. Unique identifier of condition in condx and pltcondx.
- **treex**: Data frame. Tree data within population, used for estimation, including trees per acre adjustment factor (tadjfac), and adjusted trees per acre (TPA_ADJ) (if treef is included).
- **tuniqueid**: String. Unique identifier of plot in treex (if treef is included).
- **ACI.filter**: String. If ACI=FALSE, ACI.filter="COND_STATUS_CD == 1".
- **unitarea**: String. Returned table of area by estimation unit.
- **unitvar**: String. Variable name for estimation unit.
- **strlut**: String. Strata-level table with pixel counts by strata (PIPOINTCNT), strata weights (strwt), number of plots by strata (n.strata), total number of plots in estimation unit (n.total), sum of condition proportions (_UNADJ_SUM), area adjustments (*_ADJFAC), total area, and area expansion by strata (EXPNS).
- **strvar**: String. Variable name for strata. If strata=FALSE, strvar="ONESTRAT".
- **expcondtab**: String. If ACI=FALSE, ACI.filter="COND_STATUS_CD == 1".
modGBpop

plotsampcnt  Data frame. Number of plots by PLOT_STATUS_CD.
condsampcnt  Data frame. Number of conditions by COND_STATUS_CD.
states       String. State names in dataset.
invyrs       String. Range of inventory years in dataset.
stratdat     Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>stratum value</td>
</tr>
<tr>
<td>strwtvar</td>
<td>number of pixels by strata and estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or plots) by strata and estimation unit (i.e., strata weight)</td>
</tr>
<tr>
<td>CONDPROP_UNADJ_SUM</td>
<td>summed condition proportion by strata and estimation unit</td>
</tr>
<tr>
<td>CONDPROP_ADJFAC</td>
<td>adjusted condition proportion by strata after nonsampled plots removed</td>
</tr>
<tr>
<td>AREA_USED</td>
<td>total area of estimation unit</td>
</tr>
<tr>
<td>expfac</td>
<td>strata-level expansion factor after nonsampled plots and conditions removed (AREA_USED/n.strata)</td>
</tr>
<tr>
<td>EXPNS</td>
<td>strata-level area expansions (expfac * strwt)</td>
</tr>
</tbody>
</table>

Table(s) are also written to outfolder.

Note

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata by summing the unadjusted proportions of the subplot, microplot, and macroplot (i.e. *PROP_UNADJ) and dividing by the number of plots in the strata/estimation unit.

An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor.
If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

unitcombine:
If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10.
If there are no estimation units following in order, it is combined with the estimation unit previous in order.

\textit{stratcombine:}

If TRUE and less than 2 plots in any one strata class within an estimation unit, all strata classes with 2 or less plots are combined. The current method for combining is to group the strata with less than 2 plots with the strata class following in consecutive order (numeric or alphabetical), restrained by estimation unit (if \texttt{unitcombine=FALSE}), and continuing until the number of plots equals 10. If there are no strata classes following in order, it is combined with the estimation unit previous in order.

\section*{Author(s)}

Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

\section*{References}


\section*{Examples}

\begin{verbatim}
GBpopdat <- modGBpop(
popTabs = list(cond = FIESTA::WYcond,
                tree = FIESTA::WYtree,
                seed = FIESTA::WYseed),
popTabIDs = list(cond = "PLT_CN"),
pltassgn = FIESTA::WYpltassgn,
pltassgnid = "CN",
pjoinid = "PLT_CN",
unitarea = FIESTA::WYunitarea,
unitvar = "ESTN_UNIT",
strata = TRUE,
stratalut = WYstratalut,
strata_opts = strata_options(getwt = TRUE)
)
str(GBpopdat, max.level = 1)
\end{verbatim}

\section*{modGBratio}

\emph{Green-Book module - Generate ratio estimates.}

\section*{Description}

Generates per-acre and per-tree estimates by domain and/or tree domain (and estimation unit). Calculations are based on chapter 4 of Scott et al. 2005 ("the green-book") for mapped forest inventory plots. The ratio estimator for estimating per-acre or per-tree by stratum and domain is used, referred to as Ratio of Means (ROM).
**Usage**

```r
modGBratio(
    GBpopdat,
    estseed = "none",
    ratiotype = "PERACRE",
    landarea = "FOREST",
    pcfilter = NULL,
    estvarn = NULL,
    estvarn.filter = NULL,
    estvard = NULL,
    estvard.filter = NULL,
    rowvar = NULL,
    colvar = NULL,
    sumunits = TRUE,
    returntitle = FALSE,
    savedata = FALSE,
    table_opts = NULL,
    title_opts = NULL,
    savedata_opts = NULL,
    gui = FALSE,
    ...
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GBpopdat</td>
<td>List. Population data objects returned from <code>modGBpop()</code>.</td>
</tr>
<tr>
<td>estseed</td>
<td>String. Use seedling data only or add to tree data. Seedling estimates are only for counts (estvar='TPA_UNADJ')-('none', 'only', 'add').</td>
</tr>
<tr>
<td>ratiotype</td>
<td>String. The type of ratio estimates (&quot;PERACRE&quot;, &quot;PERTREE&quot;).</td>
</tr>
<tr>
<td>landarea</td>
<td>String. The sample area filter for estimates (&quot;FOREST&quot;, &quot;TIMBERLAND&quot;). If landarea=FOREST, filtered to COND_STATUS_CD = 1; If landarea=TIMBERLAND, filtered to SITECLCD in(1:6) and RESERVCD = 0.</td>
</tr>
<tr>
<td>pcfilter</td>
<td>String. A filter for plot or cond attributes (including pltassgn). Must be R logical syntax.</td>
</tr>
<tr>
<td>estvarn</td>
<td>String. Name of the tree estimate variable (numerator).</td>
</tr>
<tr>
<td>estvarn.filter</td>
<td>String. A tree filter for the estimate variable (numerator). Must be R syntax (e.g., &quot;STATUSCD == 1&quot;).</td>
</tr>
<tr>
<td>estvard</td>
<td>String. Name of the tree estimate variable (denominator).</td>
</tr>
<tr>
<td>estvard.filter</td>
<td>String. A tree filter for the estimate variable (denominator). Must be R syntax (e.g., &quot;STATUSCD == 1&quot;).</td>
</tr>
<tr>
<td>rowvar</td>
<td>String. Name of the row domain variable in cond or tree. If only one domain, rowvar = domain variable. If more than one domain, include colvar. If no domain, rowvar = NULL.</td>
</tr>
<tr>
<td>colvar</td>
<td>String. Name of the column domain variable in cond or tree.</td>
</tr>
<tr>
<td>sumunits</td>
<td>Logical. If TRUE, estimation units are summed and returned in one table.</td>
</tr>
</tbody>
</table>
returntitle Logical. If TRUE, returns title(s) of the estimation table(s).
savedata Logical. If TRUE, saves table(s) to outfolder.
table_opts List. See help(table_options()) for a list of options.
title_opts List. See help(title_options()) for a list of options.
savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.
gui Logical. If gui, user is prompted for parameters.

... Parameters for modGBpop() if GBpopdat is NULL.

Details

If variable = NULL, then it will prompt user for input.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot.</td>
<td></td>
</tr>
<tr>
<td>TPA_UNADJ</td>
<td>Number of trees per acre each sample tree represents (ex. DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on microplot).</td>
<td></td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
<td></td>
</tr>
<tr>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
<td></td>
</tr>
<tr>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.).</td>
<td></td>
</tr>
<tr>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest, nonsampled nonforest).</td>
<td></td>
</tr>
<tr>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
<td></td>
</tr>
<tr>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
<td></td>
</tr>
<tr>
<td>SUBPROP_UNADJ</td>
<td>Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.</td>
<td></td>
</tr>
<tr>
<td>MICRPROP_UNADJ</td>
<td>If microplot tree attributes. Unadjusted proportion of microplot conditions on each plot.</td>
<td></td>
</tr>
<tr>
<td>MACRPROP_UNADJ</td>
<td>If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot.</td>
<td></td>
</tr>
<tr>
<td>pltassgn</td>
<td>puniqueid</td>
<td>Unique identifier for each plot, to link to cond (ex. CN).</td>
</tr>
<tr>
<td>STATECD</td>
<td>Identifies state each plot is located in.</td>
<td></td>
</tr>
<tr>
<td>INVYR</td>
<td>Identifies inventory year of each plot.</td>
<td></td>
</tr>
<tr>
<td>PLOT_STATUS_CD</td>
<td>Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed sampled.</td>
<td></td>
</tr>
</tbody>
</table>

For available reference tables: sort(unique(FIESTAutils::ref_codes$VARIABLE))

Value

A list with estimates with percent sampling error for rowvar (and colvar). If sumunits=TRUE or unitvar=NULL and colvar=NULL, one data frame is returned. Otherwise, a list object is returned with the following information. If savedata=TRUE, all data frames are written to outfolder.

est Data frame. Tree estimates by rowvar, colvar (and estimation unit). If sumunits=TRUE or one estimation unit and colvar=NULL, estimates and percent sampling error are in one data frame.
Data frame. Percent sampling errors (Confidence level 68 colvar (and estimation unit). Note: for 95 percent sampling error by 1.96.

List with 1 or 2 string vectors. If returntitle=TRUE a list with table title(s). The list contains one title if est and pse are in the same table and two titles if est and pse are in separate tables.

List of data frames. If rawdata=TRUE, a list including the processing data used for estimation including: number of plots and conditions; stratification information; and 1 to 8 tables with calculated values for table cells and totals (See processing data below).

Table. Number of plots by plot status (ex. sampled forest on plot, sampled nonforest, nonsampled).

DF. Number of conditions by condition status (forest land, nonforest land, non-census water, census water, nonsampled).

DF. Area by estimation unit.

DF. Condition-level area expansion factors.

DF. Final data table used for estimation.

Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>stratum value</td>
</tr>
<tr>
<td>strwtvar</td>
<td>number of pixels by strata and estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or plots) by strata and estimation unit (i.e., strata weight)</td>
</tr>
<tr>
<td>CONDPROP_UNADJ_SUM</td>
<td>summed condition proportion by strata and estimation unit</td>
</tr>
<tr>
<td>CONDPROP_ADJFAC</td>
<td>adjusted condition proportion by strata after nonsampled plots removed</td>
</tr>
</tbody>
</table>

Data frames. Separate data frames of variables used in estimation process for the rowvar, colvar and combination of rowvar and colvar (if colvar is not NULL), and grand total by estimation unit (unit.rowest, unit.colest, unit.grpest, unit.totest, respectively) and summed estimation units, if sumunits=TRUE (rowset, colest, grpest, totest, respectively).

The data frames include the following information:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nhat</td>
<td>estimated proportion of trees for numerator</td>
</tr>
<tr>
<td>nhat.var</td>
<td>variance estimate of estimated proportion of trees for numerator</td>
</tr>
<tr>
<td>dhat</td>
<td>estimated proportion of trees for denominator</td>
</tr>
<tr>
<td>dhat.var</td>
<td>variance estimate of estimated proportion of trees for denominator</td>
</tr>
<tr>
<td>covar</td>
<td>covariance for ratio</td>
</tr>
</tbody>
</table>
NBRPLT.gt0  Number of non-zero plots used in estimates
ACRES   total area for estimation unit
estn   estimated area of trees, for numerator nhat*ACRES
estn.var   variance estimate of estimated area of trees nhat.var*areavar^2
estd   estimated area of land (ratio=’PERACRE’), for denominator dhat*areavar
estd.var   variance of estimated area, for denominator dhat.var*areavar^2
estd.covar   estimated covariance of numerator and denominator covar*areavar^2
rhat   estimated ratio estn/estd
rhat.var   variance estimate of estimation ratio estn.var+rhat^2*estd.var-2*rhat*est.covar)/estd^2
rhat.se   estimated standard error of ratio sqrt(rhat.var)
rhat.cv   estimated percent standard error or ratio rhat.se/rhat
rhat.pse   estimated percent standard error of ratio rhat.cv*100
CI99left   left tail of 99 percent confidence interval for estimated area
CI99right  right tail of 99 percent confidence interval for estimated area
CI95left   left tail of 95 percent confidence interval for estimated area
CI95right  right tail of 95 percent confidence interval for estimated area
CI67left   left tail of 67 percent confidence interval for estimated area
CI67right  right tail of 67 percent confidence interval for estimated area

Table(s) are also written to outfolder.

Note

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata by summing the unadjusted proportions of the subplot, microplot, and macroplot (i.e. *PROP_UNADJ) and dividing by the number of plots in the strata/estimation unit.

An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=False, only nonsampled forest conditions are accounted for in the adjustment factor.
If ACI=True, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

STRATA:
Stratification is used to reduce variance in population estimates by partitioning the population into homogenous classes (strata), such as forest and nonforest. For stratified sampling methods, the strata sizes (weights) must be either known or estimated. Remotely-sensed data is often used to generate strata weights with proportion of pixels by strata. If stratification is desired (strata=True), the required data include: stratum assignment for the center location of each plot, stored in either pltassgn or cond; and a look-up table with the area or proportion of the total area of each strata.
value by estimation unit, making sure the name of the strata (and estimation unit) variable and
values match the plot assignment name(s) and value(s).

sumunits:
An estimation unit is a population, or area of interest, with known area and number of plots. Individual counties or combined Super-counties are common estimation units for FIA. An estimation unit may also be a subpopulation of a larger population (e.g., Counties within a State). Subpopulations are mutually exclusive and independent within a population, therefore estimated totals and variances are additive. For example, State-level estimates are generated by summing estimates from all subpopulations within the State (Bechtold and Patterson. 2005. Chapter 2). Each plot must be assigned to only one estimation unit.

If sumunits=TRUE, estimates are generated by estimation unit, summed together, and returned as one estimate. If rawdata=TRUE, estimates by individual estimation unit are also returned.

If sumunits=FALSE, estimates are generated and returned by estimation unit as one data frame. If savedata=TRUE, a separate file is written for each estimation unit.

stratcombine:
If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.

rowlut/collut:
There are several objectives for including rowlut/collut look-up tables: 1) to include descriptive names that match row/column codes in the input table; 2) to use number codes that match row/column names in the input table for ordering rows; 3) to add rows and/or columns with 0 values for consistency. No duplicate names are allowed.

Include 2 columns in the table:
1-the merging variable with same name as the variable in the input merge table;
2-the ordering or descriptive variable.

If the ordering variable is the rowvar/colvar in the input table and the descriptive variable is in rowlut/collut, set row.orderby/col.orderby equal to rowvar/colvar. If the descriptive variable is the rowvar/colvar in the input table, and the ordering code variable is in rowlut/collut, set row.orderby/col.orderby equal to the variable name of the code variable in rowlut/collut.

UNITS:
The following variables are converted from pounds (in NIMS) to short tons by multiplying the variable by 0.0005. DRYBIO_AG, DRYBIO_BG, DRYBIO_WDLD_SPP, DRYBIO_SAPLING, DRYBIO_STUMP, DRYBIO_TOP, DRYBIO_BOLE, DRYBIOT, DRYBIOM, DRYBIOTB, JBIOTOT, CARBON_BG, CARBON_AG

MORTALITY:
For Interior-West FIA, mortality estimates are mainly based on whether a tree has died within the last 5 years of when the plot was measured. If a plot was remeasured, mortality includes trees that were alive the previous visit but were dead in the next visit. If a tree was standing the previous visit, but was not standing in the next visit, no diameter was collected (DIA = NA) but the tree is defined as mortality.

Common tree filters:
### FILTER

<table>
<thead>
<tr>
<th>Filter Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;STATUSCD == 1&quot;</td>
<td>Live trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2&quot;</td>
<td>Dead trees</td>
</tr>
<tr>
<td>&quot;TPAMORT_UNADJ &gt; 0&quot;</td>
<td>Mortality trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; DIA &gt;= 5.0&quot;</td>
<td>Dead trees &gt;= 5.0 inches diameter</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; AGENTCD == 30&quot;</td>
<td>Dead trees from fire</td>
</tr>
</tbody>
</table>

### Author(s)

Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

### References


### Examples

```r
GBpopdat <- modGBpop(
popTabs = list(cond = FIESTA::WYcond,
                tree = FIESTA::WYtree,
                seed = FIESTA::WYseed),
popTabIDs = list(cond = "PLT_CN"),
pltassgn = FIESTA::WYpltassgn,
pltassgnid = "CN",
pjoinid = "PLT_CN",
unitarea = FIESTA::WYunitarea,
unitvar = "ESTN_UNIT",
strata = TRUE,
stratalut = WYstratalut,
strata_opts = strata_options(getwt = TRUE))
```

```r
ratio1.1 <- modGBratio(
  GBpopdat = GBpopdat, # pop - population calculations
  landarea = "TIMBERLAND", # est - forest land filter
  sumunits = TRUE, # est - sum estimation units to population
  estvarn = "VOLCFNET", # est - net cubic-foot volume, numerator
  estvarn.filter = "STATUSCD == 1", # est - live trees only, numerator
  returntitle = TRUE, # out - return title information
)
```

```r
ratio1.2 <- modGBratio(
  GBpopdat = GBpopdat, # pop - population calculations
  landarea = "TIMBERLAND", # est - forest land filter
  sumunits = TRUE, # est - sum estimation units to population
```
modGBtree

estvar = "VOLCFNET", # est - net cubic-foot volume
estvar.filter = "STATUSCD == 1", # est - live trees only
rowvar = "FORTYPCD", # est - row domain
returntitle = TRUE # out - return title information
}
str(ratio1.2, max.level = 1)

modGBtree

Green-Book module - Generate tree estimates.

Description

Generates tree and or seedling estimates by domain and/or tree domain (and estimation unit). Calculations are based on Scott et al. 2005 ("the green-book") for mapped forest inventory plots. The non-ratio estimator for estimating tree attributes by stratum and domain is used. Plots that are totally nonsampled are excluded from estimation dataset. Next, an adjustment factor is calculated by strata to adjust for nonsampled (nonresponse) conditions that have proportion less than 1. Attributes adjusted to a per-acre value are summed by plot, divided by the adjustment factor, and averaged by stratum. Strata means are combined using the strata weights and then expanded to using the total land area in the population.

Usage

modGBtree(
  GBpopdat,
  estvar,
  estvar.filter = NULL,
  estseed = "none",
  landarea = "FOREST",
  pcfilter = NULL,
  rowvar = NULL,
  colvar = NULL,
  sumunits = TRUE,
  returntitle = FALSE,
  savedata = FALSE,
  table_opts = NULL,
  title_opts = NULL,
  savedata_opts = NULL,
  gui = FALSE,
  ...
)

Arguments

GBpopdat List. Population data objects returned from FIESTA::modGBpop().
estvar String. Name of the tree-level estimate variable (e.g., 'VOLCFNET').
estvar.filter String. A tree-level filter for estvar. Must be R syntax (e.g., 'STATUSCD == 1').
estseed: String. Use seedling data only or add to tree data. Seedling estimates are only for counts (estvar = 'TPA_UNADJ') - ('none', 'only', 'add').

landarea: String. The condition-level filter for defining land area ('ALL', 'FOREST', 'TIMBERLAND'). If landarea = 'FOREST', COND_STATUS_CD = 1; if landarea = 'TIMBERLAND', SITECLCD in(1:6) & RESERVCD = 0.

pcfilter: String. A filter for plot or cond attributes (including pltassgn). Must be R logical syntax.

rowvar: String. Optional. Name of domain variable to group estvar by for rows in table output. Rowvar must be included in an input data frame (i.e., plt, cond, tree). If no rowvar is included, an estimate is returned for the total estimation unit. Include colvar for grouping by 2 variables.

colvar: String. Optional. If rowvar != NULL, name of domain variable to group estvar by for columns in table output. Colvar must be included in an input data frame (i.e., plt, cond, tree).

sumunits: Logical. If TRUE, estimation units are summed and returned in one table.

returntitle: Logical. If TRUE, returns title(s) of the estimation table(s).

savedata: Logical. If TRUE, saves table(s) to outfolder.

table_opts: List. See help(table_options()) for a list of options.

title_opts: List. See help(title_options()) for a list of options.

savedata_opts: List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

gui: Logical. If gui, user is prompted for parameters.

... Parameters for modGBpop() if GBpopdat is NULL.

**Details**

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>TPA_UNADJ</td>
<td>Number of trees per acre each sample tree represents (ex. DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on microplot).</td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.).</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest).</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td></td>
<td>SUBPROP_UNADJ</td>
<td>Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.</td>
</tr>
<tr>
<td></td>
<td>MICRPROP_UNADJ</td>
<td>If microplot tree attributes. Unadjusted proportion of microplot conditions on each plot.</td>
</tr>
<tr>
<td></td>
<td>MACRPROP_UNADJ</td>
<td>If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot.</td>
</tr>
<tr>
<td>pltassgn</td>
<td>puniqueid</td>
<td>Unique identifier for each plot, to link to cond (ex. CN).</td>
</tr>
</tbody>
</table>
STATECD Identifies state each plot is located in.
INVYR Identifies inventory year of each plot.
PLOT_STATUS_CD Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed to be sampled.

For available reference tables: sort(unique(FIESTAutils::ref_codes$VARIABLE))

**Value**

A list with estimates with percent sampling error for rowvar (and colvar). If sumunits=TRUE or unitvar=NULL and colvar=NULL, one data frame is returned. Otherwise, a list object is returned with the following information. If savedata=TRUE, all data frames are written to outfolder.

- **est**: Data frame. Tree estimates by rowvar, colvar (and estimation unit). If sumunits=TRUE or one estimation unit and colvar=NULL, estimates and percent sampling error are in one data frame.
- **pse**: Data frame. Percent sampling errors (Confidence level 68 colvar (and estimation unit). Note: for 95 percent sampling error by 1.96.
- **titlelst**: List with 1 or 2 string vectors. If returntitle=TRUE a list with table title(s). The list contains one title if est and pse are in the same table and two titles if est and pse are in separate tables.
- **raw**: List of data frames. If rawdata=TRUE, a list including the processing data used for estimation including: number of plots and conditions; stratification information; and 1 to 8 tables with calculated values for table cells and totals (See processing data below).

**Raw data**

- **plotsampcnt**: Table. Number of plots by plot status (ex. sampled forest on plot, sampled nonforest, nonsampled).
- **condsampcnt**: DF. Number of conditions by condition status (forest land, nonforest land, non-census water, census water, nonsampled).
- **unitarea**: DF. Area by estimation unit.
- **expcondtab**: DF. Condition-level area expansion factors.
- **tdomdat**: DF. Final data table used for estimation.
- **stratdat**: Data frame. Strata information by estimation unit.

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>stratum value</td>
</tr>
<tr>
<td>strwtvar</td>
<td>number of pixels by strata and estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or plots) by strata and estimation unit (i.e., strata weight)</td>
</tr>
<tr>
<td>CONDPROP_UNADJ_SUM</td>
<td>summed condition proportion by strata and estimation unit</td>
</tr>
<tr>
<td>CONDPROP_ADJFAC</td>
<td>adjusted condition proportion by strata after nonsampled plots removed</td>
</tr>
</tbody>
</table>
processing data

Data frames. Separate data frames containing calculated variables used in estimation process. The number of processing tables depends on the input parameters. The tables include: total by estimation unit (unit.totest); rowvar totals (unit.rowest), and if colvar is not NULL, colvar totals, (unit.colvar); and a combination of rowvar and colvar (unit.grpvar). If sumunits=TRUE, the raw data for the summed estimation units are also included (totest, rowest, colest, grpest, respectively). These tables do not included estimate proportions (nhat and nhat.var).

The data frames include the following information:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nhat</td>
<td>estimated proportion of trees</td>
</tr>
<tr>
<td>nhat.var</td>
<td>variance estimate of estimated proportion of trees</td>
</tr>
<tr>
<td>NBRPLT.gt0</td>
<td>Number of non-zero plots used in estimates</td>
</tr>
<tr>
<td>ACRES</td>
<td>total area for estimation unit</td>
</tr>
<tr>
<td>est</td>
<td>estimated area of trees nhat*ACRES</td>
</tr>
<tr>
<td>est.var</td>
<td>variance estimate of estimated area of trees nhat.var*areavar^2</td>
</tr>
<tr>
<td>est.se</td>
<td>standard error of estimated area of trees sqrt(est.var)</td>
</tr>
<tr>
<td>est.cv</td>
<td>coefficient of variation of estimated area of trees est.se/est</td>
</tr>
<tr>
<td>pse</td>
<td>percent sampling error of estimate est.cv*100</td>
</tr>
<tr>
<td>CI99left</td>
<td>left tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI99right</td>
<td>right tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95left</td>
<td>left tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95right</td>
<td>right tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67left</td>
<td>left tail of 67 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67right</td>
<td>right tail of 67 percent confidence interval for estimated area</td>
</tr>
</tbody>
</table>

Table(s) are also written to outfolder.

Note

ADJUSTMENT FACTOR:

The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata. by summing the unadjusted proportions of the subplot, microplot, and macroplot (i.e. *PROP_UNADJ) and dividing by the number of plots in the strata/estimation unit).

An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor.
If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in
adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

sumunits:
An estimation unit is a population, or area of interest, with known area and number of plots. Individual counties or combined Super-counties are common estimation units for FIA. An estimation unit may also be a subpopulation of a larger population (e.g., Counties within a State). Subpopulations are mutually exclusive and independent within a population, therefore estimated totals and variances are additive. For example, State-level estimates are generated by summing estimates from all subpopulations within the State (Bechtold and Patterson. 2005. Chapter 2). Each plot must be assigned to only one estimation unit.

If sumunits=TRUE, estimates are generated by estimation unit, summed together, and returned as one estimate. If rawdata=TRUE, estimates by individual estimation unit are also returned.

If sumunits=FALSE, estimates are generated and returned by estimation unit as one data frame. If savedata=TRUE, a separate file is written for each estimation unit.

stratcombine:
If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.

rowlut/collut:
There are several objectives for including rowlut/collut look-up tables: 1) to include descriptive names that match row/column codes in the input table; 2) to use number codes that match row/column names in the input table for ordering rows; 3) to add rows and/or columns with 0 values for consistency. No duplicate names are allowed.

Include 2 columns in the table:
1-the merging variable with same name as the variable in the input merge table;
2-the ordering or descriptive variable.

If the ordering variable is the rowvar/colvar in the input table and the descriptive variable is in rowlut/collut, set row.orderby/col.orderby equal to rowvar/colvar. If the descriptive variable is the rowvar/colvar in the input table, and the ordering code variable is in rowlut/collut, set row.orderby/col.orderby equal to the variable name of the code variable in rowlut/collut.

UNITS:
The following variables are converted from pounds (from FIA database) to short tons by multiplying the variable by 0.0005. DRYBIO_AG, DRYBIO_BG, DRYBIO_WDLD_SPP, DRYBIO_SAPLING, DRYBIO_STUMP, DRYBIO_TOP, DRYBIO_BOLE, DRYBIOT, DRYBIOM, DRYBIOTB, JBIOTOT, CARBON_BG, CARBON_AG

MORTALITY:
For Interior-West FIA, mortality estimates are mainly based on whether a tree has died within the last 5 years of when the plot was measured. If a plot was remeasured, mortality includes trees that were alive the previous visit but were dead in the next visit. If a tree was standing the previous visit, but was not standing in the next visit, no diameter was collected (DIA = NA) but the tree is defined as mortality.

Common tree filters:
FILTER DESCRIPTION
"STATUSCD == 1" Live trees
"STATUSCD == 2" Dead trees
"TPAMORT_UNADJ > 0" Mortality trees
"STATUSCD == 2 & DIA >= 5.0" Dead trees >= 5.0 inches diameter
"STATUSCD == 2 & AGENTCD == 30" Dead trees from fire

Author(s)
Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

References

Examples
```r
GBpopdat <- modGBpop(
  popTabs = list(cond = FIESTA::WYcond,
                  tree = FIESTA::WYtree,
                  seed = FIESTA::WYseed),
  popTabIDs = list(cond = "PLT_CN"),
  pltassgn = FIESTA::WYpltassgn,
  pltassgnid = "CN",
  pjoinid = "PLT_CN",
  unitarea = FIESTA::WYunitarea,
  unitvar = "ESTN_UNIT",
  strata = TRUE,
  stratalut = WYstratalut,
  strata_opts = strata_options(getwt = TRUE)
)

tree1.1 <- modGBtree(
  GBpopdat = GBpopdat, # pop - population calculations
  landarea = "FOREST", # est - forest land filter
  sumunits = TRUE, # est - sum estimation units to population
  estvar = "VOLCFNET", # est - net cubic-foot volume
  estvar.filter = "STATUSCD == 1", # est - live trees only
  returntitle = TRUE # out - return title information
)
str(tree1.1, max.level = 1)

tree1.2 <- modGBtree(
  GBpopdat = GBpopdat, # pop - population calculations
  landarea = "FOREST", # est - forest land filter
  sumunits = TRUE, # est - sum estimation units to population
  estvar = "VOLCFNET", # est - net cubic-foot volume
)```

modMAarea

Model-Assisted module - Generate model-assisted area estimates.

Description

Generates area estimates by estimation unit. Estimates are calculated from McConville et al. (2018)'s mase R package.

Usage

modMAarea(
  MApopdat,
  MAmethod,
  FIA = TRUE,
  prednames = NULL,
  modelselect = FALSE,
  landarea = "FOREST",
  pcfilter = NULL,
  rowvar = NULL,
  colvar = NULL,
  bootstrap = FALSE,
  returntitle = FALSE,
  savedata = FALSE,
  table_opts = NULL,
  title_opts = NULL,
  savedata_opts = NULL,
  gui = FALSE,
  ...
)

Arguments

MApopdat  List. Population data objects returned from modMApop().
MAmethod  String. mase (i.e., model-assisted) method to use (‘greg’, ‘gregEN’, ‘ratio’).
FIA  Logical. If TRUE, the finite population term is removed from estimator to match FIA estimates.
prednames  String vector. Name(s) of predictor variables to include in model.
modelselect  Boolean. TRUE if you want model selection to occur.
landarea  String. The sample area filter for estimates (‘ALL’, ‘FOREST’, ‘TIMBER-LAND’). If landarea=FOREST, filtered to COND_STATUS_CD = 1; If landarea=TIMBERLAND, filtered to SITECLCD in(1:6) and RESERVCD = 0.
pcfilter  String. A filter for plot or cond attributes (including pltassign). Must be R logical syntax.

rowvar   String. Name of the row domain variable in cond or tree. If only one domain, rowvar = domain variable. If more than one domain, include colvar. If no domain, rowvar = NULL.

colvar   String. Name of the column domain variable in cond or tree.

bootstrap Logical. If TRUE, returns bootstrap variance estimates, otherwise uses Horvitz-Thompson estimator under simple random sampling without replacement.

returntitle Logical. If TRUE, returns title(s) of the estimation table(s).

savedata Logical. If TRUE, saves table(s) to outfolder.

table_opts List. See help(table_options()) for a list of options.

title_opts List. See help(title_options()) for a list of options.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

gui Logical. If gui, user is prompted for parameters.

... Parameters for modMApop() if MApopdat is NULL.

Details

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltstrat (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot (ex. PLT_CN).</td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltstrat (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDSTATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.).</td>
</tr>
<tr>
<td></td>
<td>NF_CONDSTATUS_CD</td>
<td>Status of each nonforest condition on plot (i.e. accessible nonforest, nonsampled nonforest).</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td>pltstrat</td>
<td>puniqueid</td>
<td>Unique identifier for each plot, to link to cond (ex. CN).</td>
</tr>
<tr>
<td></td>
<td>STATECD</td>
<td>Identifies state each plot is located in.</td>
</tr>
<tr>
<td></td>
<td>INVYR</td>
<td>Identifies inventory year of each plot.</td>
</tr>
<tr>
<td></td>
<td>PLOTSTATUS_CD</td>
<td>Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed sampled.</td>
</tr>
</tbody>
</table>

Reference names are available for the following variables:

ADFORD, AGENTCD, CCLCD, DECAYCD, DSTRBCD, KINDCD, OWNCD, OWNGRPDCD, FORTYPDCD, FLDTYPDCD, FORTYPDCDCALC, TYPGRCDCD, FORINDC, RESERVCD, LANDCLCD, STDTSZCD, FLDSZCD, PHYSCLCD, MISTCLCD, PLOTSTATUS_CD, STATECD, TREECLCD, TRTCD, SPCD, SPGRPCD
**Value**

If FIA=TRUE or unitvar=NULL and colvar=NULL, one data frame is returned with tree estimates and percent sample errors. Otherwise, a list is returned with tree estimates in one data frame (est) and percent sample errors in another data frame (est.pse). If rawdata=TRUE, another list is returned including raw data used in the estimation process. If addtitle=TRUE and returntitle=TRUE, the title for est/pse is returned. If savedata=TRUE, all data frames are written to outfolder.

- **est** Data frame. Tree estimates by rowvar, colvar (and estimation unit). If FIA=TRUE or one estimation unit and colvar=NULL, estimates and percent sampling error are in one data frame.

- **pse** Data frame. Percent sampling errors for estimates by rowvar and colvar (and estimation unit).

- **titlelst** List with 1 or 2 string vectors. If returntitle=TRUE a list with table title(s). The list contains one title if est and pse are in the same table and two titles if est and pse are in separate tables.

- **raw** List of data frames. If rawdata=TRUE, a list including: number of plots by plot status, if in dataset (plotsampcnt); number of conditions by condition status (condsampcnt); data used for post-stratification (stratdat); and 1-8 tables with calculated variables used for processing estimates and percent sampling error for table cell values and totals (See processing data below).

**Raw data**

- **plotsampcnt** Table. Number of plots by plot status (ex. sampled forest on plot, sampled nonforest, nonsampled).

- **condsampcnt** DF. Number of conditions by condition status (forest land, nonforest land, non-census water, census water, nonsampled).

- **stratdat** Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESTUNIT</td>
<td>estimation unit</td>
</tr>
<tr>
<td>STRATA</td>
<td>strata</td>
</tr>
<tr>
<td>ACRES</td>
<td>area by strata for estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (and estimation unit)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>TOTACRES</td>
<td>total area for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or number of plots) by strata (strata weight)</td>
</tr>
<tr>
<td>expfac.strata</td>
<td>expansion factor (in area unit (e.g., acres) by strata (areavar/n.strata)</td>
</tr>
</tbody>
</table>

**processing data**

Data frames. Separate data frames containing calculated variables used in estimation process. The number of processing tables depends on the input parameters. The tables include: total by estimation unit (unit.totest); rowvar totals (unit.rowest), and if colvar is not NULL, colvar totals, (unit.colvar); and a combination of rowvar and colvar (unit.grpvar). If FIA=TRUE, the raw data for the summed estimation units are also included (totest, rowest, colest, grpest, respec-
These tables do not include estimate proportions (nhat and nhat.var). The data frames include the following information:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nhat</td>
<td>estimated proportion of trees</td>
</tr>
<tr>
<td>nhat.var</td>
<td>estimated variance of estimated proportion of trees</td>
</tr>
<tr>
<td>ACRES</td>
<td>total area for estimation unit</td>
</tr>
<tr>
<td>est</td>
<td>estimated area of trees nhat*ACRES</td>
</tr>
<tr>
<td>est.var</td>
<td>estimated variance of estimated area of trees nhat.var*areavar^2</td>
</tr>
<tr>
<td>est.se</td>
<td>standard error of estimated area of trees sqrt(est.var)</td>
</tr>
<tr>
<td>est.cv</td>
<td>coefficient of variation of estimated area of trees est.se/est</td>
</tr>
<tr>
<td>pse</td>
<td>percent sampling error of estimate est.cv*100</td>
</tr>
<tr>
<td>CI99left</td>
<td>left tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI99right</td>
<td>right tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95left</td>
<td>left tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95right</td>
<td>right tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67left</td>
<td>left tail of 67 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67right</td>
<td>right tail of 67 percent confidence interval for estimated area</td>
</tr>
</tbody>
</table>

Table(s) are also written to outfolder.

**Note**

**ADJUSTMENT FACTOR:**

The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata. by summing the unadjusted proportions of the subplot, microplot, and macroplot (i.e. PROP_UNADJ) and dividing by the number of plots in the strata/estimation unit).

An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor. If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

**stratcombine:**

If MAmethod='PS', and stratcombine=TRUE, and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.
autoxreduce:
If MAMethod='GREG', and autoxreduce=TRUE, and there is an error because of multicolinearity, a variable reduction method is applied to remove correlated variables. The method used is based on the variance-inflation factor (vif) from a linear model. The vif estimates how much the variance of each x variable is inflated due to multicolinearity in the model.

rowlut/collut:
There are several objectives for including rowlut/collut look-up tables: 1) to include descriptive names that match row/column codes in the input table; 2) to use number codes that match row/column names in the input table for ordering rows; 3) to add rows and/or columns with 0 values for consistency. No duplicate names are allowed.

Include 2 columns in the table:
1-the merging variable with same name as the variable in the input merge table;
2-the ordering or descriptive variable.
If the ordering variable is the rowvar/colvar in the input table and the descriptive variable is in rowlut/collut, set row.orderby/col.orderby equal to rowvar/colvar. If the descriptive variable is the rowvar/colvar in the input table, and the ordering code variable is in rowlut/collut, set row.orderby/col.orderby equal to the variable name of the code variable in rowlut/collut.

UNITS:
The following variables are converted from pounds (from FIA database) to short tons by multiplying the variable by 0.0005. DRYBIO_AG, DRYBIO_BG, DRYBIO_WLDP_SPP, DRYBIO_SAPLING, DRYBIO_STUMP, DRYBIO_TOP, DRYBIO_BOLE, DRYBIOT, DRYBIOM, DRYBIOTB, JBIOTOT, CARBON_BG, CARBON_AG

MORTALITY:
For Interior-West FIA, mortality estimates are mainly based on whether a tree has died within the last 5 years of when the plot was measured. If a plot was remeasured, mortality includes trees that were alive the previous visit but were dead in the next visit. If a tree was standing the previous visit, but was not standing in the next visit, no diameter was collected (DIA = NA) but the tree is defined as mortality.

Common tree filters:

<table>
<thead>
<tr>
<th>FILTER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;STATUSCD == 1&quot;</td>
<td>Live trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2&quot;</td>
<td>Dead trees</td>
</tr>
<tr>
<td>&quot;TPAMORT_UNADJ &gt; 0&quot;</td>
<td>Mortality trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; DIA &gt;= 5.0&quot;</td>
<td>Dead trees &gt;= 5.0 inches diameter</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; AGENTCD == 30&quot;</td>
<td>Dead trees from fire</td>
</tr>
</tbody>
</table>

Author(s)

Tracey S. Frescino

References

Examples

# Set up population dataset (see ?modMApop() for more information)
MApopdat <- modMApop(popTabs = list(tree = FIESTA::WYtree,
  cond = FIESTA::WYcond),
  pltassgn = FIESTA::WYpltassgn,
  pltassgnid = "CN",
  unitarea = FIESTA::WYunitarea,
  unitvar = "ESTN_UNIT",
  unitzonal = FIESTA::WYunitzonal,
  prednames = c("dem", "tcc", "tpi", "tnt"),
  predfac = "tnt")

# Use GREG estimator to estimate area of forest land in our population
modMAarea(MApopdat = MApopdat,
  MAmethod = "greg",
  landarea = "FOREST")

# Use GREG estimator to estimate area of forest land by forest type and # stand-size class
modMAarea(MApopdat = MApopdat,
  MAmethod = "greg",
  landarea = "FOREST",
  rowvar = "FORTYPCD",
  colvar = "STDSZCD")

---

**modMApop**

*Model-Assisted module - Generate population data for MA module.*

**Description**

Generates population data for generating model-assisted estimation. Plots that are totally nonsampled are excluded from estimation dataset. Next, an adjustment factor is calculated by strata (if MAmethod="PS") or by estimation unit to adjust for nonsampled (nonresponse) conditions that have proportion less than 1. Attributes adjusted to a per-acre value are summed by plot, divided by the adjustment factor, and averaged by stratum and/or estimation unit. Note: population data must be generated by MA method.

**Usage**

modMApop(
  popType = "VOL",
  popTabs = popTables(),
  popTabIDs = popTableIDs(),
  popFilter = popFilters(),
  pltassgn = NULL,
  pltassgnid = "PLT_CN",
)

dsn = NULL,
pjoinid = "CN",
areawt = "CONDPROP_UNADJ",
adjplot = TRUE,
unitvar = NULL,
unitarea = NULL,
areavar = "ACRES",
unitzonal = NULL,
prednames = NULL,
predfac = NULL,
standardize = TRUE,
savedata = FALSE,
saveobj = FALSE,
objnm = "MApopdat",
unit_opts = NULL,
savedata_opts = NULL,
MADATA = NULL,
pltdat = NULL,
auxdat = NULL,
gui = FALSE,
...
)

Arguments

popType String. Type of evaluation(s) to include in population data. Note: currently only c('CURR', 'VOL', 'LULC') are available. See details below for descriptions of each.

popTabs List of population tables the user would like returned. See help(popTables) for a list of options.

popTabIDs List of unique IDs corresponding to the population tables that the user has requested. See help(popTableIDs) for a list of options.

popFilter List of population filters. See help(popFilters) for a list of options.

pltassgn DF/DT, Optional. R object, sf R object, comma-delimited file,.csv), layer or spatial layer in dsn, or shapefile(.shp). Plot-level assignment of estimation unit and/or strata, with one record for each plot.

pltassgnid String.


pjoinid String. Join variable in plot to match pltassgnid. Does not need to be unique. If using most current XY coordinates for plot assignments, use identifier for plot (e.g., PLOT_ID).

areawt String. Name of variable for summarizing area weights (e.g., CONDPROP_UNADJ).

adjplot Logical. If TRUE, adjusts for nonresponse at plot-level.

unitvar String. Name of the estimation unit variable in unitarea and cond or pltassgn data frame with estimation unit assignment for each plot (e.g., 'ESTN_UNIT'). Optional if only one estimation unit.
unitarea  Numeric or DF. Total area by estimation unit. If only 1 estimation unit, include number of total acreage for the area of interest or a data frame with area and estimation unit. If more than one estimation unit, provide a data frame of total area by estimation unit, including unitvar and areavar.

areavar  String. Name of area variable in unitarea. Default="ACRES".

unitzonal  DF/DT. Table with zonal auxiliary information by estimation unit. For continuous data, means by estimation unit; for categorical data, proportion of class by estimation unit.

prednames  String vector. Name(s) of predictor variables to include in model.

predfac  String vector. Name(s) of prednames that are factors (i.e., categorical). Names will change in output depending on number of categories.

standardize  Logical. If TRUE, predictors are standardized.

savedata  Logical. If TRUE, saves table(s) to outfolder.

saveobj  Logical. If TRUE, saves returned list object to outfolder.

objnm  String. Name of *.rds object.

unit_opts  List. See help(unit_options()) for a list of options.

savedata_opts  List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

MAdata  List. Data output from FIESTA::MAdata().

pltdat  R List object. Output data list components from FIESTA::spGetPlots().

auxdat  List. Auxiliary data output from FIESTA::spGetAuxiliary().

gui  Logical. If gui, user is prompted for parameters.

...  For extendibility.

Details

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (e.g. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>TPA_UNADJ</td>
<td>Number of trees per acre each sample tree represents (e.g. DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on macroplot).</td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (e.g. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.).</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest, nonsampled nonforest).</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td></td>
<td>SUBPROP_UNADJ</td>
<td>Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 subplot per plot.</td>
</tr>
<tr>
<td></td>
<td>MICRPROP_UNADJ</td>
<td>If microplot tree attributes. Unadjusted proportion of microplot conditions on each microplot.</td>
</tr>
<tr>
<td></td>
<td>MACRPROP_UNADJ</td>
<td>If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each macroplot.</td>
</tr>
</tbody>
</table>
pltassign puniqueid Unique identifier for each plot, to link to cond (e.g. CN).
STATECD Identifies state each plot is located in.
INVYR Identifies inventory year of each plot.
PLOT_STATUS_CD Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed as sampled.

For available reference tables: sort(unique(FIESTAutils::ref_codes$VARIABLE))

**Value**

A list with population data for Green-Book estimates.

- **condx** Data frame. Condition-level data including plot-level assignment of estimation unit and stratum (if strata=TRUE) and adjusted condition proportion.
- **pltcondx** Data frame. Condition-level data, merged with plot data.
- **cuniqueid** String. Unique identifier of plot in condx and pltcondx.
- **condid** String. Unique identifier of condition in condx and pltcondx.
- **treex** Data frame. If esttype='TREE', tree-level data, including sample adjustment factor.
- **tuniqueid** String. If esttype='TREE', unique identifier of plot in treex.
- **ACI.filter** String. If ACI=FALSE, ACI.filter="COND_STATUS_CD == 1".
- **unitarea** String. Returned table of area by estimation unit.
- **unitvar** String. Variable name for estimation unit.
- **strlut** String. Strata-level table with pixel counts by strata (P1POINTCNT), strata weights (strwt), number of plots by strata (n.strata), total number of plots in estimation unit (n.total), sum of condition proportions (_UNADJ_SUM), area adjustments (*_ADJFAC), total area, and area expansion by strata (EXPNS).
- **strvar** String. Variable name for strata. If strata=FALSE, strvar="ONESTRAT".
- **expcondtab** String. If ACI=FALSE, ACI.filter="COND_STATUS_CD == 1".
- **plotsampcnt** Data frame. Number of plots by PLOT_STATUS_CD.
- **condsampcnt** Data frame. Number of conditions by COND_STATUS_CD.
- **states** String. State names in dataset.
- **invyrs** String. Range of inventory years in dataset.
- **stratdat** Data frame. Strata information by estimation unit.

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>stratum value</td>
</tr>
<tr>
<td>strwtvar</td>
<td>number of pixels by strata and estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or plots) by strata and estimation unit (i.e., strata weight)</td>
</tr>
</tbody>
</table>
Table(s) are also written to outfolder.

**Note**

**ADJUSTMENT FACTOR:**

The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata, by summing the unadjusted proportions of the subplot, microplot, and macroplot (i.e. *PROP_UNADJ) and dividing by the number of plots in the strata/estimation unit.

An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor. If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

**unitcombine:**

If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.

**stratcombine:**

If TRUE and less than 2 plots in any one strata class within an estimation unit, all strata classes with 2 or less plots are combined. The current method for combining is to group the strata with less than 2 plots with the strata class following in consecutive order (numeric or alphabetical), restrained by estimation unit (if unitcombine=FALSE), and continuing until the number of plots equals 10. If there are no strata classes following in order, it is combined with the estimation unit previous in order.

**Author(s)**

Tracey S. Frescino, Paul L. Patterson
modMAtree

References


Examples

# NOTE: FIA data objects used in these examples are stored in 'FIESTA', but  # can be generated for populations of interest by the user with functions in  # 'FIESTA' such as `spGetPlots()`, `spGetAuxiliary()`, etc. For more  # information, see 'FIESTA'’s extensive vignettes.

# Population data for counties in Wyoming
modMApop(popTabs = list(tree = FIESTA::WYtree,  
    cond = FIESTA::WYcond),  
    pltassgn = FIESTA::WYpltassgn,  
    pltassgnid = "CN",  
    unitarea = FIESTA::WYunitarea,  
    unitvar = "ESTN_UNIT",  
    unitzonal = FIESTA::WYunitzonal,  
    prednames = c("dem", "tcc", "tpi", "tnt"),  
    predfac = "tnt")

# Adding seedling data as well
modMApop(popTabs = list(tree = FIESTA::WYtree,  
    cond = FIESTA::WYcond,  
    seed = FIESTA::WYseed),  
    pltassgn = FIESTA::WYpltassgn,  
    pltassgnid = "CN",  
    unitarea = FIESTA::WYunitarea,  
    unitvar = "ESTN_UNIT",  
    unitzonal = FIESTA::WYunitzonal,  
    prednames = c("dem", "tcc", "tpi", "tnt"),  
    predfac = "tnt")

---

modMAtree

Model-Assisted module - Generate model-assisted tree estimates.

Description

Generates tree estimates by estimation unit. Estimates are calculated from McConville et al. (2018)’s mase R package.
Usage

```r
modMAtree(
  MApopdat,
  MAmethod,
  estvar,
  estvar.filter = NULL,
  estseed = "none",
  landarea = "FOREST",
  pcfilter = NULL,
  rowvar = NULL,
  colvar = NULL,
  prednames = NULL,
  modelselect = FALSE,
  FIA = TRUE,
  bootstrap = FALSE,
  returntitle = FALSE,
  savedata = FALSE,
  table_opts = NULL,
  title_opts = NULL,
  savedata_opts = NULL,
  gui = FALSE,
  ...
)
```

Arguments

- **MApopdat**: List. Population data objects returned from `modMApop()`.
- **MAmethod**: String. mase (i.e., model-assisted) method to use ('greg', 'gregEN', 'ratio').
- **estvar**: String. Name of the tree-level estimate variable (e.g., 'VOLCFNET').
- **estvar.filter**: String. A tree-level filter for estvar. Must be R syntax (e.g., 'STATUSCD == 1').
- **estseed**: String. Use seedling data only or add to tree data. Seedling estimates are only for counts (estvar='TPA_UNADJ')-(none', 'only', 'add').
- **landarea**: String. The condition-level filter for defining land area ('ALL', 'FOREST', 'TIMBERLAND'). If landarea='FOREST', COND_STATUS_CD = 1; if landarea='TIMBERLAND', SITECLCD in(1:6) & RESERVCD = 0.
- **pcfilter**: String. A filter for plot or cond attributes (including pltassign). Must be R logical syntax.
- **rowvar**: String. Optional. Name of domain variable to group estvar by for rows in table output. Rowvar must be included in an input data frame (i.e., plt, cond, tree). If no rowvar is included, an estimate is returned for the total estimation unit. Include colvar for grouping by 2 variables.
- **colvar**: String. Optional. If rowvar != NULL, name of domain variable to group estvar by for columns in table output. Colvar must be included in an input data frame (i.e., plt, cond, tree).
- **prednames**: String vector. Name(s) of predictor variables to include in model.
**modMAtree**

- `modelselect` Logical. If TRUE, variable selection occurs.
- `FIA` Logical. If TRUE, the finite population term is removed from estimator to match FIA estimates.
- `bootstrap` Logical. If TRUE, returns bootstrap variance estimates, otherwise uses Horvitz-Thompson estimator under simple random sampling without replacement.
- `returntitle` Logical. If TRUE, returns title(s) of the estimation table(s).
- `savedata` Logical. If TRUE, saves table(s) to outfolder.
- `table_opts` List. See help(table_options()) for a list of options.
- `title_opts` List. See help(title_options()) for a list of options.
- `savedata_opts` List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.
- `gui` Logical. If gui, user is prompted for parameters.
- ...

Parameters for `modMApop()` if MApopdat is NULL.

**Details**

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (e.g. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>TPA_UNADJ</td>
<td>Number of trees per acre each sample tree represents (e.g., DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on microplot).</td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.).</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest, nonsampled nonforest).</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td></td>
<td>SUBPROP_UNADJ</td>
<td>Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.</td>
</tr>
<tr>
<td></td>
<td>MICROPROP_UNADJ</td>
<td>If microplot tree attributes. Unadjusted proportion of microplot conditions on each plot.</td>
</tr>
<tr>
<td></td>
<td>MACROPROP_UNADJ</td>
<td>If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot.</td>
</tr>
<tr>
<td>pltassgn</td>
<td>puniqueid</td>
<td>Unique identifier for each plot, to link to cond (ex. CN).</td>
</tr>
<tr>
<td></td>
<td>STATECD</td>
<td>Identifies state each plot is located in.</td>
</tr>
<tr>
<td></td>
<td>INVYR</td>
<td>Identifies inventory year of each plot.</td>
</tr>
<tr>
<td></td>
<td>PLOT_STATUS_CD</td>
<td>Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed sampled.</td>
</tr>
</tbody>
</table>

Reference names are available for the following variables:

ADFORCD, AGENTCD, CCLCD, DECAYCD, DSTRBCD, KINDCD, OWNCD, OWNGRPCD, FORTYPCD, FLDTYPCD, FORTYPCDCALC, TPGRPDC, FORINDCD, RESERVCD, LANDCLCD, STDSZCD, FLDSZCD, PHYSCLCD, MIST_CL_CD, PLOT_STATUS_CD, STATECD, TREECLCD, TRTCD, SPCD, SPGRPCD
Value

If FIA=TRUE or unitvar=NULL and colvar=NULL, one data frame is returned with tree estimates and percent sample errors. Otherwise, a list is returned with tree estimates in one data frame (est) and percent sample errors in another data frame (est.pse). If rawdata=TRUE, another list is returned including raw data used in the estimation process. If addtitle=TRUE and returntitle=TRUE, the title for est/pse is returned. If savedata=TRUE, all data frames are written to outfolder.

**est**
Data frame. Tree estimates by rowvar, colvar (and estimation unit). If FIA=TRUE or one estimation unit and colvar=NULL, estimates and percent sampling error are in one data frame.

**pse**
Data frame. Percent sampling errors for estimates by rowvar and colvar (and estimation unit).

**titlelst**
List with 1 or 2 string vectors. If returntitle=TRUE a list with table title(s). The list contains one title if est and pse are in the same table and two titles if est and pse are in separate tables.

**raw**
List of data frames. If rawdata=TRUE, a list including: number of plots by plot status, if in dataset (plotsampcnt); number of conditions by condition status (condsampcnt); data used for post-stratification (stratdat); and 1-8 tables with calculated variables used for processing estimates and percent sampling error for table cell values and totals (See processing data below).

Raw data

**plotsampcnt**
Table. Number of plots by plot status (ex. sampled forest on plot, sampled nonforest, nonsampled).

**condsampcnt**
DF. Number of conditions by condition status (forest land, nonforest land, non-census water, census water, nonsampled).

**stratdat**
Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESTUNIT</td>
<td>estimation unit</td>
</tr>
<tr>
<td>STRATA</td>
<td>strata</td>
</tr>
<tr>
<td>ACRES</td>
<td>area by strata for estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (and estimation unit)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>TOTACRES</td>
<td>total area for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or number of plots) by strata (strata weight)</td>
</tr>
<tr>
<td>expfac.strata</td>
<td>expansion factor (in area unit (e.g., acres) by strata (areavar/n.strata)</td>
</tr>
</tbody>
</table>

processing data

Data frames. Separate data frames containing calculated variables used in estimation process. The number of processing tables depends on the input parameters. The tables include: total by estimation unit (unit.totest); rowvar totals (unit.rowest), and if colvar is not NULL, colvar totals, (unit.colvar); and a combination of rowvar and colvar (unit.grpvar). If FIA=TRUE, the raw data for the summed estimation units are also included (totest, rowest, colest, grpest, respec-
These tables do not include estimate proportions (nhat and nhat.var). The data frames include the following information:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nhat</td>
<td>estimated proportion of trees</td>
</tr>
<tr>
<td>nhat.var</td>
<td>estimated variance of estimated proportion of trees</td>
</tr>
<tr>
<td>ACRES</td>
<td>total area for estimation unit</td>
</tr>
<tr>
<td>est</td>
<td>estimated area of trees nhat*ACRES</td>
</tr>
<tr>
<td>est.var</td>
<td>estimated variance of estimated area of trees nhat.var*areavar^2</td>
</tr>
<tr>
<td>est.se</td>
<td>standard error of estimated area of trees sqrt(est.var)</td>
</tr>
<tr>
<td>est.cv</td>
<td>coefficient of variation of estimated area of trees est.se/est</td>
</tr>
<tr>
<td>pse</td>
<td>percent sampling error of estimate est.cv*100</td>
</tr>
<tr>
<td>CI99left</td>
<td>left tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI99right</td>
<td>right tail of 99 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95left</td>
<td>left tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI95right</td>
<td>right tail of 95 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67left</td>
<td>left tail of 67 percent confidence interval for estimated area</td>
</tr>
<tr>
<td>CI67right</td>
<td>right tail of 67 percent confidence interval for estimated area</td>
</tr>
</tbody>
</table>

Table(s) are also written to outfolder.

**Note**

**ADJUSTMENT FACTOR:**
The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata. by summing the unadjusted proportions of the subplot, microplot, and macroplot (i.e. *PROP_UNADJ) and dividing by the number of plots in the strata/estimation unit). An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor. If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

An estimation unit is a population, or area of interest, with known area and number of plots. Individual counties or combined Super-counties are common estimation units for FIA. An estimation unit may also be a subpopulation of a larger population (e.g., Counties within a State). Subpopulations are mutually exclusive and independent within a population, therefore estimated totals and variances are additive. For example, State-level estimates are generated by summing estimates from
all subpopulations within the State (Bechtold and Patterson. 2005. Chapter 2). Each plot must be assigned to only one estimation unit.

stratcombine:
If MAmethod='PS', and stratcombine=TRUE, and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.

autoxreduce:
If MAmethod='GREG', and autoxreduce=TRUE, and there is an error because of multicolinearity, a variable reduction method is applied to remove correlated variables. The method used is based on the variance-inflation factor (vif) from a linear model. The vif estimates how much the variance of each x variable is inflated due to multicolinearity in the model.

rowlut/collut:
There are several objectives for including rowlut/collut look-up tables: 1) to include descriptive names that match row/column codes in the input table; 2) to use number codes that match row/column names in the input table for ordering rows; 3) to add rows and/or columns with 0 values for consistency. No duplicate names are allowed.

Include 2 columns in the table:
1-the merging variable with same name as the variable in the input merge table;
2-the ordering or descriptive variable.
If the ordering variable is the rowvar/colvar in the input table and the descriptive variable is in rowlut/collut, set row.orderby/col.orderby equal to rowvar/colvar. If the descriptive variable is the rowvar/colvar in the input table, and the ordering code variable is in rowlut/collut, set row.orderby/col.orderby equal to the variable name of the code variable in rowlut/collut.

UNITS:
The following variables are converted from pounds (from FIA database) to short tons by multiplying the variable by 0.0005. DRYBIO_AG, DRYBIO_BG, DRYBIO_WDL_D_SPP, DRYBIO_SAPLING, DRYBIO_STUMP, DRYBIO_TOP, DRYBIO_BOLE, DRYBIOT, DRYBIOM, DRYBIOTB, JBIOTOT, CARBON_BG, CARBON_AG

MORTALITY:
For Interior-West FIA, mortality estimates are mainly based on whether a tree has died within the last 5 years of when the plot was measured. If a plot was remeasured, mortality includes trees that were alive the previous visit but were dead in the next visit. If a tree was standing the previous visit, but was not standing in the next visit, no diameter was collected (DIA = NA) but the tree is defined as mortality.

Common tree filters:

<table>
<thead>
<tr>
<th>FILTER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;STATUSCD == 1&quot;</td>
<td>Live trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2&quot;</td>
<td>Dead trees</td>
</tr>
<tr>
<td>&quot;TPAMORT_UNADJ &gt; 0&quot;</td>
<td>Mortality trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; DIA &gt;= 5.0&quot;</td>
<td>Dead trees &gt;= 5.0 inches diameter</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; AGENTCD == 30&quot;</td>
<td>Dead trees from fire</td>
</tr>
</tbody>
</table>
modPB

Author(s)

Tracey S. Frescino

References


Examples

# Set up population dataset (see ?modMApop() for more information)
MApopdat <- modMApop(popTabs = list(tree = FIESTA::WYtree,
                      cond = FIESTA::WYcond),
                      pltassgn = FIESTA::WYpltassgn,
                      pltassgnid = "CN",
                      unitarea = FIESTA::WYunitarea,
                      unitvar = "ESTN_UNIT",
                      unitzonal = FIESTA::WYunitzonal,
                      prednames = c("dem", "tcc", "tpi", "tnt"),
                      predfac = "tnt")

# Use GREG Estimator to Estimate cubic foot volume of live trees in our
# population
modMAtree(MApopdat = MApopdat,
          MAmethod = "greg",
          estvar = "VOLCFNET",
          estvar.filter = "STATUSCD == 1")

# Use GREG Elastic Net Estimator to Estimate basal area of live trees in our
# population
modMAtree(MApopdat = MApopdat,
          MAmethod = "gregEN",
          estvar = "BA",
          estvar.filter = "STATUSCD == 1")

modPB

Photo-Based module - Generate photo-based estimates.

Description

Generates percent, area or ratio-of-means estimates, with associated sampling error by domain (and estimation unit). Calculations are based on Patterson (2012) photo-based estimators for the Nevada photo-based inventory.
Usage

modPB(
    PBpopdat = NULL,
    tabtype = "PCT",
    sumunits = FALSE,
    ratio = FALSE,
    landarea = "ALL",
    landarea.filter = NULL,
    nonsamp.pntfilter = NULL,
    pntfilter = NULL,
    pfilter = NULL,
    rowvar = NULL,
    colvar = NULL,
    domlut = NULL,
    domvarlst = NULL,
    ratioden = "ROWVAR",
    gainloss = FALSE,
    gainloss.vals = NULL,
    addtitle = FALSE,
    returntitle = FALSE,
    savedata = FALSE,
    table_opts = NULL,
    title_opts = NULL,
    savedata_opts = NULL,
    gui = FALSE,
    ...
)

Arguments

PBpopdat List. Population data objects returned from modPBpop().
tabtype String. Type of units for the table ("PCT", "AREA").
sumunits Logical. If TRUE, estimation units are combined to one table for output. Note: only available if tabtype="AREA". Acres
ratio Logical. If TRUE, ratio estimates are generated.
landarea String. Sample area for estimates ("ALL", "CHANGE"). Used to describe landarea.filter.
landarea.filter String. filter for land area. Must be R syntax.
nonsamp.pntfilter String. An expression for filtering nonsampled points (e.g., cloud coverage). Must be R syntax.
pntfilter String. A global filter for the pnt file. Must be R syntax.
pfilter String. A global filter for the plt file. Must be R syntax.
rowvar String. Name of domain variable in pnt used for output estimation table rows. If only 1 domain, must be rowvar. If no domain, rowvar=NULL.
colvar String. Name of domain variable in pnt used for output estimation table columns. If only 1 domain, colvar=NULL.

domlut DF/DT or comma-delimited (*.csv). Look-up table to define the variables in the pnt table with category codes (DOMCODE) and code names (DOMNAME), and to set a pretty name for the variable to use in output table (DOMTITLE). This table is also used to populate rowvar/colvar, row.orderby/col.orderby, and title.rowvar/title.colvar parameters. Optional.

donvarlst String vector. A vector of variable names that can be row or column domains (codes and names). Optional.

ratioden String. ("ROWVAR" or "COLVAR"). If ratio, defines whether the rowvar or colvar in estimation output table is the denominator.

gainloss Logical. If TRUE, a table with the difference of gain and loss along with the variance and standard error, in percent, is generated.

gainloss.vals String vector. A vector of names for values in gainloss table.

addtitle Logical. If TRUE and savedata=TRUE, adds title to outfile.

returntitle Logical. If TRUE, returns a character string of the title of the output data frame.

savedata Logical. If TRUE, saves table(s) to outfolder.

table_opts List. See help(table_options()) for a list of options.

title_opts List. See help(title_options()) for a list of options.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

gui Logical. If gui, user is prompted for parameters.

... Parameters for modPBpop() if PBpopdat is NULL.

Details

If variables are NULL, then it will prompt user to input variables.

Value

A list with estimates with percent sampling error for rowvar (and colvar). If sumunits=TRUE or unitvar=NULL and colvar=NULL, one data frame is returned. Otherwise, a list object is returned with the following information. If savedata=TRUE, all data frames are written to outfolder.

est DF. Estimated percent cover or area by rowvar, colvar, (and estimation unit).

pse DF. Percent sampling error of estimates by rowvar, colvar (and estimation unit).

titlelst List with 1 or 2 string vectors. If returntitle=TRUE a list with table title(s). The list contains one title if est and pse are in the same table and two titles if est and pse are in separate tables. Row and column tables are also included in list.

raw List of data frames. If rawdata=TRUE, a list including the processing data used for estimation including: number of plots and conditions; stratification information; and 1 to 8 tables with calculated values for table cells and totals (See processing data below).

Raw data
Table. Number of points by rowvar/colvar (sampled and nonsampled).

Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>strata</td>
</tr>
<tr>
<td>areavar</td>
<td>If tabtype='AREA', area by strata for estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>TOTAREA</td>
<td>If tabtype='AREA', total area for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or number of plots) by strata (strata weight)</td>
</tr>
</tbody>
</table>

Data frames. Separate data frames of variables used in estimation process for the rowvar, colvar and combination of rowvar and colvar (if colvar is not NULL), and grand total by estimation unit (unit.rowest, unit.colest, unit.grpest, respectively) and summed estimation units, if FIA=TRUE (rowest, colest, grpest, respectively).

The data frames include the following information:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phat</td>
<td>estimated proportion of covered land</td>
</tr>
<tr>
<td>phat.var</td>
<td>variance of estimated proportion of covered land</td>
</tr>
<tr>
<td>areavar</td>
<td>If tabtype='AREA', total area for estimation unit</td>
</tr>
<tr>
<td>est</td>
<td>If tabtype='AREA', estimated area of land phat*areavar</td>
</tr>
<tr>
<td>est.var</td>
<td>variance of estimated area of land phat.var*areavar</td>
</tr>
<tr>
<td>est.se</td>
<td>standard error of estimated area or percent cover</td>
</tr>
<tr>
<td>est.cv</td>
<td>coefficient of variance of estimated area or percent cover</td>
</tr>
<tr>
<td>est.pse</td>
<td>percent sampling error of estimated area of percent cover</td>
</tr>
<tr>
<td>CI99left</td>
<td>left tail of 99 percent confidence interval for est</td>
</tr>
<tr>
<td>CI99right</td>
<td>right tail of 99 percent confidence interval for est</td>
</tr>
<tr>
<td>CI95left</td>
<td>left tail of 95 percent confidence interval for est</td>
</tr>
<tr>
<td>CI95right</td>
<td>right tail of 95 percent confidence interval for est</td>
</tr>
<tr>
<td>CI67left</td>
<td>left tail of 67 percent confidence interval for est</td>
</tr>
<tr>
<td>CI67right</td>
<td>right tail of 67 percent confidence interval for est</td>
</tr>
</tbody>
</table>

If tabtype='PCT', estimated percent cover of land phat.var*100

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phat.n</td>
<td>estimated proportion of covered land, for numerator (colvar)</td>
</tr>
<tr>
<td>phat.var.n</td>
<td>variance of estimated proportion of covered land, for numerator (colvar)</td>
</tr>
<tr>
<td>phat.d</td>
<td>estimated proportion of covered land, for denominator (rowvar)</td>
</tr>
<tr>
<td>phat.var.d</td>
<td>variance of estimated proportion of covered land, for denominator (rowvar)</td>
</tr>
<tr>
<td>covar</td>
<td>covariance of estimated proportion of numerator (rowvar) and denominator (colvar)</td>
</tr>
<tr>
<td>rhat</td>
<td>ratio of estimated proportions (numerator-colvar)</td>
</tr>
</tbody>
</table>

if ratio=TRUE:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phat.n</td>
<td>estimated proportion of covered land, for numerator (colvar)</td>
</tr>
<tr>
<td>phat.var.n</td>
<td>variance of estimated proportion of covered land, for numerator (colvar)</td>
</tr>
<tr>
<td>phat.d</td>
<td>estimated proportion of covered land, for denominator (rowvar)</td>
</tr>
<tr>
<td>phat.var.d</td>
<td>variance of estimated proportion of covered land, for denominator (rowvar)</td>
</tr>
<tr>
<td>covar</td>
<td>covariance of estimated proportion of numerator (rowvar) and denominator (colvar)</td>
</tr>
<tr>
<td>rhat</td>
<td>ratio of estimated proportions (numerator-colvar)</td>
</tr>
</tbody>
</table>
Note

**STRATA:**
Stratification is used to reduce variance in population estimates by partitioning the population into homogenous classes (strata), such as forest and nonforest. For stratified sampling methods, the strata sizes (weights) must be either known or estimated. Remotely-sensed data is often used to generate strata weights with proportion of pixels by strata. If stratification is desired (strata=TRUE), the required data include: stratum assignment for the center location of each plot, stored in either pltassgn or cond; and a look-up table with the area or proportion of the total area of each strata value by estimation unit, making sure the name of the strata (and estimation unit) variable and values match the plot assignment name(s) and value(s).

**sumunits:**
An estimation unit is a population, or area of interest, with known area and number of plots. Individual counties or combined Super-counties are common estimation units for FIA. An estimation unit may also be a subpopulation of a larger population (e.g., Counties within a State). Subpopulations are mutually exclusive and independent within a population, therefore estimated totals and variances are additive. For example, State-level estimates are generated by summing estimates from all subpopulations within the State (Bechtold and Patterson. 2005. Chapter 2). Each plot must be assigned to only one estimation unit.

If sumunits=TRUE, estimates are generated by estimation unit, summed together, and returned as one estimate. If rawdata=TRUE, estimates by individual estimation unit are also returned.

If sumunits=FALSE, estimates are generated and returned by estimation unit as one data frame. If savedata=TRUE, a separate file is written for each estimation unit.

**stratcombine:**
If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10.
If there are no estimation units following in order, it is combined with the estimation unit previous in order.

rowlut/collut:
There are several objectives for including rowlut/collut look-up tables: 1) to include descriptive names that match row/column codes in the input table; 2) to use number codes that match row/column names in the input table for ordering rows; 3) to add rows and/or columns with 0 values for consistency. No duplicate names are allowed.

Include 2 columns in the table:
1-the merging variable with same name as the variable in the input merge table;
2-the ordering or descriptive variable.
If the ordering variable is the rowvar/colvar in the input table and the descriptive variable is in rowlut/collut, set row.orderby/col.orderby equal to rowvar/colvar. If the descriptive variable is the rowvar/colvar in the input table, and the ordering code variable is in rowlut/collut, set row.orderby/col.orderby equal to the variable name of the code variable in rowlut/collut.

Author(s)
Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

References

Examples
# Load necessary data from FIESTA
## Point data
icepntfn <- system.file("extdata",
                       "PB_data/icepnt_utco1135.csv",
                       package = "FIESTA")
icepnt <- read.csv(icepntfn)

## Plot data
icepltfn <- system.file("extdata",
                        "PB_data/icepltassign_utco1135.csv",
                        package = "FIESTA")
iceplt <- read.csv(icepltfn)

## County data
unitareafn <- system.file("extdata",
                          "PB_data/unitarea_utco1135.csv",
                          package = "FIESTA")
unitarea <- read.csv(unitareafn)
## ICE Cover

```r
coverfn <- system.file("extdata",
  "PB_data/cover_LUT.csv",
  package = "FIESTA")

cover <- read.csv(covfn)
names(cov) <- sub("cover", "cover_1", names(cov))

# Set up population data (see ?modPBpop() for more information)
PBpopunit <- modPBpop(pnt = icepnt,
  pltassgn = iceplt,
  pltassgnid = "plot_id",
  pntid = "dot_cnt",
  unitarea = unitarea,
  unitvar = "ESTN_UNIT")

# Photo-based estimation with point-level data by estimation unit (county)
## Without summing units
cover1.unit.area <- modPB(
  PBpopdat = PBpopunit,
  tabtype = "AREA",
  rowvar = "cover_1",
  nonsamp.pntfilter = "cover_1 != 999",
  table_opts = list(rowlut = cov),
  title_opts = list(title.rowvar = "Land Cover (2011)"))

cover1.unit.area$est

## With summing units
cover1.unit.area.sum <- modPB(
  PBpopdat = PBpopunit,
  tabtype = "AREA",
  rowvar = "cover_1",
  nonsamp.pntfilter = "cover_1 != 999",
  sumunits = TRUE,
  table_opts = list(rowlut = cov),
  title_opts = list(title.rowvar = "Land Cover (2011)"))

cover1.unit.area.sum$est
```

---

**modPBpop**  
*Photo-Based module - Generate population data for PB module.*

### Description

Generates population data for generating photo-based estimation. Plots that are totally nonsampled are excluded from estimation dataset. Next, an adjustment factor is calculated by strata to adjust for nonsampled (nonresponse) conditions that have proportion less than 1. Attributes adjusted to a per-acre value are summed by plot, divided by the adjustment factor, and averaged by stratum.
Strata means are combined using the strata weights and then expanded to using the total land area in the population.

Usage

```r
modPBpop(
  pntdat = NULL,
  pltpct = NULL,
  plotid = "plot_id",
  pntid = NULL,
  pltpctvars = NULL,
  plt = NULL,
  pltassgn = NULL,
  puniqueid = "CN",
  pltassgnid = "CN",
  nonsamp.pfilter = NULL,
  sumunits = FALSE,
  unitvar = NULL,
  unitarea = NULL,
  areavar = "ACRES",
  strata = FALSE,
  strtype = "POST",
  stratalut = NULL,
  strvar = "STRATUMCD",
  pvars2keep = NULL,
  saveobj = FALSE,
  objnm = "PBpopdat",
  savedata = FALSE,
  unit_opts = NULL,
  strata_opts = NULL,
  savedata_opts = NULL,
  PBstratdat = NULL,
  gui = FALSE
)
```

Arguments

- **pntdat**: DF/DT or comma-delimited file (*.csv). Point-level table with one record per point. If NULL, aggregated point counts must be in pntcnt.
- **pltpct**: DF/DT or comma-delimited file (*.csv). Plot-domain-level table with percent observed by domain per plot.
- **plotid**: String. Unique identifier of plot in pnt. All values must match puniqueid values, if pltassgn is not NULL.
- **pntid**: String. Unique identifier of points in pnt.
- **pltpctvars**: String vector. Variables in pltpct for estimation. If NULL, all variables are used except plotid in pltpct.
- **plt**: DF/DT, comma-separated values (CSV) file(*.csv), or layer in dsn. Can also be a shapefile(*.shp) with one record per plot, a spatial layer in dsn, or a sf R object.
modPBpop

Plot-level variables. If nonsampled plots are included, PLOT_STATUS_CD variable must be in table. Optional.

pltassgn DF/DT, comma-delimited file(*.csv), SpatialDataFrame, or shapefile(*.shp). The plot-level data with one record per plot, including estimation unit and/or strata information. Optional.

puniqueid String. Unique identifier of plot.

pltassgnid String. Name of unique identifier of plot in pltassgn with All values must match plotid values if pnt is not NULL.


sumunits Logical. If TRUE, estimation units are combined to one table for output. Note: only available if tabtype="AREA". Acres

unitvar String. Name of the estimation unit variable in cond or pltassgn with estimation unit assignment for each plot (e.g., 'ESTN_UNIT'). If one estimation unit, set unitvar=NULL.

unitarea Numeric or DF. Total area by estimation unit. If only 1 estimation unit, include number of total acreage for the area of interest or a data frame with areavar. If more than one estimation unit, provide a data frame of total area by estimation unit, including unitvar and areavar.

areavar String. Name of acre variable in unitarea. Default="ACRES".

strata Logical. If TRUE, add data information for stratification.

strtype String. If strata=TRUE, the type of strata ('POST', 'PRE'). Note: the variance equations are slightly different.

stratalut DF/DT. If strata=TRUE, look-up table with strata proportions ('strwt') by strata (and estimation unit). To calculate 'strwt', set getwt=TRUE and getwtvar= name of variable with information to calculate weights from (e.g., pixel counts)

strvar String. If strata=TRUE, name of the strata variable in stratalut and cond or pltassgn data frame with stratum assignment for each plot (Default = 'STRATUMCD').

pvars2keep String vector. Additional plot variables to keep in dataset.

saveobj Logical. If TRUE, saves SApopdat object to outfolder.

objnm String. Name of *.rda object.

savedata Logical. If TRUE, saves table(s) to outfolder.

unit_opts List. See help(unit_options()) for a list of options.

strata_opts List. See help(strata_options()) for a list of options. Only used when strata = TRUE.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

PBstratdat R List object. Output data list components from FIESTA::DBgetStrata().

gui Logical. If gui, user is prompted for parameters.
Details

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>TPA_UNADJ</td>
<td>Number of trees per acre each sample tree represents (ex. DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on microplot).</td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.).</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest, nonsampled nonforest).</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td></td>
<td>SUBPROP_UNADJ</td>
<td>Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.</td>
</tr>
<tr>
<td></td>
<td>MICRPROP_UNADJ</td>
<td>If microplot tree attributes. Unadjusted proportion of microplot conditions on each plot. Set MICRPROP_UNADJ=1, if only 1 condition per microplot.</td>
</tr>
<tr>
<td></td>
<td>MACRPROP_UNADJ</td>
<td>If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot. Set MACRPROP_UNADJ=1, if only 1 condition per macroplot.</td>
</tr>
<tr>
<td>pltassgn</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to cond (ex. CN).</td>
</tr>
<tr>
<td></td>
<td>STATECD</td>
<td>Identifies state each plot is located in.</td>
</tr>
<tr>
<td></td>
<td>INVYR</td>
<td>Identifies inventory year of each plot.</td>
</tr>
<tr>
<td></td>
<td>PLOT_STATUS_CD</td>
<td>Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed to be sampled.</td>
</tr>
</tbody>
</table>

For available reference tables: sort(unique(FIESTAutils::ref_codes$VARIABLE))

Value

A list with population data for Green-Book estimates.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>condx</td>
<td>Data frame. Condition-level data including plot-level assignment of estimation unit and stratum (if strata=TRUE) and adjusted condition proportion.</td>
</tr>
<tr>
<td>pltcondx</td>
<td>Data frame. Condition-level data, merged with plot data.</td>
</tr>
<tr>
<td>cuniqueid</td>
<td>String. Unique identifier of plot in condx and pltcondx.</td>
</tr>
<tr>
<td>condid</td>
<td>String. Unique identifier of condition in condx and pltcondx.</td>
</tr>
<tr>
<td>treex</td>
<td>Data frame. If esttype='TREE', tree-level data, including sample adjustment factor.</td>
</tr>
<tr>
<td>tuniqueid</td>
<td>String. If esttype='TREE', unique identifier of plot in treex.</td>
</tr>
<tr>
<td>ACI.filter</td>
<td>String. If ACI=FALSE, ACI.filter=&quot;COND_STATUS_CD == 1&quot;.</td>
</tr>
<tr>
<td>unitarea</td>
<td>String. Returned table of area by estimation unit.</td>
</tr>
<tr>
<td>unitvar</td>
<td>String. Variable name for estimation unit.</td>
</tr>
<tr>
<td>strlut</td>
<td>String. Strata-level table with pixel counts by strata (P1POINTCNT), strata weights (strwt), number of plots by strata (n.strata), total number of plots in</td>
</tr>
</tbody>
</table>
estimation unit (n.total), sum of condition proportions (*_UNADJ_SUM), area adjustments (*_ADJFAC), total area, and area expansion by strata (EXPNS).

strvar
String. Variable name for strata. If strata=FALSE, strvar="ONESTRAT".

expcondtab
String. If ACI=FALSE, ACI.filter="COND_STATUS_CD == 1".

plotsampcnt
Data frame. Number of plots by PLOT_STATUS_CD.

condsampcnt
Data frame. Number of conditions by COND_STATUS_CD.

states
String. State names in dataset.

invyrs
String. Range of inventory years in dataset.

stratdat
Data frame. Strata information by estimation unit.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unitvar</td>
<td>estimation unit</td>
</tr>
<tr>
<td>strvar</td>
<td>stratum value</td>
</tr>
<tr>
<td>strwtvar</td>
<td>number of pixels by strata and estimation unit</td>
</tr>
<tr>
<td>n.strata</td>
<td>number of plots in strata (after totally nonsampled plots removed)</td>
</tr>
<tr>
<td>n.total</td>
<td>number of plots for estimation unit</td>
</tr>
<tr>
<td>strwt</td>
<td>proportion of area (or plots) by strata and estimation unit (i.e., strata weight)</td>
</tr>
<tr>
<td>CONDPROP_UNADJ_SUM</td>
<td>summed condition proportion by strata and estimation unit</td>
</tr>
<tr>
<td>CONDPROP_ADJFAC</td>
<td>adjusted condition proportion by strata after nonsampled plots removed</td>
</tr>
<tr>
<td>AREA_USED</td>
<td>total area of estimation unit</td>
</tr>
<tr>
<td>expfac</td>
<td>strata-level expansion factor after nonsampled plots and conditions removed</td>
</tr>
<tr>
<td>EXPNS</td>
<td>strata-level area expansions (expfac * strwt)</td>
</tr>
</tbody>
</table>

Table(s) are also written to outfolder.

**Note**

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. It is calculated for each estimation unit by strata. by summing the unadjusted proportions of the subplot, microplot, and macroplot (i.e. *PROP_UNADJ) and dividing by the number of plots in the strata/estimation unit).

An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor.
If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.
unitcombine:
If TRUE and less than 2 plots in any one estimation unit, all estimation units with 10 or less plots are combined. The current method for combining is to group the estimation unit with less than 10 plots with the estimation unit following in consecutive order (numeric or alphabetical), restrained by survey unit (UNITCD) if included in dataset, and continuing until the number of plots equals 10. If there are no estimation units following in order, it is combined with the estimation unit previous in order.

stratcombine:
If TRUE and less than 2 plots in any one strata class within an estimation unit, all strata classes with 2 or less plots are combined. The current method for combining is to group the strata with less than 2 plots with the strata class following in consecutive order (numeric or alphabetical), restrained by estimation unit (if unitcombine=FALSE), and continuing until the number of plots equals 10. If there are no strata classes following in order, it is combined with the estimation unit previous in order.

Author(s)
Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

References

Examples
# Load necessary data from FIESTA
## Point data
icepntfn <- system.file("extdata",
                      "PB_data/icepnt_utco1135.csv",
                      package = "FIESTA")
icepnt <- read.csv(icepntfn)

## Plot data
icepltfn <- system.file("extdata",
                       "PB_data/icepltassgn_utco1135.csv",
                       package = "FIESTA")
iceplt <- read.csv(icepltfn)

# Percent land cover at Time 1 (2011) for all land in Davis and Salt Lake Counties, UT
PBpopdat <- modPBpop(pnt = icepnt,
                     pltassgn = iceplt,
                     pltassgnid = "plot_id",
                     pntid = "dot_cnt")
str(PBpopdat, max.level = 1)

# We can also create population data for estimates by estimation unit
## Read in data for multiple estimation units

```r
unitareafn <- system.file("extdata",  
  "PB_data/unitarea_utco1135.csv",  
  package = "FIESTA")
unitarea <- read.csv(unitareafn)
```

## Run modPBpop

```r
PBpopunit <- modPBpop(pnt = icepnt,  
  pltassgn = iceplt,  
  pltassgnid = "plot_id",  
  pntid = "dot_cnt",  
  unitarea = unitarea,  
  unitvar = "ESTN_UNIT")
```

---

**modSAarea**  
*Small area module - Generate small area tree estimates.*

### Description

Generates small area estimates by domain and/or tree domain (and estimation unit).

### Usage

```r
modSAarea(  
  SApopdatlst = NULL,  
  prednames = NULL,  
  SApackage = "JoSAE",  
  SAMethod = "area",  
  estseed = "none",  
  largebnd.unique = NULL,  
  landarea = "FOREST",  
  pcfilter = NULL,  
  rowvar = NULL,  
  modelselect = FALSE,  
  prior = function(x) 1/(sqrt(x) * (1 + x)),  
  na.fill = "NONE",  
  savedata = FALSE,  
  savesteps = FALSE,  
  multest = FALSE,  
  addSAdomsdf = TRUE,  
  SAdomvars = NULL,  
  savemultest = FALSE,  
  returntitle = FALSE,  
  table.opts = NULL,  
  title.opts = NULL,  
  savedata.opts = NULL,  
  multest.opts = NULL,  
  save4testing = FALSE)
```
guī = FALSE,
...
)

**Arguments**

- **SApopdatlst** List. List of population data objects returned from `modSApop()`.
- **prednames** String vector. Name(s) of predictor variables to use in model.
- **SApackage** String. Small area package to use (‘JoSAE’, ‘sae’, ‘hbsae’)
- **SAmethod** String. Small area method to use (‘unit’, ‘area’)
- **estseed** String. Use seedling data only or add to tree data. Seedling estimates are only for counts (estvar=‘TPA_UNADJ’)-(‘none’, ‘only’, ‘add’).
- **largebnd.unique** String. Name of the large boundary unique identifier to define plots within a model extent. If NULL, all plots are used for model extent.
- **landarea** String. The sample area filter for estimates (‘ALL’, ‘FOREST’, ‘TIMBERLAND’). If landarea=FOREST, filtered to COND_STATUS_CD = 1; If landarea=TIMBERLAND, filtered to SITECLCD in(1:6) and RESERVCD = 0.
- **pcfilter** String. A filter for plot or cond attributes (including pltassgn). Must be R logical syntax.
- **rowvar** String. Name of the row domain variable in cond or tree. If only one domain, rowvar = domain variable. If more than one domain, include colvar. If no domain, rowvar = NULL.
- **modelselect** Logical. If TRUE, selects useful predictors using mase:ElasticNet.
- **prior** Function. A prior function to use for hbsae models.
- **na.fill** String. An estimate to fill in for NA values (i.e., when model is unstable or no predictors are selected). Choose from the following list that does not include SApackage used (‘NONE’, ‘DIR’, ‘JoSAE’, ‘sae’, ‘hbsae’). DIR is suggested value for no NA values.
- **savedata** Logical. If TRUE, saves table(s) to outfolder.
- **savesteps** Logical. Saves graphs of predictors and response with labels whether selected or not for both area- and unit-level models.
- **multest** Logical. If TRUE, returns a data frame of SA estimates using both unit-level and area-level estimates.
- **addSAdomsdf** Logical. If TRUE, appends SAdomdf to unit.multest table for output.
- **SAdomvars** String vector. List of attributes from SAdoms to include in multest output.
- **savemultest** Logical. If TRUE, save table with area- and unit-level estimates.
- **returntitle** Logical. If TRUE, returns title(s) of the estimation table(s).
- **table_opts** List. See help(table_options()) for a list of options.
- **title_opts** List. See help(title_options()) for a list of options.
- **savedata_opts** List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.
multest_opts  List. See help(multest_options()) for a list of options. Only used when multest = TRUE.

save4testing Logical. If TRUE, saves intermediate steps as R objects to outfolder for testing (pdomdat, dunitlut).

gui Logical. If gui, user is prompted for parameters.

... Parameters for modSApop if SApopdat is NULL.

Details

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltstrat (e.g., PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>TPA_UNADJ</td>
<td>Number of trees per acre each sample tree represents (ex. DESIGNCD=1: TPA_UNADJ).</td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltstrat (ex. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.).</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest, nonforest, etc.).</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td></td>
<td>SUBPROP_UNADJ</td>
<td>Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.</td>
</tr>
<tr>
<td></td>
<td>MICRPROP_UNADJ</td>
<td>If microplot tree attributes. Unadjusted proportion of microplot conditions on each macroplot.</td>
</tr>
<tr>
<td></td>
<td>MACRPROP_UNADJ</td>
<td>If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot.</td>
</tr>
<tr>
<td>pltassign</td>
<td>puniqueid</td>
<td>Unique identifier for each plot, to link to cond (ex. CN).</td>
</tr>
<tr>
<td></td>
<td>STATECD</td>
<td>Identifies state each plot is located in.</td>
</tr>
<tr>
<td></td>
<td>INVYR</td>
<td>Identifies inventory year of each plot.</td>
</tr>
<tr>
<td></td>
<td>PLOT_STATUS_CD</td>
<td>Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed sampled.</td>
</tr>
</tbody>
</table>

Reference names are available for the following variables:
ADFORCD, AGENTCD, CCLCD, DECAYCD, DSTRBCD, KINDCD, OWNCD, OWNGRPCD, FORTYPED, FLDTYPCD, FORTYPDCALC, TYPGRPDCD, FORINDCD, RESERVCD, LANDCLCD, STDSZCD, FLDSZCD, PHYSCLCD, MIST_CL_CD, PLOT_STATUS_CD, STATECD, TREECLCD, TRTCD, SPCD, SGRPCD

Value

<table>
<thead>
<tr>
<th>est</th>
<th>Data frame. Tree estimates and percent sampling error by domain. Estimates are based on the SApackage and SAmethod parameters defined.</th>
</tr>
</thead>
<tbody>
<tr>
<td>titlelst</td>
<td>List. List of titles used for table output.</td>
</tr>
<tr>
<td>raw</td>
<td>List of raw data. If rawdata=TRUE, a list including raw data components used for calculating estimate.</td>
</tr>
</tbody>
</table>
dunit.multest  Data frame. Table comparing different estimation strategies for SAE.
Raw data
domdat  Data frame. Domain-level data used for estimation.
dunit.totest  String. Table of estimates, including more details.

Note

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. For model-based estimation, we calculate adjustment factors by plot.

It is calculated by dividing 1 / summed condition proportions by plot. An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>

If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor.
If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

Author(s)
Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

References

Examples

# Set up population dataset (see ?modSApop() for more information)
SApopdat <- modSApop(popTabs = list(tree = FIESTA::WYtree,
cond = FIESTA::WYcond),
pltassgn = FIESTA::WYpltassgn,
pltassgnid = "CN",
dunitarea = FIESTA::WYunitarea,
dunitvar = "ESTN_UNIT",
dunitzonal = FIESTA::WYunitzonal,
prednames = c("dem", "tcc", "tpi", "tnt"),
modSApop

predfac = "tnt")

# Fit a unit level EBLUP with `JoSAE`
modSAarea(SApopdatlst = SApopdat,
           SApackage = "JoSAE",
           SAmethod = "unit")

# Fit an area level Fay-Herriot hierarchical Bayesian model with `hbsae`
modSAarea(SApopdatlst = SApopdat,
           SApackage = "hbsae",
           SAmethod = "area")

# Fit an area level Fay-Herriot EBLUP with `sae`, while using Elastic Net
# variable selection
modSAarea(SApopdatlst = SApopdat,
           SApackage = "sae",
           SAmethod = "area",
           modelselect = TRUE)

modSApop

Small area module - Compile population data for SA module.

Description

Compile population data for input to the modSA* modules.

Usage

modSApop(
  popType = "VOL",
  popTabs = popTables(),
  popTabIDs = popTableIDs(),
  popFilter = popFilters(),
  pltassgn = NULL,
  pltassgnid = "PLT_CN",
  dsn = NULL,
  pjoinid = "CN",
  areawt = "CONDPROP_UNADJ",
  adjplot = TRUE,
  dunitvar = NULL,
  dunitarea = NULL,
  areavar = "ACRES",
  dunitzonal = NULL,
  prednames = NULL,
  predfac = NULL,
  savedata = FALSE,
  saveobj = FALSE,
objnm = "SApopdat",
unit_opts = NULL,
savedata_opts = NULL,
SAdoms = NULL,
smallbnd = NULL,
smallbnd.domain = NULL,
SAdata = NULL,
pltdat = NULL,
auxdat = NULL,
gui = FALSE,
...
)

Arguments

popType String. Type of evaluation(s) to include in population data. Note: currently only
\{’CURR’, ’VOL’, ’LULC’\} are available. See details below for descriptions of each.

popTabs List of population tables the user would like returned. See help(popTables) for
a list of options.

popTabIDs List of unique IDs corresponding to the population tables that the user has re-
quested. See help(popTableIDs) for a list of options.

popFilter List of population filters. See help(popFilters) for a list of options.

pltassgn DF/DT, comma-separated values (CSV) file(*.csv), or layer in dsn. Can also be
a shapefile(*.shp) with one record per plot, a spatial layer in dsn, or a sf R object.
Plot-level assignment of estimation unit and/or strata. Optional.

pltassgnid String. Unique identifier of plot in pltassgn.

dsn String. Name of database where tree, cond, and pltassgn tables reside. The dsn
varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).

pjoinid String. Join variable in plot to match pltassgnid. Does not need to be uniqueid.
If using most current XY coordinates for plot assignments, use identifier for plot
(e.g., PLOT_ID).

areawt String. Name of variable for summarizing area weights (e.g., CONDPROP_UNADJ).

adjplot Logical. If TRUE, adjusts for nonresponse at plot-level.

dunitvar String. Name of the domain unit variable in cond, plt, or pltassgn with domain
unit assignment for each plot.

dunitarea Numeric or DF. Total area by domain unit.

areavar String. Name of area variable in unitarea. Default="ACRES".

dunitzonal DF/DT. Data frame with zonal auxiliary information by domain unit. For con-
tinuous data, means by domain unit; for categorical data, proportion of class by
domain unit.

prednames String vector. Name(s) of predictor variables to use in model.

predfac String vector. Name(s) of factor predictor variables to use in model. Names will
change in output depending on number of categories.
**modSApop**

- **savedata**: Logical. If TRUE, saves table(s) to outfolder.
- **saveobj**: Logical. If TRUE, saves returned list object to outfolder.
- **objnm**: String. Name of *.rds object.
- **unit_opts**: List. See help(unit_options()) for a list of options.
- **savedata_opts**: List. See help(savedata_options()) for a list of options.
- **SAdoms**: sf object. SA domains with attributes for joining.
- **smallbnd**: sf object. Small bound.
- **smallbnd.domain**: String. Name of attribute defining domain attribute.
- **SAdata**: R List object. Output data list components from FIESTA::SAdata().
- **pltdat**: R List object. Output data list components from FIESTA::spGetPlots().
- **auxdat**: R List object. Output data list components from FIESTA::spGetAuxiliary().
- **gui**: Logical. If gui, user is prompted for parameters.
- **...**: For extendibility.

**Details**

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (e.g. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>TPA_UNADJ</td>
<td>Number of trees per acre each sample tree represents (e.g. DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on microplot).</td>
</tr>
<tr>
<td>cond</td>
<td>cuniqueid</td>
<td>Unique identifier for each plot, to link to pltassgn (e.g. PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>CONDPROP_UNADJ</td>
<td>Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.</td>
</tr>
<tr>
<td></td>
<td>COND_STATUS_CD</td>
<td>Status of each forested condition on plot (i.e. accessible forest, nonforest, water, etc.)</td>
</tr>
<tr>
<td></td>
<td>NF_COND_STATUS_CD</td>
<td>If ACI=TRUE. Status of each nonforest condition on plot (i.e. accessible nonforest, non-sampled nonforest)</td>
</tr>
<tr>
<td></td>
<td>SITECLCD</td>
<td>If landarea=TIMBERLAND. Measure of site productivity.</td>
</tr>
<tr>
<td></td>
<td>RESERVCD</td>
<td>If landarea=TIMBERLAND. Reserved status.</td>
</tr>
<tr>
<td></td>
<td>SUBPROP_UNADJ</td>
<td>Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.</td>
</tr>
<tr>
<td></td>
<td>MICRPROP_UNADJ</td>
<td>If microplot tree attributes. Unadjusted proportion of microplot conditions on each plot.</td>
</tr>
<tr>
<td></td>
<td>MACRPROP_UNADJ</td>
<td>If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot.</td>
</tr>
</tbody>
</table>

| pltassgn | puniqueid        | Unique identifier for each plot, to link to cond (e.g. CN). |
|          | STATECD          | Identifies state each plot is located in.                           |
|          | INVYR            | Identifies inventory year of each plot.                             |
|          | PLOT_STATUS_CD   | Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed sampled. |

For available reference tables: `sort(unique(FIESTAutils::ref_codes$VARIABLE))`
Value

A list with population data for Small-Area estimates.

SAdomsdf  Data frame. Attribute table from SAdoms spatial layer. Includes DOMAIN and AOI attributes. DOMAIN represents modeling domains. AOI identifies the small area of interest.

condx  Data frame. Condition-level data with condition proportions, domain and predictor assignments, and adjusted condition proportions, if adjplot = TRUE.

pltcondx  Data frame. Plot/Condition data used for estimation.

cuniqueid  String. Unique identifier of plot in condx and pltcondx.

condid  String. Unique identifier of condition in condx and pltcondx.

treex  Data frame. If esttype='TREE', tree-level data, including adjustment factors, if adjplot = TRUE.

tuniqueid  String. If esttype='TREE', unique identifier of plot in treex.

ACI.filter  String. If ACI=FALSE, ACI.filter="COND_STATUS_CD == 1".

dunitarea  Data frame. Area by model domain unit.

areavar  String. Name of area variable in dunitarea.

dunitvar  String. Name of variable defining model domain units in dunitarea.

dunitlut  Data frame. Table of model domain units with zonal statistics of predictor values, number of plots by domain unit.

prednames  String vector. Name of variables in dunitlut and condx defining potential predictors for small area estimation.

plotsampcnt  Data frame. Number of plots by PLOT_STATUS_CD.

condsampcnt  Data frame. Number of conditions by COND_STATUS_CD.

states  String. State names in dataset.

invyrs  String. Range of inventory years in dataset.

adjtree  Logical. If TRUE, treex includes adjustment factors.

Note

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. For model-based estimation, we calculate adjustment factors by plot.

It is calculated by dividing 1 / summed condition proportions by plot. An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:

<table>
<thead>
<tr>
<th>PLOT SIZE</th>
<th>TPA_UNADJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBPLOT</td>
<td>6.018046</td>
</tr>
<tr>
<td>MICROPLOT</td>
<td>74.965282</td>
</tr>
<tr>
<td>MACROPLOT</td>
<td>0.999188</td>
</tr>
</tbody>
</table>
If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor. If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

Author(s)

Tracey S. Frescino, Paul L. Patterson

Examples

# NOTE: FIA data objects used in these examples are stored in `FIESTA`\'s, but can be generated for populations of interest by the user with functions in `FIESTA` such as `spGetPlots()`, `spGetAuxiliary()`, etc. For more information, see `FIESTA`'s extensive vignettes.

# Population data for counties in Wyoming
modSApop(popTabs = list(tree = FIESTA::WYtree, cond = FIESTA::WYcond), 
pltassgn = FIESTA::WYpltassgn, 
pltassgnid = "CN", 
dunitarea = FIESTA::WYunitarea, 
dunitvar = "ESTN_UNIT", 
dunitzonal = FIESTA::WYunitzonal, 
prednames = c("dem", "tcc", "tpi", "tnt"), 
predfac = "tnt")

# Adding seedling data as well
modSApop(popTabs = list(tree = FIESTA::WYtree, 
cond = FIESTA::WYcond, 
seed = FIESTA::WYseed), 
pltassgn = FIESTA::WYpltassgn, 
pltassgnid = "CN", 
dunitarea = FIESTA::WYunitarea, 
dunitvar = "ESTN_UNIT", 
dunitzonal = FIESTA::WYunitzonal, 
prednames = c("dem", "tcc", "tpi", "tnt"), 
predfac = "tnt")

modSAtree

Small area module - Generate small area tree estimates.

Description

Generates small area estimates by domain and/or tree domain (and estimation unit).
modSAtree

Usage

modSAtree(
  SApopdatlst = NULL,
  prednames = NULL,
  SApackage = "JoSAE",
  SAmethod = "area",
  estseed = "none",
  largebnd.unique = NULL,
  landarea = "FOREST",
  pcfilter = NULL,
  estvar = NULL,
  estvar.filter = NULL,
  rowvar = NULL,
  modelselect = FALSE,
  prior = function(x) 1/(sqrt(x) * (1 + x)),
  na.fill = "NONE",
  savedata = FALSE,
  savesteps = FALSE,
  multest = FALSE,
  addSAdomsdf = TRUE,
  SAdomvars = NULL,
  savemultest = FALSE,
  returntitle = FALSE,
  table_opts = NULL,
  title_opts = NULL,
  savedata_opts = NULL,
  multest_opts = NULL,
  save4testing = FALSE,
  gui = FALSE,
  ...
)

Arguments

SApopdatlst List. List of population data objects returned from modSApop().
nprednames String vector. Name(s) of predictor variables to use in model.
SApackage String. Small area package to use (‘JoSAE’, ‘sae’, ‘hbsae’)
SAmethod String. Small area method to use (‘unit’, ‘area’)
estseed String. Use seedling data only or add to tree data. Seedling estimates are only for counts (estvar='TPA_UNADJ')-(‘none’, ‘only’, ‘add’).
largebnd.unique String. Name of the large boundary unique identifier to define plots within a model extent. If NULL, all plots are used for model extent.
landarea String. The sample area filter for estimates (‘ALL’, ‘FOREST’, ‘TIMBERLAND’). If landarea=FOREST, filtered to COND_STATUS_CD = 1; If landarea=TIMBERLAND, filtered to SITECLCD in(1:6) and RESERVCD = 0.
**modSAtree**

- **pcf\_filter**: String. A filter for plot or cond attributes (including pltassgn). Must be R logical syntax.
- **est\_var**: String. Name of the tree estimate variable.
- **est\_var.\_filter**: String. A tree filter for estimate variable. Must be R syntax (e.g., "STATUSCD == 1").
- **row\_var**: String. Name of the row domain variable in cond or tree. If only one domain, row\_var = domain variable. If more than one domain, include col\_var. If no domain, row\_var = NULL.
- **modelselect**: Logical. If TRUE, selects useful predictors using mase: ElasticNet.
- **prior**: Function. A prior function to use for hbsae models.
- **na.\_fill**: String. An estimate to fill in for NA values (i.e., when model is unstable or no predictors are selected). Choose from the following list that does not include SApackage used (‘NONE’, ‘DIR’, ‘JoSAE’, ‘sae’, ‘hbsae’). DIR is suggested value for no NA values.
- **saved\_data**: Logical. If TRUE, saves table(s) to outfolder.
- **savesteps**: Logical. Saves graphs of predictors and response with labels whether selected or not for both area- and unit-level models.
- **multest**: Logical. If TRUE, returns a data frame of SA estimates using both unit-level and area-level estimates.
- **addSA\_doms\_df**: Logical. If TRUE, appends SADom\_df to unit.multest table for output.
- **SA\_dom\_vars**: String vector. List of attributes from SAdoms to include in multest output.
- **savemultest**: Logical. If TRUE, save table with area- and unit-level estimates.
- **return\_title**: Logical. If TRUE, returns title(s) of the estimation table(s).
- **table\_opts**: List. See help(table_options()) for a list of options.
- **title\_opts**: List. See help(title_options()) for a list of options.
- **savedata\_opts**: List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.
- **multest\_opts**: List. See help(multest_options()) for a list of options. Only used when multest = TRUE.
- **save\_4\_testing**: Logical. If TRUE, saves intermediate steps as R objects to outfolder for testing (pdom\_dat, dunit\_lut).
- **gui**: Logical. If gui, user is prompted for parameters.
- **...**: Parameters for modSApop() if SApop\_dat is NULL.

**Details**

If variables are NULL, then it will prompt user to input variables.

Necessary variables:

<table>
<thead>
<tr>
<th>Data</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>tvuniqueid</td>
<td>Unique identifier for each plot, to link to pltstrat (e.g., PLT_CN).</td>
</tr>
<tr>
<td></td>
<td>CONDID</td>
<td>Unique identifier of each condition on plot, to link to cond. Set CONDID=1, if on...</td>
</tr>
</tbody>
</table>
TPA_UNADJ: Number of trees per acre each sample tree represents (e.g. DESIGNCD=1: TPA_UNADJ=6.018046 for trees on subplot; 74.965282 for trees on microplot).

cond: Unique identifier for each plot, to link to pltstrat (ex. PLT_CN).

CONDID: Unique identifier of each condition on plot. Set CONDID=1, if only 1 condition per plot.

CONDPROP_UNADJ: Unadjusted proportion of condition on each plot. Set CONDPROP_UNADJ=1, if only 1 condition per plot.

COND_STATUS_CD: Status of each forested condition on plot (i.e. accessible forest, non-forest, water, etc.).

NF_COND_STATUS_CD: If ACI=TRUE. Status of each non-forest condition on plot (i.e. accessible non-forest).

SITECLCD: If landarea=TIMBERLAND. Measure of site productivity.

RESERVCD: If landarea=TIMBERLAND. Reserved status.

SUBPROP_UNADJ: Unadjusted proportion of subplot conditions on each plot. Set SUBPROP_UNADJ=1, if only 1 condition per subplot.

MICRPROP_UNADJ: If microplot tree attributes. Unadjusted proportion of microplot conditions on each plot.

MACRPROP_UNADJ: If macroplot tree attributes. Unadjusted proportion of macroplot conditions on each plot.

pltassign: Unique identifier for each plot, to link to cond (ex. CN).

STATECD: Identifies state each plot is located in.

INVYR: Identifies inventory year of each plot.

PLOT_STATUS_CD: Status of each plot (i.e. sampled, nonsampled). If not included, all plots are assumed sampled.

Reference names are available for the following variables:
ADFORCD, AGENTCD, CCLCD, DECAYCD, DISTRBCD, KINDCD, OWNCD, OWNGRPCD, FORTYPCD, FLDTYPCD, FORTYPCDCALC, TYPGRPCD, FORINDCD, RESERVCD, LANDCLCD, STDSSZCD, FLDSZCD, PHYSCLCD, MIST_CL_CD, PLOT_STATUS_CD, STATECD, TREECLCD, TRTCD, SPCD, SPGRPCD

Value

est: Data frame. Tree estimates and percent sampling error by domain. Estimates are based on the SApakage and SAmethod parameters defined.
titlelst: List. List of titles used for table output.
raw: List of raw data. If rawdata=TRUE, a list including raw data components used for calculating estimate.
dunit.multest: Data frame. Table comparing different estimation strategies for SAE.

Raw data

domdat: Data frame. Domain-level data used for estimation.
estvar: String. Name of estimation variable.
estvar.filter: String. Logical filter specified for tree data.
dunit.totest: String. Table of estimates, including more details.

Note

ADJUSTMENT FACTOR:
The adjustment factor is necessary to account for nonsampled conditions. For model-based estimation, we calculate adjustment factors by plot.

It is calculated by dividing 1 / summed condition proportions by plot. An adjustment factor is determined for each tree based on the size of the plot it was measured on. This is identified using TPA_UNADJ as follows:
If ACI=FALSE, only nonsampled forest conditions are accounted for in the adjustment factor. If ACI=TRUE, the nonsampled nonforest conditions are removed as well and accounted for in adjustment factor. This is if you are interested in estimates for all lands or nonforest lands in the All-Condition-Inventory.

Common tree filters for estvar.filter:

<table>
<thead>
<tr>
<th>FILTER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;STATUSCD == 1&quot;</td>
<td>Live trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2&quot;</td>
<td>Dead trees</td>
</tr>
<tr>
<td>&quot;TPAMORT_UNADJ &gt; 0&quot;</td>
<td>Mortality trees</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; DIA &gt;= 5.0&quot;</td>
<td>Dead trees &gt;= 5.0 inches diameter</td>
</tr>
<tr>
<td>&quot;STATUSCD == 2 &amp; AGENTCD == 30&quot;</td>
<td>Dead trees from fire</td>
</tr>
</tbody>
</table>

Author(s)

Tracey S. Frescino, Paul L. Patterson, Elizabeth A. Freeman

References


Examples

```r
# Set up population dataset (see ?modSApop() for more information)
SApopdat <- modSApop(popTabs = list(tree = FIESTA::WYtree, cond = FIESTA::WYcond),
                     pltassgn = FIESTA::WYpltassgn, pltassgnid = "CN",
                     dunitarea = FIESTA::WYunitarea, dunitvar = "ESTN_UNIT",
                     dunitzonal = FIESTA::WYunitzonal, prenames = c("dem", "tcc", "tpi", "tnt"),
                     predfac = "tnt")

# Use an area level Fay-Herriot model to estimate total net cubic-foot volume
# of live trees (at least 5 inches diameter)
modSAtree(SApopdatlst = SApopdat,
          predfac = "tnt")
```
popFilters

*Population data filters.*

**Description**

Returns a list of user-supplied parameters and parameter values for population data filters.

**Usage**

```r
popFilters(
  evalid = NULL,
  invyrs = NULL,
  intensity = NULL,
  measCur = FALSE,
  measEndyr = NULL,
  measEndyr.filter = NULL,
  ACI = FALSE,
  ...
)
```

**Arguments**

- `evalid` Numeric. FIA Evaluation identifier for subsetting plots for population.
- `invyrs` Integer vector. Inventory year(s) (e.g., c(2000, 2001, 2002)).
- `intensity` Integer code. Code(s) indicating intensity to use for population.
- `measCur` Logical. Inventory span defining variable. If TRUE, extract plots with most current measurement for state(s).
- `measEndyr` Logical. Inventory span defining variable. If TRUE, extract plots with most current measurement for state(s) for years measured in or before measEndyr.
- `measEndyr.filter` Filter for extracting plots using measEndyr. Must be in R syntax (e.g., 'AOI == 1').

# Use a unit level EBLUP to estimate basal area of live trees (at least 5 # inches diameter)

```r
modSAtree(SApopdatlst = SApopdat,
  SApackage = "JoSAE",
  SAmethod = "unit",
  landarea = "FOREST",
  estvar = "BA",
  estvar.filter = "STATUSCD == 1")
```
**popTableIDs**

ACI Logical. If TRUE, including All Condition Inventory (ACI) plots.

... For extendibility.

**Details**

If no parameters, an empty list is returned.

**Value**

A list of user-supplied parameters and parameter values for population data filters.

**Author(s)**

Grayson W. White

**Examples**

```r
popFilters(ACI = TRUE)
```

**Description**

Returns a list of user-supplied parameters and parameter values for data table unique IDs to be supplied to *pop functions.

**Usage**

```r
popTableIDs(
  cond = "PLT_CN",
  plt = "CN",
  tree = "PLT_CN",
  seed = "PLT_CN",
  vsubpssp = "PLT_CN",
  vsuspstr = "PLT_CN",
  subplot = "PLT_CN",
  subp_cond = "PLT_CN",
  lulc = "PLT_CN",
  cond_dwm_calc = "PLT_CN",
  ...
)
```
Arguments

cond String. Unique identifier of plot in cond.
plt String. Unique identifier of plot in plt.
tree String. Unique identifier of plot in tree and seed.
seed String.
vsubpspp String.
vsubpstr String.
subplot String.
subp_cond String.
lulc String.
cond_dwm_calc String.
... For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Grayson W. White

Examples

popTableIDs(cond = "my_unique_id", tree = "my_tree_id")

Description

Returns a list of user-supplied parameters and parameter values for data tables to be supplied to *pop functions.
popTables

Usage

```
popTables(
    cond = NULL,
    plt = NULL,
    tree = NULL,
    seed = NULL,
    vsubpspp = NULL,
    vsubpstr = NULL,
    invsubp = NULL,
    subplot = NULL,
    subp_cond = NULL,
    lulc = NULL,
    cond_dwm_calc = NULL,
    ...
)
```

Arguments

- **cond**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Condition-level data with one record for each condition, including or excluding nonsampled conditions. Plot variables and strata/estimation unit variable(s) may be included if plt and pltassgn=NULL. See details for necessary variables to include.
- **plt**: DF/DT, Optional. R object, sf R object, comma-delimited file(*.csv), layer or spatial layer in dsn, or shapefile(*.shp). Plot-level data with one record for each plot, including or excluding nonsampled conditions. If nonsampled plots are included, PLOT_STATUS_CD variable must be in table or a filter defined in plt.nonsamp.filter.
- **tree**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level data with one record for each tree. Tree data are aggregated to condition-level. See details for necessary variables to include.
- **seed**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Seedling data with one record for each seedling count.
- **vsubpspp**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Vegetation species-level data with one record for each species (P2VEG_SUBPLOT_SPP).
- **vsubpstr**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Vegetation species-structure data with one record for each species (P2VEG_SUBP_STRUCTURE).
- **invsubp**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Invasive species data with one record for each species (INVASIVE_SUBPLOT_SPP).
- **subplot**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Subplot-level data with one record for each species (SUBPLOT).
- **subp_cond**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Subplot condition-level data with one record for each species (SUBP_COND).
- **lulc**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Land use/Land cover data with current and previous observations.
- **cond_dwm_calc**: DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Calculated down woody material.
For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Grayson W. White

Examples

popTables(cond = FIESTA::WYcond, seed = FIESTA::WYseed)

---

Reference table - for Automated Land Program National Forests.

Description

National Forest region and names from the USDA S_USA.BasicOwnershipFS (Surface Ownership Parcels) attribute table. An area depicted as surface ownership parcels dissolved on the same ownership classification.

Format

A dataframe with 2 columns, R1 and FORESTNAME.

Note

```r
## ALP National Forest boundary
alp_dsn <- "S_USA.BasicOwnershipFS.gdb"
alp_layer <- "BasicOwnershipFS"
alp_att <- "FORESTNAME"
alp_bnd <- spImportSpatial(dsn=alp_dsn, layer=alp_layer)
alp_names <- unique(sf::st_drop_geometry(alp_bnd[, c("REGION", "FORESTNAME")]))
alp_names <- alp_names[order(alp_names$REGION, alp_names$FORESTNAME), ]
alp_names <- alp_names[grepl("National Forest", alp_names$FORESTNAME)==TRUE, ]
alp_names$FORESTNAME <- sub(" National Forest", ", alp_names$FORESTNAME)

Ownership classes that are not USDA Forest Service were removed from the layer.
OWNERCLASSIFICATION = "USDA FOREST SERVICE" subset with following query
alpfs <- alpfs[!is.na(alpfs$OWNERCLASSIFICATION) & alpfs$OWNERCLASSIFICATION != 'NON-FS', ]
```

Source

Downloaded from the USDA Forest Service FSGeodata Clearinghouse on 2020 June 3, format ESRI geodatabase (https://data.fs.usda.gov/geodata/edw/datasets.php?xmlKeyword=surface+ownership)
**ref_codes**

Reference tables - Code definitions.

**Description**

Reference tables - Code definitions.

**Usage**

`ref_codes`

**Format**

An object of class `data.frame` with 2008 rows and 7 columns.

**Source**

FIA look-up tables.

**References**


---

**ref_cond**

Reference table - Metadata for cond default variables output from `DBgetPlots()`.

**Description**

Data frame with variable names and descriptions.

**Format**

A data frame with 61 rows and 3 columns VARIABLE - Variable in cond data frame DESCRIPTION - Description of variable in cond data frame TABLE - Table in database where variable originates or if derived.

**Source**

FIA look-up table.
References


---

### ref_conversion

Reference table - for generating tables.

**Description**

Table conversion factors from English to metric units.

**Format**

A dataframe with 4 columns: TYPE, ENGLISH, METRIC, CONVERSION.

**Source**

Conversion table.

---

### ref_diacl2in

Reference table - diameter 2-inch class codes (DIA).

**Description**

Reference table - diameter 2-inch class codes (DIA).

**Usage**

ref_diacl2in

**Format**

An object of class `data.frame` with 40 rows and 3 columns.

**Source**

Imported from comma-delimited file.

**References**

**ref_domain**

Reference table - for generating tables.

**Description**

Reference table - for generating tables.

**Usage**

`ref_domain`

**Format**

An object of class `data.frame` with 32 rows and 3 columns.

**Source**

FIA look-up table.

---

**ref_estvar**

Reference table - for generating estimates

**Description**

Reference table - for generating estimates

**Usage**

`ref_estvar`

**Format**

An object of class `data.frame` with 30 rows and 9 columns.
ref_plt

Reference table - Metadata for plt default variables output from DBgetPlots()

Description

Data frame with variable names and descriptions.

Format

A data frame with 43 rows and 3 columns

- VARIABLE - Variable in plt data frame
- DESCRIPTION - Description of variable in plt data frame
- TABLE - Table in database where variable originates or if derived

Source

FIA look-up table

References


ref_shp

Reference table - Metadata for shp_* default variables output from DBgetPlots()

Description

Data frame with variable names and descriptions

Format

A dataframe with 63 rows and 4 columns

- VARIABLE - Variable in plt data frame
- DESCRIPTION - Description of variable in plt data frame
- TABLE - Table in database where variable originates or if derived
- SHPEXPORT - Name of variable for exported shapefile (<= 10 characters)

Source

FIA look-up table
References


ref_species

Reference tables - Code definitions.

Description

Table with species information.

Format

A dataframe with 8 columns: SPCD, COMMON_NAME, GENUS, SPECIES, EXISTS_IN_NRCS, EXISTS_IN_NERS, EXISTS_IN_PNWRS, EXISTS_IN_RMRS, EXISTS_IN_SRS.

Source

FIA look-up tables.

References


ref_statecd

Reference table - state codes (STATECD).

Description

Reference table - state codes (STATECD).

Usage

ref_statecd

Format

An object of class dataframe with 59 rows and 5 columns.
Source
Imported from comma-delimited file.

References

ref_titles
Reference table - Variable titles.

Description
Table with variable name (VARNM) and associated title (TITLE).

Format
A dataframe with 2 columns, VARNM and TITLE.

Source
Comma-delimited file.

ref_tree
Reference table - Metadata for tree default variables output from DBgetPlots()

Description
Data frame with variable names and descriptions

Format
A data frame with 72 rows and 3 columns VARIABLE - Variable in tree data frame DESCRIPTION - Description of variable in tree data frame TABLE - Table in database where variable originates

Source
FIA look-up table

References
spClipPoint

Spatial - Clip (intersect) point vector layer with polygon vector layer.

Description

Wrapper for sf::st_intersection, to clip (intersect) point vector layer with a polygon vector layer.

Usage

spClipPoint(
  xyplt,
  xyplt_dsn = NULL,
  uniqueid = "PLT_CN",
  clippolyv,
  clippolyv_dsn = NULL,
  clippolyv.filter = NULL,
  showext = FALSE,
  keepNA = FALSE,
  returnsp = TRUE,
  otherabnms = NULL,
  stopifnotin = TRUE,
  savedata = FALSE,
  exportsp = FALSE,
  spMakeSpatial_opts = NULL,
  savedata_opts = NULL
)

Arguments

xyplt  sf R object or String. Point data to clip. Can be a spatial points object, full path name to a shapefile, or name of a layer within a database.
xyplt_dsn  String. Data source name (dsn; e.g., sqlite or shapefile path name) of layer to clip. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).
uniqueid  String.* Unique identifier of xyplt rows.
clippolyv  sf R object or String. Name of clipping polygon spatial polygon object, full path to shapefile, or name of a layer within a database.
clippolyv_dsn  String. Data source name (dsn; e.g., sqlite or shapefile path name) of clipping polygon. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).
clippolyv.filter  String. Filter to subset clippolyv spatial layer.
showext  Logical. If TRUE, layer extents are displayed in plot window.
keepNA  Logical. If TRUE, keep NA values after data intersection.
returnsp  Logical. If TRUE, returns sf object of points. If FALSE, returns data frame of points (i.e., drops sf geometry).
otherabms String vector. Name(s) of R objects, comma-delimited files, or database layers to subset. Must include quotes (e.g., otherabms=c("tree", "cond")).

stopifnotin Logical. If TRUE, stops if boundaries do not overlap. If FALSE, returns NULL.

savedata Logical. If TRUE, save data to outfolder.

exportsp Logical. If TRUE, the clipped spatial point data are exported.

spMakeSpatial_opts List. See help(spMakeSpatial_options()) for a list of options. Use to convert X/Y values to simple feature (sf) coordinates.

savedata_opts List. See help(savedata_options()) for a list of options for saving data. If out_layer = NULL, default = 'pntclip'.

Details

The sf::st_intersection function is used to clip points.

If the projection of clippolyv is not the same as the xyplt, the xyplt layer layer will be reprojected to the same projection as the clippoly before intersection.

Value

A list of the following objects:

clip_xyplt sf object. The input xyplt, clipped to polygon boundary layer. The projection will be same as clippolyv projection.

xy.uniqueid String. Unique identifier of clip_xy.

clip_polyv SpatialPolygonsDataFrame. The polygon boundary layer used for clipping.

clip_tabs Data frame(s). Other tables in intabs clipped to boundary.

If exportsp=TRUE, the sf object will be written to out_dsn (See note).

Note

On-the-fly projection conversion

The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

ESRI Shapefile Driver
If exportsp=TRUE:
The writeOGR (rgdal) function is called. If out_fmt="shp", the ESRI Shapefile driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If sf object has more than 1 record, it will be returned but not exported.
spClipPoly

Author(s)

Tracey S. Frescino

Examples

# Get point data from WYplt data in FIESTA
WYplt <- FIESTA::WYplt

# Get polygon vector layer from FIESTA external data
WYbhdistfn <- system.file("extdata", 
    "sp_data/WYbighorn_districtbnd.shp", 
    package = "FIESTA")

# Extract points from polygon vector layer
xyext <- spClipPoint(xyplt = WYplt,
    clippolyv = WYbhdistfn,
    clippolyv.filter = "DISTRICTNU == '03'",
    uniqueid = "CN",
    spMakeSpatial_opts = list(xvar = "LON_PUBLIC",
        yvar = "LAT_PUBLIC",
        xy.crs = 4269))

names(xyext)
xyplt <- xyext$clip_xyplt
polyv <- xyext$clip_polyv

# Plot extracted values of national forest district
plot(sf::st_geometry(polyv))
plot(sf::st_geometry(xyplt), add = TRUE)

spClipPoly

Spatial - Clip (intersect) polygon vector layer with polygon vector layer.

Description

Wrapper for sf::st_intersection, to clip (intersect) polygon vector layer with another polygon vector layer.

Usage

spClipPoly(
    polyv,
    polyv_dsn = NULL,
    clippolyv,
    clippolyv_dsn = NULL,
    clippolyv.filter = NULL,
    showext = FALSE,
    areacalc = FALSE,
    areaunits = "ACRES",
)
Arguments

polyv sf R object or String. Polygon data to clip. Can be a spatial polygon object, full pathname to a shapefile, or name of a layer within a database.

colyv_dsn String. Data source name (dsn; e.g., sqlote or shapefile pathname) of layer to clip. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr Formats.html). Optional if polyv is sf object.

clippolyv SpatialPolygons class R object or String. Name of the polygon spatial layer to use for clipping.

clippolyv_dsn String. Data source name (dsn; i.e., pathname or database name) of clippolyv_layer. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr Formats.html). Optional if clippolyv_layer is an R object.

clippolyv.filter String. Filter to subset clippolyv spatial layer.

showext Logical. If TRUE, layer extents are displayed in plot window.

areaCalc Logical. If TRUE, calculate area of clipped polygons and append to attribute table (See details).

areaUnits String. If TRUE, calculate area of clipped polygons and append to attribute table ("ACRES", "HECTARES", "SQKM"). If NULL, units of polyv.

nolonglat Logical. If TRUE, and both layer’s coordinate system is long/lat, the layers are converted to a projected CRS before clipping.

exportsp Logical. If TRUE, the spatial clipped object is exported to outfolder (see spExportSpatial for details).

savedata_opts List. See help(savedata_options()) for a list of options for saving data. If out_layer = NULL, default = 'polyclip'.

Details

The sf::st_intersection function is used to clip polygons.

areacalc
If areacalc = TRUE and the clipped spatial object is not in a projected coordinate system (i.e., long/lat), the object will be reprojected to the Albers Equal Area projection before area is calculated.

Value

sf object of clipped polygon. If polyv and clippolyv have different projections, the projection of returned object will have the same projection as polyv (See note about on-the-fly projection conversion).

If exportsp=TRUE, the sf object will be written to outfolder (See note).
spClipRast

Spatial - Subsets a raster to a polygon extent or boundary.

Description

Subsets a raster to the extent or masked boundary of a spatial polygon object or shapefile (*.shp), with option to write the new file to the outfolder with specified format (fmt).

Note

On-the-fly projection conversion

The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

ESRI Shapefile Driver

If exportsp=TRUE:

The writeOGR (rgdal) function is called. If out_fmt="shp", the ESRI Shapefile driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If sf object has more than 1 record, it will be returned but not exported.

Author(s)

Tracey S. Frescino

Examples

## Not run:
# Load in USAco data from raster package
USAco <- raster::getData("GADM", country = "USA", level = 2)

# Set up data from FIESTA
WYbh <- spImportSpatial(system.file("extdata",
                                  "sp_data/WYbighorn_adminbnd.shp",
                                  package = "FIESTA"))

# Clip polygon with WY Bighorn object from FIESTA
WYbhco <- spClipPoly(polyv = USAco,
                      clippolyv = WYbh)

## End(Not run)

---

spClipRast

Spatial - Subsets a raster to a polygon extent or boundary.
Usage

spClipRast(
  rast,
  rastfolder = NULL,
  clippolyv,
  clippolyv_dsn = NULL,
  clippolyv.filter = NULL,
  rast.crs = NULL,
  bands = NULL,
  NODATA = NULL,
  buffdist = NULL,
  maskByPolygons = TRUE,
  showext = FALSE,
  fmt = "HFA",
  compress = NULL,
  outfolder = NULL,
  outfn = "rastclip",
  outfn.pre = NULL,
  outfn.date = FALSE,
  overwrite = FALSE
)

Arguments

rast String or Raster. Raster name, including extension. Option to include full path.
rastfolder String. Name of the raster folder. Optional.
clippolyv SpatialPolygons class R object or String. Name of the polygon spatial layer to use for clipping.
clippolyv_dsn String. The data source name (dsn; i.e., pathname or database name) of clippolyv. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if polyv_layer is an R object.
clippolyv.filter String. Filter to subset clippolyv spatial layer.
rast.crs EPSG code or PROJ.4 string. Defined coordinate reference system if rast has no crs defined.
bands Numeric vector. If rast is a multi-layer raster and only 1 or some layers are desired, specify layer number(s) in a vector format. If NULL, all layers are summed.
NODATA Number. The NODATA value for background values. If NODATA is NULL, and a NODATA value is defined on the input raster, the default is the defined NODATA value, else it is defined based on its datatype (see DEFAULT_NODATA for default data values).
buffdist Number. The distance to buffer the polygon before clipping raster, in units of raster.
maskByPolygons Logical. If TRUE, rast is clipped to boundary of polygon. If FALSE, rast is clipped to extent of polygon.
showext Logical. If TRUE, layer extents are displayed in plot window.
fmt String. Format for exported raster. Default is format of unput raster. ("raster", "ascii", "SAGA", "IDRISI", "CDF", "GTiff", "ENVI", "EHdr", "HFA", "VRT"). VRT is a virtual raster (See note below).
compress String. An optional compression type ('LZW', 'DEFLATE', 'PACKBITS').
outfolder String. The output folder.
outfn String. Name of output data file. If NULL, default is 'rastclip'. If no extension, a default is provided to match output format.
outfn.pre String. Add a prefix to output name (e.g., "01").
outfn.date Logical. If TRUE, add date to end of outfile (e.g., outfn_.date.csv).
overwrite Logical. If TRUE, overwrite files in outfolder.

Details
Use spClipRast() to prompt for input.
If the projection of polyv is different than the projection of rast, the polyv SpatialPolygons object is converted to the projection of rast (See note about on-the-fly projection conversion).

Value
value Spatial S4 object. A clipped raster file.
The clipped raster is written to outfolder with specified format or same format as input raster.

Note
On-the-fly projection conversion
The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

VRT format Virtual raster format is a pointer to a temporary file, commonly used as an intermediate step between processes. The VRT format ignores option to maskByPolygons.

Author(s)
Tracey S. Frescino

Examples
# Get polygon vector layer from FIESTA external data
WYbhdistfn <- system.file("extdata",
  "sp_data/Wybighorn_districtbnd.shp",
  package = "FIESTA")
spExportSpatial <- FIESTA::spImportSpatial(WYbhdistfn)
WYbhdist

# Get raster layers from FIESTA external data
demfn <- system.file("extdata",
   "sp_data/WYbighorn_dem_250m.img",
   package = "FIESTA")
raster::plot(raster::raster(demfn))

# Clip raster to district = '03'
dem03 <- spClipRast(rast = demfn,
   clippolyv = WYbhdistfn,
   clippolyv.filter = "DISTRICTNU == '03'",
   overwrite = TRUE,
   outfolder = tempdir())
raster::plot(raster::raster(dem03))

# Clip raster to district = '06'
dem06 <- spClipRast(rast = demfn,
   clippolyv = WYbhdistfn,
   clippolyv.filter = "DISTRICTNU == '06'",
   overwrite = TRUE,
   outfolder = tempdir())

# Plot extracted values of national forest district
raster::plot(raster::raster(dem06))

---

spExportSpatial

**Spatial - Exports an sf object.**

**Description**

Exports an sf object to a specified output.

**Usage**

spExportSpatial(sfobj, savedata_opts = NULL)

**Arguments**

sfobj

sf class R object. Spatial object to export.

savedata_opts

List. See help(savedata_options()) for a list of options for saving data. If out_layer = NULL, default = 'datsp'.

**Details**

Wrapper for sf::st_write function.
Value

An sf spatial object is written to outfolder.

Note

If out_fmt='shp':
The ESRI shapefile driver truncates variable names to 10 characters or less. Variable names are
changed before export using an internal function (trunc10shp). Name changes are output to the
outfolder, 'out_layer'_newnames.csv.

If sf object has more than 1 record, it cannot be exported to a shapefile.

Author(s)

Tracey S. Frescino

Examples

# Set up data from FIESTA
WYbh <- spImportSpatial(system.file("extdata",
    "sp_data/WYbighorn_adminbnd.shp",
    package = "FIESTA"))

# Export data with spExportSpatial
spExportSpatial(WYbh,
    savedata_opts = list(out_dsn = "WYbh.shp",
        outfolder = tempdir(),
        overwrite_dsn = TRUE))

spExtractPoly
Spatial - Extracts point attribute values from SpatialPolygons layer(s).

Description

Extracts values from one or more polygon layers and appends to input SpatialPoints layer or data
frame. Points are reprojected on-the-fly to projection of SpatialPolygons using PROJ.4 transformation
parameters and rgdal spTransform function.

Usage

spExtractPoly(
    xyplt,
    xyplt_dsn = NULL,
    xy.uniqueid = "PLT_CN",
    polyvlist,
    polyv_dsn = NULL,
    polyvarlist = NULL,
    polyvarnmist = NULL,
Arguments

**xyplt** Data frame object or String. Name of layer with xy coordinates and unique identifier. Can be layer with `xy_dsn`, full pathname, including extension, or file name (with extension) in `xy_dsn` folder.

**xyplt_dsn** String. Name of database where xyplt is. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).

**xy.uniqueid** String.* Unique identifier of xyplt rows.

**polyvlst** sf R object or String. Name(s) of polygon layers to extract values. A spatial polygon object, full path to shapefile, or name of a layer within a database.

**polyv_dsn** String. Data source name (dsn) where polyvlst layers are found (e.g., *.sqlite, *.gdb, folder name). The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).

**polyvarlst** String vector or list. The name(s) of variable(s) to extract from polygon(s). If extracting multiple variables from more than one polygon, specify names in a list format, corresponding to polyvlst.

**polyvarnmlst** String vector or list. Output name(s) of variable(s) extracted from polygon(s). If extracting multiple variables from more than one polygon, specify names in a list format, corresponding to polyvlst. The number of names must match the number of variables in polyvarlst.

**keepNA** Logical. If TRUE, keep NA values.

**showext** Logical. If TRUE, layer extents are displayed in plot window.

**savedata** Logical. If TRUE, the input data with extracted values are saved to outfolder.

**exportsp** Logical. If TRUE, the extracted point data are exported to outfolder.

**exportNA** Logical. If TRUE, NULL values are exported to outfolder.

**spMakeSpatial_opts** List. See help(spMakeSpatial_options()) for a list of options. Use to convert X/Y values to simple feature (sf) coordinates.

**savedata_opts** List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = 'polyext'.

**gui** Logical. If gui, user is prompted for parameters.
Details

*If variable = NULL, then it will prompt user for input.

keepnull
If keepnull=FALSE, points are excluded when all extracted variables from any one SpatialPolygons are NULL, returning the points that fall within the intersecting polygons.

Value

pltdat SpatialPointsDataFrame object or data frame. Input point data with extracted raster values appended. For multi-part polygons, more than 1 row per point may be output.

var.name String vector. Variable names of extracted variables.

If savedata=TRUE, outdat data frame is saved to outfolder (Default name: datext_'date'.csv). If exportsp=TRUE, the SpatialPointsDataFrame object is exported to outfolder (Default name: datext_'date'.shp). Variable names are truncated to 10 characters or less (See note below). Name changes are output to 'outfn'_newnames_'data'.csv in outfolder.

Note

If exportshp=TRUE:
The writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.

The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

Any names in polygon layers that are the same as in xyplt are renamed to name'_1'.
polyvlist = Wbhdistfn,
xy.uniqueid = "CN",
spMakeSpatial_opts = list(xvar = "LON_PUBLIC",
yvar = "LAT_PUBLIC",
xy.crs = 4269))

names(xyext)
xyext$outnames
spxyext <- xyext$spxyext
head(spxyext)
NAlst <- xyext$NAlst

# Plot extracted values of national forest district
plot(spxyext["DISTRICTNU"])

spExtractRast  

---

**spExtractRast**  

**Spatial - Extracts point attribute values from raster layer(s).**

**Description**

Extracts values from one or more raster layers and appends to input SpatialPoints layer or data frame. Points are reprojected on-the-fly to projection of raster(s) using PROJ.4 transformation parameters and rgdal spTransform function. Includes options to use bilinear interpolation or summarize over a window of n pixels using a specified statistic.

**Usage**

```r
spExtractRast(
  xyplt,
  xyplt_dsn = NULL,
  xy.uniqueid = "PLT_CN",
rastlst,
rastfolder = NULL,
rast.crs = NULL,
bandlst = NULL,
var.name = NULL,
interpolate = FALSE,
windowsize = 1,
windowstat = NULL,
rast.NODATA = NULL,
keepNA = TRUE,
ncores = 1,
showext = FALSE,
savedata = FALSE,
exportsp = FALSE,
exportNA = FALSE,
spMakeSpatial_opts = NULL,
savedata_opts = NULL,
gui = FALSE
)
```
Arguments

xyplt  Data frame object or String. Name of layer with xy coordinates and unique identifier. Can be layer with xy_dsn, full pathname, including extension, or file name (with extension) in xy_dsn folder.


xy.uniqueid  String. Unique identifier of xyplt rows.

rastlist  String vector or list or strings and/or rasters. File name(s) with extensions, or raster object(s). Note: raster objects must be written to file.

rastfolder  String. Name of the folder with raster layers. Optional. Useful if all raster layers are in same folder.

rast.crs  EPSG code or PROJ.4 String. Name of coordinate reference system for rasters with no projection defined. If more than one raster has no projection defined, the same crs will be used.

bandlst  Numeric named list. If rastfnlst includes a multi-layer raster and only 1 or some layers are desired, specify layer numbers in a named list format with names matching the base names in rastfnlst (e.g., list(rast1=5, rast3=1:3)). If NULL, all layers are extracted.

var.name  String vector. Extracted variable name(s). If NULL, uses the basename of raster layer, including band number for multi-band rasters.

interpolate  Logical vector. If TRUE, uses bilinear interpolation of pixel values, weighted average of 4 nearest pixels (i.e., continuous data).

windowsize  Number vector. The size of window for summarizing data.

windowstat  Character vector. If windowsize is greater than one, the statistic to use for summarizing data ("mean", "min", "max", "median", "sum", "range", "var", "sd", "rsd", "mode", "value"). If windowstat="value", all pixel values are returned, otherwise 1 value per row in xyplt is returned.

rast.NODATA  Numeric vector. NODATA value(s) of raster if not predefined (See notes below). This value will be converted to NA and removed if keepNA=FALSE. If rastfnlst includes more than one raster, the rast.NODATA value should coincide with number of rasters in rastfnlst. If only one rast.NODATA, the same NODATA value is used for all rasters.

keepNA  Logical. If TRUE, keeps NA values after data extraction.

ncore  Integer. Number of cores to use for extracting values.

showext  Logical. If TRUE, layer extents are displayed in plot window.

savedata  Logical. If TRUE, the input data with extracted values are saved to outfolder.

exportsp  Logical. If TRUE, the extracted raster point data are exported to outfolder.

exportNA  Logical. If TRUE, NA values are exported to outfolder.

spMakeSpatial_opts  List. See help(spMakeSpatial_options()) for a list of options. Use to convert X/Y values to simple feature (sf) coordinates.

savedata_opts  List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE. If out_layer = NULL, default = 'rastext'.

gui  Logical. If gui, user is prompted for parameters.
Details

*If variable = NULL, then it will prompt user for input.

Value

- `sppltex` sf object or data frame. Input xyplt data with extracted raster values appended.
- `outnames` String vector. Raster output names.
- `rastfnlst` String vector. Raster pathnames.
- `inputdf` Data frame. Raster information input to zonal summaries.
- `NAlst` sf List. If NA values exist after data extraction, the spatial NA points are returned.

If `savedata=TRUE`, pltassgn and unitarea are saved to outfolder.
If `exportsp=TRUE`, the spatial sf points object is exported to outfolder.
- If `exportNA=TRUE` and NA values exist after data extraction, the spatial NA points are exported to outfolder.

Note

- `rast.NODATA`
  NODATA values are raster pixel values that have no data of interest, including pixels within the extent of the layer, but outside the area of interest. Sometimes these pixels have been defined previously. The defined NODATA pixels are imported to R as NULL values. When not previously defined, the pixels outside the area of interest will be the minimum or maximum value depending on the data type (e.g., 16-bit signed: min=-32,768; max=32,768) or byte size (1 byte: min=0; max=255). These NODATA values will be added to the zonal statistic calculations if not specified in rast.NODATA.

The `spTransform` (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See `spTransform` help documentation for more details.

Author(s)

Tracey S. Frescino

Examples

```r
# Get point data from WYplt data in FIESTA
WYplt <- FIESTA::WYplt

# Get raster layers from FIESTA external data
fromfn <- system.file("extdata",
                   "sp_data/WYbighorn_forest_nonforest_250m.tif",
                   package = "FIESTA")

demfn <- system.file("extdata",
```

```
"sp_data/WYbighorn_dem_250m.img",
package = "FIESTA")

# Extract points from raster
xyext <- spExtractRast(xyplt = WYplt,
  rastlst = c(fornffn, demfn),
  var.name = c("fornf", "dem"),
  xy.uniqueid = "CN",
  spMakeSpatial_opts = list(xvar = "LON_PUBLIC",
    yvar = "LAT_PUBLIC",
    crs = 4269))

names(xyext)
xyext$outnames
sppltext <- xyext$sppltext
head(sppltext)
xyext$inputdf

# Plot extracted values of forest/nonforest
plot(sppltext["fornf"])

# Plot extracted values of dem (i.e., elevation)
plot(sppltext["dem"])
```

---

```
spGetAuxiliary

Spatial wrapper - Extracts and compiles auxiliary data within a specified boundary.

Description

Wrapper to extract and compile auxiliary data by domain unit (i.e, estimation unit or small area domain). The following information is compiled:
- Attribute defining domain (i.e., estimation unit) from domain layer
- Area by domain (i.e., estimation unit)
- Zonal statistics by domain (i.e., estimation unit) - spZonalRast()

Usage

spGetAuxiliary(
  xyplt,
  xyplt_dsn = NULL,
  uniqueid = "PLT_CN",
  unittype = "POLY",
  unit_layer = NULL,
  unit_dsn = NULL,
  unitvar = NULL,
  rastlst.cont = NULL,
  rastlst.cont.name = NULL,
  rastlst.cont.stat = "mean",
```
Arguments

xyplt: Data frame object or String. Name of layer with xy coordinates and unique identifier. Can be layer with xy_dsn, full pathname, including extension, or file name (with extension) in xy_dsn folder.


uniqueid: String.* Unique identifier of xyplt records.

unittype: String. Type of spatial layer unit_layer is ("POLY", "RASTER").

unit_layer: sf R object or String. Name of the domain spatial layer. Can be a spatial polygon object, full pathname to a shapefile, name of a polygon layer within a database, or a full pathname to raster file.

unit_dsn: String. The data source name (dsn; i.e., folder or database name) of unit_layer. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional.

unitvar: String. Name of domain variable in domlayer. If NULL, assuming one domain. An attribute names ONEUNIT is added to layer with value=1.

rastlst.cont: String vector or list. A list of raster(s) with continuous data values (e.g., DEM).
The list may include file name of raster(s) or raster objects that are not InMemory.

**rastlst.cont.name**
String vector. Output names for continuous rasters. Optional. If NULL, name of raster is used as default or name+'_'+'layer number for multi-band layers.

**rastlst.cont.stat**
String. Zonal statistic for continuous rasters.

**rastlst.cont.NODATA**
Numeric vector. NODATA value for continuous rasters (See notes). These values will be converted to NA and removed from output if keepNA=FALSE. If 1 number, the same value will be used for all categorical rasters. If more than 1 number, the number of values must be equal to the number of rasters in rastlst.cont.

**rastlst.cat**
String vector or list. A list of raster(s) with thematic (i.e., categorical) data values. The list may include file name of raster(s) or raster objects that are not InMemory.

**rastlst.cat.name**
String vector. Output names for categorical rasters. If NULL, name of raster is used as default or name+'_'+'layer number for multi-band layers.

**rastlst.cat.NODATA**
Numeric vector. NODATA value for categorical rasters (See notes). These values will be converted to NA and removed from output if keepNA=FALSE. If 1 number, the same value will be used for all categorical rasters. If more than 1 number, the number of values must be equal to the number of rasters in rastlst.cat.

**rastfolder**
String. Name of the folder with raster layers. Optional. Useful if all raster layers are in same folder.

**asptranform**
Logical. If TRUE, transforms aspect to Northness and Eastness indices using sin and cosine functions.

**rast.asp**
String or raster object. The raster in rastlst.cont that is the aspect raster (Note: aspect must have units in degrees).

**rast.lut**
String. A raster in rastlst.cat to group class values. Only one raster is allowed.

**rastlut**
String or raster object. The raster lookup table used for collapsing rast.lut values.

**extract**
Logical. If TRUE, extracts values from rastlst.cont and rastlst.cat along with values from unit_layer. If FALSE, extracts only values from unit_layer.

**areacalc**
Logical. If TRUE, returns area by domvar.

**areaunits**
String. Output area units ("ACRES", "HECTARES", "SQMETERS").

**keepNA**
Logical. If TRUE, returns data frame of NA values.

**ncores**
Integer. Number of cores to use for extracting values.

**NAt0**
Logical. If TRUE, converts extracted NA values to 0.

**npixels**
Logical. If TRUE, include number of pixels.

**addN**
Logical. If TRUE, adds N to unitzonal output with number of plots by unit.
showext Logical. If TRUE, layer extents are displayed in plot window.
returnxy Logical. If TRUE, returns xy data as sf object (spxyplt).
savedata Logical. If TRUE, the input data with extracted values are saved to outfolder.
exportsp Logical. If savedata=TRUE and returnxy=TRUE, If TRUE, the extracted strata point data are exported to outfolder.
exportNA Logical. If TRUE, NA values are exported to outfolder.
spMakeSpatial_opts List. See help(spMakeSpatial_options()) for a list of options. Use to convert X/Y values to simple feature (sf) coordinates.
savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.
vars2keep String vector. Attributes in SAdoms, other than domvar to include in unitzonal output and extract to pltassgn points.
gui Logical. If gui, user is prompted for parameters.

Details

*If variable = NULL, then it will prompt user for input.
If there is a raster and SpatialPolygon layer, and the projection of the SpatialPolygons is different than the projection of the raster, the SpatialPolygons object is reprojected to the projection of raster (See note about on-the-fly projection conversion).

Value

pltassgn sf object. xyplt data with extracted values from rastlst*.
unitzonal Data frame. Number of pixels and zonal statistics from continuous rasters or zonal proportions from categorical raster for each domain (i.e., estimation unit).
unitvar Data frame. Domain (i.e., estimation unit) name.
inputdf Data frame. Raster information input to zonal summaries.
prednames String vector. Name(s) of predictor variable(s).
zonalnames String vector. Name(s) of zonal variable(s).
predfac String vector. Name(s) of categorical (i.e. factor) variable(s).
npixelvar String. Name of variable describing number of pixels.
unitarea Data frame. Area by domain (i.e., estimation unit).
areavar String. Name of variable describing acres in domarea.
pltassgnid String. Unique identifier of plot.
spxy Simple feature. If returnxy=TRUE, Spatial coordinates.
xy.uniqueid String. If returnxy=TRUE, unique identifier of spxy.

If savedata=TRUE, datstrat and unitarea are saved to outfolder. If exportsp=TRUE, the sf object is exported to outfolder.
Note

rast.NODATA
NODATA values are raster pixel values that have no data of interest, including pixels within the extent of the layer, but outside the area of interest. Sometimes these pixels have been defined previously. The defined NODATA pixels are imported to R as NULL values. When not previously defined, the pixels outside the area of interest will be the minimum or maximum value depending on the data type (e.g., 16-bit signed: min=-32,768; max=32,768) or byte size (1 byte: min=0; max=255). These NODATA values will be added to the zonal statistic calculations if not specified in rast.NODATA.

If exportsp=TRUE:
If out_fmt="shp", the writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.

On-the-fly projection conversion
The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

Author(s)
Tracey S. Frescino

Examples

# Get layers from FIESTA external data
## dem (continuous)
demfn <- system.file("extdata",
    "sp_data/WYbighorn_dem_250m.img",
    package = "FIESTA")

## tnt (categorical)
tntfn <- system.file("extdata",
    "sp_data/WYbighorn_forest_nonforest_250m.tif",
    package = "FIESTA")

## unit layer
WYbhdistfn <- system.file("extdata",
    "sp_data/WYbighorn_districtbnd.shp",
    package = "FIESTA")

# Get Auxiliary data
spGetAuxiliary(xyplt = FIESTA::WYplt,
    uniqueid = "CN",
    unit_layer = WYbhdistfn,
    unitvar = "DISTRICTNA",
    rastlst.cont = demfn,)
spGetEstUnit

Spatial wrapper - Extracts point attribute values and area from a simple feature or raster estimation unit layer.

Description

Wrapper to get point attribute values and area from a simple feature or raster layer of estimation units and calculates area. Points are reprojected on-the-fly to projection of unit_layer using PROJ.4 transformation parameters and rgdal spTransform function. - Point attribute extraction from simple feature (spExtractPoly) or from raster (spExtractRast) - Calculate area by estimation unit(s) (areacalc.poly/areacalc.pixel)

Usage

spGetEstUnit(
  xyplt,
  xyplt_dsn = NULL,
  uniqueid = "PLT_CN",
  unittype = "POLY",
  unit_layer,
  unit_dsn = NULL,
  unitvar = NULL,
  unit.filter = NULL,
  areavar = NULL,
  areaunits = "acres",
  keepNA = FALSE,
  returnxy = FALSE,
  showext = FALSE,
  savedata = FALSE,
  exportsp = FALSE,
  exportNA = FALSE,
  spMakeSpatial_opts = NULL,
  savedata_opts = NULL,
  vars2keep = NULL,
  gui = FALSE
)

Arguments

xyplt  Data frame, sf object, full pathname to *.csv or *shp, or layer name in a geodatabase. Includes XY coordinates and unique identifier. If non-spatial, include options in spMakeSpatial_opts parameter.
uniqueid  String.* Unique identifier of xyplt rows.
unittype  String. Spatial layer type of unit_layer ("POLY", "RASTER").
unit_layer  String or sf object. The name of the estimation unit layer. The layer name may be a full pathname to a file, the basename to a file, a spatial layer name from a database, or a SpatialPolygons object with a defined projection.
unit_dsn  String. The data source name (dsn; i.e., folder or database name) of unit_layer. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional.
unitvar  String. Name of estimation unit variable in unit_layer.
unit.filter  String. Filter to subset unit_layer spatial layer.
areavar  String. Name of area variable unit variable in unit_layer. If NULL, calculates area by unitvar.
areaunits  String. Output area units ("acres", "hectares", "sqmeters").
keepNA  Logical. If TRUE, returns data frame of NA values.
returnxy  Logical. If TRUE, returns xy data as sf object (spxyplt).
showext  Logical. If TRUE, layer extents are displayed in plot window.
savedata  Logical. If TRUE, the input data with extracted values are saved to outfolder.
exportsp  Logical. If TRUE, the extracted strata point data are exported to outfolder.
exportNA  Logical. If TRUE, NA values are exported to outfolder.
spMakeSpatial_opts  List. See help(spMakeSpatial_options()) for a list of options. Use to convert X/Y values to simple feature (sf) coordinates.
savedata_opts  List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.
vars2keep  String vector. Attributes in SAdoms, other than domvar to include in dunitlut output and extract to pltassgn points.
gui  Logical. If gui, user is prompted for parameters.

Details

*If variable = NULL, then it will prompt user for input.

If there is a raster and simple feature layer, and the projection of the simple feature is different than the projection of the raster, the simple feature object is transformed to the projection of raster (See note about on-the-fly projection conversion).

Value

pltunit  Data frame. Input point data with extracted estimation unit and strata values appended.
spplttunit  SpatialPointsDataFrame. Spatial point data with extracted estimation unit values appended.
unitarea  Data frame. Area by estimation unit.
unitvar   Data frame. Variable name for estimation unit in unitarea.
acrevar  Data frame. Variable name for area in unitarea.
pltassgnid String. Unique identifier of plot.

If savedata=TRUE, pltstrat and unitarea are saved to outfolder (Default name: *_`date`.csv). If exportshp=TRUE, the SpatialPointsDataFrame object is exported to outfolder (Default name: data_text_`date`.shp). Variable names are truncated to 10 characters or less (See note below). Name changes are output to `outfn`_newnames_`data`.csv in outfolder.

Note

If exportsp=TRUE:
If out_fmt="shp", the writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.

On-the-fly projection conversion
The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

unitarea
Area by estimation unit is calculated and returned as object named unitarea. Area is based on the projection of unit_layer. If no unit_layer input, than area is calculated from pixel counts.

Author(s)
Tracey S. Frescino, Chris Toney

Examples

```r
# Set up data from FIESTA
WYbhfn <- system.file("extdata", "sp_data/WYbighorn_adminbnd.shp", package = "FIESTA")

# Create a `SpatialPoints` object from `WYplt`
WYspplt <- spMakeSpatialPoints(xyplt = WYplt, xy.uniqueid = "CN", xvar = "LON_PUBLIC", yvar = "LAT_PUBLIC", xy.crs = 4269)

# Get estimation unit acres for Bighorn National Forest
spGetEstUnit(xyplt = WYplt, uniqueid = "CN", unit_layer = WYbhfn,
```

spGetPlots

spMakeSpatial_opts = list(xvar = "LON_PUBLIC",
yvar = "LAT_PUBLIC",
prj = "longlat",
datum = "NAD83"))

spGetPlots  Spatial wrapper - Extracts plot data within a given boundary.

Description

Wrapper to get FIA plots within the boundary population (area of interest) - Intersect with state boundary - Get FIA plots for intersected states, including tree, and spatial - Clip spatial coordinates and other tables to boundary (spClipPoint)

Usage

spGetPlots(
  bnd = NULL,
  bnd_dsn = NULL,
  bnd.filter = NULL,
  states = NULL,
  RS = NULL,
  pltids = NULL,
  xy_datsource = NULL,
  xy = NULL,
  xy_dsn = NULL,
  xy.uniqueid = "PLT_CN",
  xvar = NULL,
  yvar = NULL,
  xy.crs = 4269,
  xyjoinid = NULL,
  pjoinid = NULL,
  clipxy = TRUE,
  datsource = NULL,
  data_dsn = NULL,
  istree = FALSE,
  isseed = FALSE,
  isveg = FALSE,
  isdwm = FALSE,
  plot_layer = "plot",
  cond_layer = "cond",
  tree_layer = "tree",
  seed_layer = "seedling",
  vsubpspp_layer = "p2veg_subplot_spp",
  vsubpstr_layer = "p2veg_subp_structure",
  invsubp_layer = "invasive_subplot_spp",
  dwm_layer = "cond_dwm_calc",
)
spGetPlots

ppsa_layer = "pop_plot_stratum_assgn",
other_layers = NULL,
puniqueid = "CN",
pltassgnid = "PLT_CN",
savePOP = FALSE,
evalid = NULL,
evalCur = FALSE,
evalEndyr = NULL,
evalType = "VOL",
measCur = FALSE,
measEndyr = NULL,
measEndyr_filter = NULL,
invyrs = NULL,
measyrs = NULL,
allyrs = FALSE,
intensity1 = FALSE,
showsteps = FALSE,
returnndata = TRUE,
savedata = FALSE,
savebnd = FALSE,
returnxy = TRUE,
exportsp = FALSE,
savedata_opts = NULL,
spXYdat = NULL,
gui = FALSE
)

Arguments

bnd sf R object. Area of Interest (AOI) boundary. Can be a spatial sf object, full path name to a shapefile, or name of a layer within a database.

bnd_dsn String. Data source name (dsn; e.g., SQLite database or shapefile path name) of bnd. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if bnd is an R object.

bnd.filter String. Filter to subset bnd spatial layer.

states String. The name of state(s) for tables (e.g., "Vermont", "Utah").

RS String. Name of FIA research station to restrict states to ('RMRS', 'SRS', 'NCRS', 'NERS', 'PNWRS'). If NULL, all research stations are included.

pltids Data frame. Non-spatial plot identifiers within bnd).

xy_datsource String. Source of XY data ("obj", "csv", "datamart", "sqlite"). If datsource=NULL, checks extension of xy_dsn or xy to identify datsource.

xy sf R object or String. Table with xy coordinates. Can be a spatial polygon object, data frame, full path name to a shapefile, or name of a layer within a database.

xy_dsn String. Data source name (dsn; i.e., path name or database name) of xy. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if bnd_layer is an R object.
spGetPlots

xy.uniqueid  String. Unique identifier of xy.
xvar  String. Name of variable in xyplt defining x coordinate.
yvar  String. Name of variable in xyplt defining y coordinate.
xy.crs  PROJ.4 String or CRS object or Integer EPSG code defining Coordinate Reference System.
xyjoinid  String. Variable in xy to join to plot data. If NULL, xyjoinid=xy.uniqueid.
pjoinid  String. Variable in plt to join to XY data. Not necessary to be unique. If using most current XY coordinates, use identifier for a plot (e.g., PLOT_ID).
clipxy  Logical. If TRUE, clips xy data to bnd.
data_dsn  String. Name of database where *_layers reside.
istree  Logical. If TRUE, extract tree data from FIA database.
isseed  Logical. If TRUE, extract seedling data from FIA database.
isveg  Logical. If TRUE, understory vegetation tables are extracted from FIA database (P2VEG_SUBPLOT_SPP, P2VEG_SUBP_STRUCTURE, INVASIVE_SUBPLOT_SPP).
isdwm  Logical. If TRUE, down woody material are extracted.
plot_layer  String. Name of layer in database of file name of FIA plot table.
cond_layer  String. Name of layer in database of file name of FIA cond table.
tree_layer  String. Name of layer in database of file name of FIA tree table if istree=TRUE.
seed_layer  String. Name of layer in database of file name of FIA seedling table if isseed=TRUE.
vsubpspp_layer  String. Name of layer in database of file name of FIA P2VEG_SUBPLOT_SPP table if isveg=TRUE.
vsubpstr_layer  String. Name of layer in database of file name of FIA P2VEG_SUBP_STRUCTURE table if isveg=TRUE.
invsubp_layer  String. Name of layer in database of file name of FIA INVASIVE_SUBPLOT_SPP table if isveg=TRUE.
dwm_layer  String. Name of layer in database of file name of FIA COND_DWM_CALC table if isdwm=TRUE.
ppsa_layer  String. Name of layer in database of file name of FIA pop_plot_stratum_assgn table, if using evaluations.
other_layers  String. Other layer(s) in database to clip and/or extract from database (Note: must include PLT_CN variable as unique identifier).
puniqueid  String. Name of unique identifier of plt.
pltassgnid  String. Name of unique identifier of pop_plot_stratum_assgn.
savePOP  Logical. If TRUE, returns and/or saves POP_PLOT_STRATUM_ASSGN table.
### spGetPlots

**evalid**  
Integer. To extract data for a specific evaluation period. See notes for more information about FIA Evaluations.

**evalCur**  
Logical. If TRUE, extract plots with most current FIA Evaluation for state(s).

**evalEndyr**  
Integer. Defining end year of Evaluation (yyyy).

**evalType**  
String vector. The type(s) of evaluation of interest ('ALL', 'AREAVOL', 'GRM', 'P2VEG', 'DWM', 'INV', 'REGEN', 'CRWN'). The evalType 'ALL' includes nonsampled plots; 'AREAVOL' includes plots used for area or tree estimates (eval_typ plots used for growth, removals, mortality, and change estimates (eval_typ database manual for regional availability and/or differences).

**measCur**  
Logical. If TRUE, extract plots with most current measurement for state(s).

**measEndyr**  
Integer year (YYYY). If measCur=TRUE, extract plots with most current measurement for state(s) for years measured before measEndyr.

**measEndyr.filter**  
Filter. If measCur=TRUE and measEndyr != NULL, a filter for bnd to identify and area to use measEndyr, such as disturbed areas where you want to exclude plots measured after disturbance.

**invyrs**  
Integer vector. Defining specific inventory years of data (e.g., 2010:2015).

**measyrs**  
Integer vector. Defining specific measurement years of data (e.g., 2010:2015).

**allyrs**  
Logical. If TRUE, selects all years (annual inventory) in database.

**intensity1**  
Logical. If TRUE, include only single intensity plots (i.e., INTENSITY = 1).

**showsteps**  
Logical. If TRUE, display data in device window.

**returndata**  
Logical. If TRUE, returns data objects.

**savedata**  
Logical. If TRUE, saves data to outfolder.

**savebnd**  
Logical. If TRUE, saves bnd. If out_fmt='sqlite', saves to a SpatiaLite database.

**returnxy**  
Logical. If returnxy=TRUE, if TRUE, saves xy data as spatial data. If FALSE, saves xy data as table.

**savedata_opts**  
List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

**spXYdat**  
R list object. Output from spGetXY().

**gui**  
Logical. If TRUE, uses gui interface.

### Details

**datasource**
Plots are extracted from 3 different data sources:
1) CSV - data have previously been extracted from the FIA database and stored as CSV files.
2) datamart - data are extracted from FIA's publically-available datamart.
3) sqlite - data have previously been extracted from the FIA database and stored within a SQLite database.

**Selection parameters**
FIA plots are selected based on the following parameters:
evalCur - the most current FIA evaluation in database

evalEndyr - the FIA evaluation ending in evalEndyr

evalType - the FIA evaluation type ('ALL', 'AREAVOL', 'GRM', 'P2VEG', 'DWM', 'INV', 'REGEN', 'CRWN')

measCur - the most current measurement of each plot in database

measEndyr - the most current measurement of each plot in database in or prior to measEndyr

Endyr.filter - a filter for bnd that specifies the boundary where measEndyr should be applied

Value

xypltx: sf object. Input xy data clipped to boundary.

bndx: sf object. Input bnd.

tabs: list object. List of input layers clipped to boundary (pltx,condx,etc.).

xy.uniqueid: String. Name of unique identifier of xy.

puniqueid: String. Name of unique identifier of plot in plt.

pjoinid: String. Name of unique identifier of plot in plt.

If savedata=TRUE, outdat data frame is saved to outfolder.

Note

If savebnd=TRUE:

If out_fmt=c('csv','shp'), the writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.

If datsource="datmart", data are imported from FIA DataMart. The plot coordinates have been altered for privacy (See https://www.fia.fs.fed.us/tools-data/spatial/Policy/index.php for details). The zip files are extracted on-the-fly from the online website. Web server connections will affect download speeds.

Author(s)

Tracey S. Frescino

Examples

## Not run:
# Get polygon vector layer from FIESTA external data
WYbhfn <- system.file("extdata", 
  "sp_data/WYbighorn_adminbnd.shp", 
  package = "FIESTA")

# Extract data from FIA datamart for measurement years 2013 thru 2015
dat <- spGetPlots(bnd = WYbhfn, 
  datsource = "datmart", 
  measyrs = 2013:2015)

names(dat)
tabs <- dat$tabs
spGetSAdoms

spGetSAdoms

Spatial wrapper - Generate a set of model domain units for Small Area Estimation (SAE) strategies.

Description

Spatial process to generate a set of model domains (i.e., helper polygons) for Small Area Estimation (SAE) strategies. If helper_autoselect=TRUE, an automated process is used to select helper polygons within a large area overlapping the small area. The helper polygons are unioned with the small area polygons, resulting in a set of model domains that can be used for SAE.

Usage

spGetSAdoms(smallbnd,
smallbnd_dsn = NULL,
smallbnd.unique = NULL,
smallbnd.domain = NULL,
smallbnd.filter = NULL,
smallbnd.stfilter = NULL,
helperbnd = NULL,
helperbnd_dsn = NULL,
helperbnd.unique = NULL,
helperbnd.filter = NULL,
largebnd = NULL,
largebnd_dsn = NULL,
largebnd.unique = NULL,
largebnd.filter = NULL,
maxbnd = NULL,
maxbnd_dsn = NULL,
spGetSAdoms

maxbnd.unique = NULL,
maxbnd.filter = NULL,
helper_autoselect = TRUE,
nbrdom.min = NULL,
maxbnd.threshold = 10,
largebnd.threshold = 5,
multiSAdoms = FALSE,
showsteps = TRUE,
savedata = FALSE,
savesteps = FALSE,
saveobj = FALSE,
objnm = "SAdomdat",
maxbnd.addtext = TRUE,
largebnd.addtext = FALSE,
savedata_opts = NULL,
addstate = FALSE,
dissolve = FALSE,
byeach = FALSE

Arguments

smallbnd sf R object or String. Small area of interest boundary. Can be a spatial polygon object, full pathname to a shapefile, or name of a layer within a database.

smallbnd_dsn String. Data source name (dsn; e.g., sqlite or shapefile pathname) of smallbnd. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if smallbnd is an R object.

smallbnd.unique String. The attribute in smallbnd that defines unique domain identifier in smallbnd that defines the unique small area(s). If NULL, an attribute is appended to smallbnd attribute table and used as smallbnd.unique, defining one polygon (SMALLAREA="SMALLAREA").

smallbnd.domain String. A different attribute to use as for grouped modeling domains (optional). If NULL, smallbnd.domain=smallbnd.unique.


smallbnd.stfilter String. A spatial filter for smallbnd to include only smallbnd polygons that intersect (or overlap >= 30 boundary. The filter is based on the stunitco internal R object, with attributes: STATECD, STATENM, UNITCD, UNITNM, COUNTYCD, COUNTYNM. The filter should include one of these attributes and must be R syntax.

helperbnd sf R object or String. Name of polygon spatial layer delineating helper polygons for small area models. Can be a spatial polygon object, full pathname to a shapefile, or name of a layer within a database.
helperbnd_dsn  String. Data source name (dsn; e.g., sqlite or shapefile pathname) of helperbnd.
The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).
Optional if helperbnd is an R object.

helperbnd.unique
String. The attribute in helper polygon layer that defines unique helper polygons.

helperbnd.filter
String. A filter for helperbnd. Must be R syntax.

largebnd  sf R object or String. Name of large area polygon spatial layer, defining the
model data extent for building small area models. Can be a spatial polygon object,
full pathname to a shapefile, or name of a layer within a database.

largebnd_dsn  String. Data source name (dsn; e.g., sqlite or shapefile pathname) of largebnd.
The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).
Optional if largebnd is an R object.

largebnd.unique
String. The attribute in largebnd polygon layer that defines unique large area
polygon(s).

largebnd.filter

maxbnd  sf R object or String. Name of polygon spatial layer, defining the maximum
model data restraint for adding more helper polygons for building small area
models. Can be a spatial polygon object, full pathname to a shapefile, or name
of a layer within a database.

maxbnd_dsn  String. Data source name (dsn; e.g., sqlite or shapefile pathname) of maxbnd.
The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).
Optional if maxbnd is an R object.

maxbnd.unique
String. The attribute in maxbnd polygon layer that defines unique max restraint
area(s).

maxbnd.filter

helper_autoselect
Logical. If TRUE, the helper boundaries are automatically selected based on
intersection with maxbnd and/or largebnd and number of helperbnds defined by
nbrdom.min.

nbrdom.min
Integer. Set number for minimum domains for modeling. If NULL, all domains
within largebnd are selected.

maxbnd.threshold
Integer. Percent for including additional maxbnds for selecting helperbnds. If
multiSAdoms=FALSE, the maxbnd with greatest percentage over the maxbnd.threshold
is selected.

largebnd.threshold
Integer. Percent for including additional largebnds for selecting helperbnds.

multiSAdoms
Logical. If TRUE, and the percent intersect of smallbnd with maxbnd is greater
than maxbnd.threshold, more than 1 SAdoms will be output in list.

showsteps
Logical. If TRUE, intermediate steps of selection process are displayed.

savedata
Logical. If TRUE, save SAdoms spatial layer to outfolder.
spGetSAdoms

savesteps Logical. If TRUE, save steps spatial intermediate layers and JPG images. All spatial layers are output as *.shp format in a separate folder (SAdoms_steps).

saveobj Logical. If TRUE, save SAdomdat object to outfolder.

objnm String. Name of *.rds object.

maxbnd.addtext Logical. If TRUE, adds text to intermediate step plots for maxbnd displays.

largebnd.addtext Logical. If TRUE, adds text to intermediate step plots for largebnd displays.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

addstate Logical. If TRUE, appends state attribute to SAdoms.

dissolve Logical. If TRUE, aggregates polygons to smallbnd.domain or smallbnd.unique.

byeach Logical. If TRUE, creates an SAdom for each smallbnd polygon.

Details

optional boundaries

The helperbnd, largebd, and maxbnd are optional. If helperbnd=NULL, the smallbnd polygons are used for model domain units. If largebd=NULL, the maxbnd is used to define the large area. If maxbnd=NULL, the largebd is used to restrain the model extent. If both, largebd=NULL and maxbnd=NULL, the extent of the smallbnd or helperbnd is used for defining and restraining the model extent.

nbrdom.min

The number of helper polygons selected are defined by nbrdom.min parameter. If nbrdom.min=NULL, all helper polygons within the large area extent are selected.

multiSAdoms

Use multiSAdoms parameter when small area of interest has multiple polygon features and the small area polygons overlap (within maxbnd.threshold) more than one maxbnd polygon. If multiSAdoms=TRUE, more than one set of model domain units are generated; one for each maxbnd where overlap is within maxbnd.threshold. If multiSAdoms=FALSE, only one set of model domain units are generated, using the maxbnd with the greatest overlap.

AOI attribute

A variable named ’AOI’ is appended to the SAdoms attribute table to distinguish between the small area of interest polygons and the helper domain units.

Value

SAdomslst List object. Set(s) of model domain units. If multiSAdoms=TRUE, the list may have more than one set of model domain units.

smallbndlst List object. smallbnd(s). If multiSAdoms=TRUE, the list may have more than one set of smallbnd.

If exportsp=TRUE, the SAdoms spatial object(s) is exported to outfolder, with format specified by out_fmt.
Note

If exportsp=TRUE and out_fmt="shp":
The writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.

The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

Author(s)

Tracey S. Frescino

spGetStates

Spatial wrapper - Extracts states that intersect a boundary.

Description

Wrapper to get state names that intersect a given boundary.

Usage

spGetStates(
  bnd_layer,
  bnd_dsn = NULL,
  bnd.filter = NULL,
  stbnd.att = "COUNTYFIPS",
  RS = NULL,
  states = NULL,
  overlap = 1,
  showsteps = FALSE,
  savebnd = FALSE,
  savedata_opts = NULL
)

Arguments

bnd_layer sf R object, Area of Interest (AOI) boundary. Can be a spatial sf object, full pathname to a shapefile, or name of a layer within a database.

bnd_dsn String. Data source name (dsn; e.g., SQLite database or shapefile pathname) of bnd. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).

bnd.filter String. Filter to subset bnd spatial layer.
spGetStrata

Spatial wrapper - Extracts point attribute values and pixel counts for strata and estimation unit spatial layers.

Description

Wrapper to extract attribute and area from a polygon or raster estimation unit layer and a polygon or raster layer with strata pixel categories.

Value

A list containing states and state names that the boundary crosses, and boundary and attribute information for the intersecting boundary.

Author(s)

Tracey S. Frescino

Examples

# Get polygon vector layer from FIESTA external data
WYbhdistfn <- system.file("extdata",
   "sp_data/WYbighorn_districtbnd.shp",
   package = "FIESTA")

# Get intersecting statenames
spGetStates(WYbhdistfn)$statenames

# Get intersecting COUNTYFIP codes
spGetStates(WYbhdistfn,
   stbnd.att = "COUNTYFIPS")$states
Usage

spGetStrata(
  xyplt,  
  xyplt_dsn = NULL,  
  unit_layer,  
  unit_dsn = NULL,  
  uniqueid = "PLT_CN",  
  unitvar = NULL,  
  unit.filter = NULL,  
  strattype = "RASTER",  
  strat_layer = NULL,  
  strat_dsn = NULL,  
  strvar = NULL,  
  strat_lut = NULL,  
  areaunits = "acres",  
  rast.NODATA = NULL,  
  keepNA = FALSE,  
  showext = FALSE,  
  returnxy = FALSE,  
  savedata = FALSE,  
  exportsp = FALSE,  
  exportNA = FALSE,  
  spMakeSpatial_opts = NULL,  
  savedata_opts = NULL,  
  vars2keep = NULL,  
  gui = FALSE)
)

Arguments

xyplt  
Data frame, sf object, full pathname to *.csv or *shp, or layer name in a geodatabase. Includes XY coordinates and unique identifier. If non-spatial, include options in spMakeSpatial_opts parameter.

xyplt_dsn  
String. Name of database where xyplt is. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).

unit_layer  
sf R object or String. Name of estimation unit spatial layer. Can be a spatial polygon object, full pathname to a shapefile, name of a polygon layer within a database, or a full pathname to raster file.

unit_dsn  
String. Data source name (dsn; e.g., sqlite or shapefile pathname) of unit_layer. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if unit_layer is sf object.

uniqueid  
String.* Unique identifier of xyplt records. Note: raster unit layers are converted to polygon.

unitvar  
String. If unittype="POLY", name of attribute in unit_layer defining estimation units. If NULL, the unit_layer represents one estimation unit.

unit.filter  
String. Filter to subset unit_layer spatial layer.
strattype String. Spatial layer type of strat_layer ("POLY", "RASTER"). Note: polygon strata layers are converted to raster.

strat_layer sf R object or full pathname of spatial stratification layer. Can be a spatial polygon object, full pathname to a shapefile, name of a polygon layer within a database, or a full pathname to raster file.

strat_dsn String. Data source name (dsn; e.g., sqlite or shapefile pathname) of strat_layer. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if unit_layer is sf object.

strvar String. If strattype="POLY", name of strata attribute in strat_layer.

strat_lut Data frame. A look-up table of codes to aggregate. The format of table includes 2 columns, one column same name as strvar. If strattype="RASTER", strvar="value".

areaunits String. Output area units ("acres", "hectares", "sqmeters").

rast.NODATA Numeric. NODATA value if stratlayer is raster (See notes). This values will be converted to NA and removed from output. if keepNA=TRUE, NA values will not be in included in strataltut but will remain in pltassgn table.

keepNA Logical. If TRUE, returns data frame of NA values.

showext Logical. If TRUE, layer extents are displayed in plot window.

returnxy Logical. If TRUE, returns xy data as sf object (spxyplt).

savedata Logical. If TRUE, the input data with extracted values are saved to outfilder.

exportsp Logical. If savedata=TRUE and returnxy=TRUE, If TRUE, the extracted strata point data are exported to outfilder.

exportNA Logical. If TRUE and keepNA=TRUE, NA values are exported to outfilder as a point shapefile.

spMakeSpatial_opts List. See help(spMakeSpatial_options()) for a list of options. Use to convert X/Y values to simple feature (sf) coordinates.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

vars2keep String vector. Attributes in SAdoms, other than domvar to include in dunitlut output and extract to pltassgn points.

gui Logical. If gui, user is prompted for parameters.

Details

*If variable = NULL, then it will prompt user for input.

If spatial layers have different projections, the polygon spatial layer is transformed to the projection of raster (See note about on-the-fly projection conversion). If both layers are long/lat coordinate system, they are transformed to default coordinate system (Conus Albers, NAD83).
Value

pltassgn  Data frame. Input xyplt data with extracted estimation unit and strata values appended.

unitarea  Data frame. Area by estimation unit.

unitvar  Data frame. Variable name for estimation unit in unitarea.

acrevar  Data frame. Variable name for area in unitarea.

stratalut  Data frame. Strata proportions (weights) by estimation unit and strata.

strvar  Data frame. Variable name for strata values in stratalut.

$\text{NAlist}$  sf List. If keepNA=TRUE, and NA values exist after data extraction, the spatial NA points are returned.

pltassgnid  String. Unique identifier of plot.

spx  Simple feature. If returnxy=TRUE, Spatial coordinates.

xy.uniqueid  String. If returnxy=TRUE, unique identifier of spx.

If savedata=TRUE, pltassgn and unitarea are saved to outfolder.

If exportsp=TRUE, the spatial sf points object is exported to outfolder.

Note

rast.NODATA

NODATA values are raster pixel values that have no data of interest, including pixels within the extent of the layer, but outside the area of interest. Sometimes these pixels have been defined previously. The defined NODATA pixels are imported to R as NULL values. When not previously defined, the pixels outside the area of interest will be the minimum or maximum value depending on the data type (e.g., 16-bit signed: min=-32,768; max=32,768) or byte size (1 byte: min=0; max=255). These NODATA values will be added to the zonal statistic calculations if not specified in rast.NODATA.

If exportsp=TRUE:
If out_fmt="shp", the writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.

On-the-fly projection conversion

The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

unitarea

Area by estimation unit is calculated and returned as object named unitarea. Area is based on the projection of unit_layer. If no unit_layer input, than area is calculated from pixel counts.
polygon to raster
If strattype="POLY", a raster template is created based on the masked extent of strat_layer, with strat_layer projected coordinate system and 30 meter pixel size.

Author(s)
Tracey S. Frescino

Examples

# Create a `SpatialPoints` object from `WYplt`
WYspplt <- spMakeSpatialPoints(xyp = WYplt,
                               xy.uniqueid = "CN",
                               xvar = "LON_PUBLIC",
                               yvar = "LAT_PUBLIC",
                               xy.crs = 4269)

# Set up stratification from object in `FIESTA`
fornffn <- system.file("extdata",
                       "sp_data/WYbighorn_forest_nonforest_250m.tif",
                       package = "FIESTA")

# Set up data from FIESTA
WYbhfn <- system.file("extdata",
                      "sp_data/WYbighorn_adminbnd.shp",
                      package = "FIESTA")

# Run `spGetStrata`
spGetStrata(WYspplt,
            uniqueid = "CN",
            unit_layer = WYbhfn,
            strattype = "RASTER",
            strat_layer = fornffn)

---

**spGetXY**

*Spatial wrapper - Extracts XY coordinates within a given boundary.*

**Description**

Wrapper to get FIA plots within the boundary population (area of interest) - Intersect with state boundary - Get FIA plots for intersected states, including tree, and spatial - Clip spatial coordinates and other tables to boundary (spClipPoint)

**Usage**

```r
spGetXY(  
  bnd,
  bnd_dsn = NULL,
  bnd.filter = NULL,
  ...)```
states = NULL,
RS = NULL,
xy_datsource = "datamart",
xy = NULL,
xy_dsn = NULL,
xy.uniqueid = "PLT_CN",
xvar = NULL,
yvar = NULL,
xy.crs = 4269,
xyjoinid = NULL,
pjoinid = "CN",
clipxy = TRUE,
plot_layer = "plot",
evalid = NULL,
evalCur = FALSE,
evalEndyr = NULL,
measCur = FALSE,
measEndyr = NULL,
measEndyr.filter = NULL,
invyrs = NULL,
measyrs = NULL,
allyrs = FALSE,
intensity1 = FALSE,
showsteps = FALSE,
returnxy = TRUE,
savedata = FALSE,
exportsp = FALSE,
savedata_opts = NULL
)

**Arguments**

- **bnd**  
  sf R object, Area of Interest (AOI) boundary. Can be a spatial sf object, full pathname to a shapefile, or name of a layer within a database.

- **bnd_dsn**  
  String. Data source name (dsn; e.g., SQLite database or shapefile pathname) of bnd. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if bnd is an R object.

- **bnd.filter**  
  String. Filter to subset bnd spatial layer.

- **states**  
  String. The name of state(s) for tables (e.g., "Vermont", "Utah").

- **RS**  
  String. Name of FIA research station to restrict states to ("RMRS", "SRS", "NCRS", "NERS", "PNWRS"). If NULL, all research stations are included.

- **xy_datsource**  
  String. Source of XY data ("obj", "csv", "datamart", "sqlite"). If datsource=NULL, checks extension of xy_dsn or xy to identify datasource.

- **xy**  
  sf R object or String. Table with xy coordinates. Can be a spatial polygon object, data frame, full pathname to a shapefile, or name of a layer within a database.

- **xy_dsn**  
  String. Data source name (dsn; i.e., pathname or database name) of xy. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).
spGetXY

Optional if bnd_layer is an R object.

xy.uniqueid String. Unique identifier of xy.
xvar String. Name of variable in xyplt defining x coordinate.
yvar String. Name of variable in xyplt defining y coordinate.
xy.crs PROJ.4 String or CRS object or Integer EPSG code defining Coordinate Reference System.
xyjoinid String. Variable in xy to join to plot data. If NULL, xyjoinid=xy.uniqueid.
pjoinid String. Variable in plt to join to XY data. Not necessary to be unique. If using most current XY coordinates, use identifier for a plot (e.g., PLOT_ID).
clipxy Logical. If TRUE, clips xy data to bnd.
plot_layer String. Name of plot_layer in database.
evalid Integer. To extract data for a specific evaluation period. See notes for more information about FIA Evaluations.
evalCur Logical. If TRUE, extract plots with most current FIA Evaluation for state(s).
evalEndyr Integer. Defining end year of Evaluation (yyyy).
measCur Logical. If TRUE, extract plots with most current measurement for state(s).
measEndyr Integer year (YYYY). If measCur=TRUE, extract plots with most current measurement for state(s) for years measured before measEndyr.
measEndyr.filter Filter. If measCur=TRUE and measEndyr != NULL, a filter for bnd to identify and area to use measEndyr, such as disturbed areas where you want to exclude plots measured after disturbance.
invyrs Integer vector. Defining specific inventory years of data (e.g., 2010:2015).
measyrs Integer vector. Defining specific measurement years of data (e.g., 2010:2015).
allyrs Logical. If TRUE, selects all years (annual inventory) in database.
intensity1 Logical. If TRUE, include only single intensity plots (i.e., INTENSITY = 1).
showsteps Logical. If TRUE, display data in device window.
returnxy Logical. If TRUE, returns XY coordinates.
savedata Logical. If TRUE, saves data to outfolder. Note: includes XY data if returnxy = TRUE.
exports Logical. If savedata = TRUE and returnxy = TRUE, if TRUE, exports XY data as spatial data.
savedata.opts List. See help(savedata_options()) for a list of options. Only used when savedata = TRUE.

Details
datasource
Plots are extracted from 3 different data sources:
1) CSV - data have previously been extracted from the FIA database and stored as CSV files.
2) datamart - data are extracted from FIA's publically-available datamart.
3) sqlite - data have previously been extracted from the FIA database and stored within a SQLite database.

**Selection parameters**

FIA plots are selected based on the following parameters:

- evalid - the FIA evaluation identifier
- evalCur - the most current FIA evaluation in database
- evalEndyr - the FIA evaluation ending in evalEndyr
- evalType - the FIA evaluation type ("ALL", "AREAVOL", "GRM", "P2VEG", "DWM", "INV", "REGEN", "CRWN")
- measCur - the most current measurement of each plot in database
- measEndyr - the most current measurement of each plot in database in or prior to measEndyr
- Endyr.filter - a filter for bnd that specifies the boundary where measEndyr should be applied

**Value**

- spxy sf. If returnxy=TRUE, spatial xy point data.
- pltids data frame. A table of pltids that are within bnd.
- spxy sf data frame. If returnxy, a simple feature with pltids within bnd.
- bndx sf object. Input bnd.
- xy.uniqueid String. Unique identifier of plots in xy.
- states String. Vector of states that intersect bnd.
- countyfips String. Vector of countyfips values that intersect bnd.
- stbnd.att String. Name of state attribute used to select plots.

If savedata=TRUE and returnxy=TRUE, the plt data frame, including XY coordinates is saved to outfolder (xyplt).
If savedata=TRUE and returnxy=FALSE, the plt data frame, without XY coordinates is saved to outfolder (pltids).
If savedata=TRUE and returnxy=TRUE and exportsp=TRUE, the spxy sf object is exported as shapefile to outfolder.

**Note**

If savebnd=TRUE:
- If out_fmt=c('csv','shp'), the writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.
- If datasource="datmart", (default), data are imported from FIA DataMart. The plot coordinates have been altered for privacy (See https://www.fia.fs.fed.us/tools-data/spatial/Policy/index.php for details). The zip files are extracted on-the-fly from the online website. Web server connections will affect download speeds.
spImportSpatial

Author(s)
Tracey S. Frescino

Examples
## Not run:
# Set up data from FIESTA
WYbhfn <- system.file("extdata",
"sp_data/WYbighorn_adminbnd.shp",
package = "FIESTA")
# Use spGetXY
WYbhxy <- spGetXY(bnd = WYbhfn,
xy_datsource = "datamart",
evalCur = TRUE,
returnxy = TRUE)
## End(Not run)

spImportSpatial  Spatial - Imports a spatial vector layer to an S4 Spatial object.

Description
Imports a spatial vector layer to an S4 Spatial object.

Usage
spImportSpatial(
  layer = NULL,
  dsn = NULL,
  sql = NULL,
  polyfix = FALSE,
  gui = FALSE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layer</td>
<td>Data frame object or String. Name of spatial layer. Can be layer with dsn, full pathname, including extension, or file name (with extension) in xy_dsn folder.</td>
</tr>
<tr>
<td>dsn</td>
<td>String. Name of database where layer is. The dsn varies by driver. See gdal OGR vector formats (<a href="https://www.gdal.org/ogr_formats.html">https://www.gdal.org/ogr_formats.html</a>).</td>
</tr>
<tr>
<td>sql</td>
<td>String. A sql syntax query to subset spatial layer.</td>
</tr>
<tr>
<td>polyfix</td>
<td>Logical. If polyfix=TRUE, uses buffer with 0 width to clean up polygons.</td>
</tr>
<tr>
<td>gui</td>
<td>Logical. If TRUE, search for layer within dsn.</td>
</tr>
</tbody>
</table>

Value
A spatial object
Note

Wrapper for sf package... st_read function.

Author(s)

Tracey S. Frescino

Examples

# Import data from 'FIESTA', save as object in environment
WYbh <- spImportSpatial(system.file("extdata",
    "sp_data/WYbighorn_adminbnd.shp",
    package = "FIESTA"))

spMakeSpatialPoints  Spatial - Generates an S4 SpatialPoints object from X/Y coordinates.

Description

Generates an S4 SpatialPoints object with defined projection from a data table or matrix including X and Y coordinates, with option to export as an ArcGIS shapefile (*.shp).

Usage

spMakeSpatialPoints(
    xyplt,
    xyplt_dsn = NULL,
    xy.uniqueid = NULL,
    xvar = NULL,
    yvar = NULL,
    xy.crs = 4269,
    prj = NULL,
    datum = NULL,
    zone = NULL,
    zoneS = FALSE,
    aea.param = "USGS",
    addxy = FALSE,
    exportsp = FALSE,
    savedata.opts = NULL
)

Arguments

xyplt  Data frame object or String. Name of layer with xy coordinates and unique identifier. Can be layer with xy_dsn, full pathname, including extension, or file name (with extension) in xy_dsn folder.
spMakeSpatialPoints

xyplt_dsn  String. Name of database or folder were xyplt is. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).

xy.uniqueid  String. Unique identifier of xyplt rows.

xvar  String. Name of variable in xyplt defining x coordinate.

yvar  String. Name of variable in xyplt defining y coordinate.

xy.crs  PROJ.4 String or CRS object or Integer EPSG code defining Coordinate Reference System. (e.g., EPSG:4269-Geodetic coordinate system for North America, NAD83).

prj  String. Projection, or coordinate system of the X/Y coordinates ("longlat", "utm", "aea"). If other, include PROJ.4 string in prj4str.

datum  String. Datum of projection ("WGS84", "NAD83", "NAD27").

zone  Integer. If prj="utm", the UTM zone.

zoneS  Logical. If prj="utm", if the UTM zone is in the Southern hemisphere.

aea.param  String. If prj="aea", the associated lat/lon parameters (USGS: " +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0 +y_0=0"). If other, include PROJ.4 string in prj4str.

addxy  Logical. If TRUE, adds x and y variables to spatial sf object.

exportsp  Logical. If TRUE, exports spatial object.

savedata_opts  List. See help(savedata_options()) for a list of options. Only used when exportsp = TRUE.

Value

spplt  sf object with spatial points and defined CRS.

If exportsp = TRUE, the sf object is written to specified output.

Note

If exportsp=TRUE and a shp output format is specified:
The ESRI shapefile driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). Name changes are output to the outfolder, ‘outshpnm’_newnames.csv. The returned Spatial object will have original names, before truncating.

If Spatial object has more than 1 record, it cannot be exported.

Common Datums and associated spheroid (ellipsoid):
NAD27 - North American Datum of 1927 - Clarke 1866 spheroid
NAD83 - North American Datum of 1983 - GRS 1980 spheroid
WGS84 - World Geodetic System of 1984 - WGS 1984 spheroid

Author(s)

Tracey S. Frescino
**spPoly2Rast**

*Spatial - Converts SpatialPolygons layer to raster.*

### Examples

```r
# Generate an `sf` points object with `spMakeSpatialPoints` for Wyoming plot data, stored in `FIESTA`
spMakeSpatialPoints(xyplt = WYplt,
  xy.uniqueid = "CN",
  xvar = "LON_PUBLIC",
  yvar = "LAT_PUBLIC",
  prj = "longlat",
  datum = "NAD83")
```

### Description

Converts SpatialPolygons layer to raster.

### Usage

```r
spPoly2Rast(
  polyv,
  polyv_dsn = NULL,
  polyv.att,
  polyv.lut = NULL,
  rastfn.template = NULL,
  NODATA = NULL,
  outfolder = NULL,
  outfn = "polyrast",
  outext = "img",
  outfn.pre = NULL,
  outfn.date = TRUE,
  overwrite = FALSE
)
```

### Arguments

- **polyv**: sf R object or String. Polygon data to convert to raster. Can be a spatial polygon object, full pathname to a shapefile, or name of a layer within a database.
- **polyv_dsn**: String. Data source name (dsn; e.g., sqlite or shapefile pathname) of layer to convert. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if polyv is sf object.
- **polyv.att**: String. Name of attribute in polyv to rasterize.
- **polyv.lut**: Data frame. Look up table of codes, if polyv.att is character or want to group codes.
- **rastfn.template**: String. Full path name of raster to use as template for new raster.
NODATA: Number. The NODATA value for background values. If NODATA is NULL, and a NODATA value is defined on the rastfn.template raster, the default is the defined NODATA value, else it is defined based on its datatype (see DEFAULT_NODATA for default data values).

outfolder: String. If exportshp=TRUE, name of output folder. If NULL, the working directory is used.

outfn: String. Name of output raster. If NULL, default is ‘polyrast’.

outext: String. Name of raster extension (fmt).

outfn.pre: String. Add a prefix to output name (e.g., "01").

outfn.date: Logical. If TRUE, add date to end of outfile (e.g., outfn_date.csv).

overwrite: Logical. If TRUE and exportshp=TRUE, overwrite files in outfolder.

Value

A list containing raster and raster information derived from the original polygon.

Note

On-the-fly projection conversion
The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

If exportshp=TRUE:
The writeOGR (rgdal) function is called. The ArcGIS driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If Spatial object has more than 1 record, it will be returned but not exported.

Author(s)

Tracey S. Frescino

Examples

# Get polygon vector layer from FIESTA external data
WYbhdistfn <- system.file("extdata",
   "sp_data/WYbighorn_districtbnd.shp",
   package = "FIESTA")

# Turn polygon into raster
# Note: raster values must be numeric, therefore names were changed to
# numeric codes based on lookup table produced from the following code.
new_rast <- spPoly2Rast(polyv = WYbhdistfn,
   polyv.att = "DISTRICTNA",
   polyv.pre = "01",
   outfn = "polyrast",
   outext = "csv",
   overwrite = TRUE
)
spReprojectRaster

Spatial - Reprojects an Esri shapefile (*.shp) or S4 Spatial object.

Description

Reprojects an Esri shapefile (*.shp) or S4 Spatial object to a new geographic or projected coordinate system, with option to save new object.

Usage

spReprojectRaster(
  rastfn,
  bands = NULL,
  crs = NULL,
  rast.ref = NULL,
  crs.new = NULL,
  res.new = NULL,
  bbox.new = NULL,
  dtype.new = NULL,
  NODATA.new = NULL,
  resamp.method = "near",
  crs.default = "EPSG:5070",
  compress = NULL,
  BigTIFF = FALSE,
  outfolder = NULL,
  outfn = NULL,
  outext = NULL,
  overwrite = FALSE
)

Arguments

rastfn String or Raster. File name(s) with extensions, or raster object(s). Note: raster objects must be written to file.

bands Numeric vector. If rast is a multi-layer raster and only 1 or some layers are desired, specify layer number(s) in a vector format. If NULL, all layers are projected.

crs Coordinate Reference System (CRS). The CRS of rastfn if not defined. EPSG:code, PROJ.4 declaration, or .prj file containing WKT. For example, PROJ.4: "+proj=moll +lon_0=0 +x_0=0 +y_0=0 +ellps=WGS84 +datum=WGS84 +units=m +no_defs". If NULL, and the CRS of rastfn is not defined, uses crs.default.
spReprojectRaster

rast.ref String or Raster. File name(s) with extensions, or raster object to use as reference raster.

crs.new Coordinate Reference System. New CRS for rastfn. EPSG:code, PROJ.4 declaration, or .prj file containing WKT. For example, PROJ.4: "+proj=moll +lon_0=0 +x_0=0 +y_0=0 +ellps=WGS84 +datum=WGS84 +units=m +no_defs".

res.new Integer vector. One or two values defining new resolution of raster (in target georeferenced units) (e.g., 30 or c(30,30)).

bbox.new <xmin ymin xmax ymax> Georeferenced extent or bounding box of new raster.

dtype.new String. Force a data type of new raster. If NULL, the data type will be same as rastfn (e.g., Byte, Int16, UInt16).

NODATA.new Integer. Set nodata values for new raster. New files will be initialized to this value and if possible the nodata value will be recorded in the output file. Use a value of "None" to ensure that nodata is not defined. If NULL, NODATA and rastfn has a set NODATA value, this value will be used for new raster.

resamp.method Method for resampling ('near', 'bilinear', 'cubic', 'cubicspline', 'landzos', 'average', 'mode', 'min', 'max', 'med', 'q1', 'q3').

crs.default Coordinate Reference System. A default CRS if crs.new=NULL. The default is: EPSG:5070, Conus Albers, PRJ4='+proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23.5 +lon_0=-96. +x_0=0 +y_0=0', '+ellps=GRS80 +towgs84=0,0,0,-0,-0,-0,0 +units=m +no_defs'.

compress String. An optional compression type ('LZW', "DEFLATE", "PACKBITS").

BigTIFF Logical. If TRUE, compress option for big files (>4GB).

outfolder String. If exportsp=TRUE, name of output folder. If NULL, the working directory is used.

outfn String. Name of output raster. If NULL, default is 'polyrast'.

outext String. Name of raster extension (fmt). If NULL, uses extension from outfn or rastfn.

overwrite Logical. If TRUE, overwrites raster file.

Value

rastfn.new String. Full path name to reprojected raster.

Note

Coordinate Reference Systems (CRS)

An ellipse is an estimated model describing the basic shape of the Earth and is the basis for all coordinate systems. There are many ellipsoids designed for local (e.g., NAD27) or global (e.g., WGS84, GRS80) use. The datum defines the reference position of the coordinate axes associated with a specific ellipsoid. Specifying the datum also defines the ellipsoid, whereas specifying the ellipsoid does not provide information of the datum.

WGS84 vs NAD83 WGS84 and NAD83 datums are often used interchangeably, and use very similar ellipsoids (WGS84 and GRS80, respectively), but have different reference points. Slight tectonic shifts through time have caused increased divergence between the two, with NAD83 datum intended to track movements more consistently.
Common Datums and associated spheroid (ellipsoid):
NAD27 - North American Datum of 1927 (Clarke 1866 spheroid)
WGS84 - World Geodetic System of 1984 (WGS 1984 spheroid)

From R, use projInfo for list of different projections and datums.
> projInfo(type="proj")
> projInfo(type="datum")

Common EPSG Geodetic codes in U.S.
EPSG:4326 - Longitude/Latitude (WGS84) - Common for global displays (used by Google Earth)
EPSG:4269 - Longitude/Latitude (NAD83) - Common by U.S. Federal Agencies

The sf::st_transform (GDAL) method is used for map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

Author(s)
Tracey S. Frescino, Chris Toney

Examples

# Get raster layers from FIESTA external data
demfn <- system.file("extdata",
  "sp_data/WYbighorn_dem_250m.img",
  package = "FIESTA")

# Plot original projection
raster::plot(raster::raster(demfn))

# Check original projection
sf::st_crs(raster::raster(demfn))$proj4string

# Reproject raster
reprojected <- spReprojectRaster(rastfn = demfn,
  crs.new = "EPSG:32613",
  outfolder = tempdir())

# Plot new projection
raster::plot(raster::raster(reprojected))

# Check new projection
sf::st_crs(raster::raster(reprojected))$proj4string
spReprojectVector

Spatial - Reprojects an sf spatial object.

Description
Reprojects an sf spatial object to a new coordinate reference system.

Usage
spReprojectVector(
  layer,
  dsn = NULL,
  crs.new,
  exportsp = FALSE,
  savedata_opts = NULL
)

Arguments
layer sf class R object or String. The spatial layer must have a defined projection (test using sf::st_crs(layer)).
dsn String. Data source name (dsn; i.e., folder or database name) of splayer. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if layer is an R object.
crs.new EPSG Integer or PROJ.4 String. New EPSG Geodetic Parameter Dataset definition or gdal PROJ.4 string identifying the new coordinate system (e.g., "+proj=longlat +datum=NAD83 +no_defs +ellps=GRS80 +towgs84=0,0,0").
exportsp Logical. If TRUE, the spatial reprojected object is exported to outfolder (see spExportSpatial for details).
savedata_opts List. See help(savedata_options()) for a list of options for saving data. If out_layer = NULL, default = 'layerprj'.

Value
layerprj sf object. Reprojected spatial layer.

If exportsp = TRUE, a spatial layer is written to outfolder (See note).

Note
Coordinate Reference Systems (CRS)
An ellipse is an estimated model describing the basic shape of the Earth and is the basis for all coordinate systems. There are many ellipsoids designed for local (e.g., NAD27) or global (e.g., WGS84, GRS80) use. The datum defines the reference position of the coordinate axes associated with a specific ellipsoid. Specifying the datum also defines the ellipsoid, whereas specifying the ellipsoid does not provide information of the datum.
WGS84 vs NAD83 WGS84 and NAD83 datums are often used interchangeably, and use very similar ellipsoids (WGS84 and GRS80, respectively), but have different reference points. Slight tectonic shifts through time have caused increased divergence between the two, with NAD83 datum intended to track movements more consistently.

Common Datums and associated spheroid (ellipsoid):
NAD27 - North American Datum of 1927 (Clarke 1866 spheroid)
WGS84 - World Geodetic System of 1984 (WGS 1984 spheroid)

From R, use projInfo for list of different projections and datums.
> projInfo(type="proj")
> projInfo(type="datum")

Common EPSG Geodetic codes in U.S.
EPSG:4326 - Longitude/Latitude (WGS84) - Common for global displays (used by Google Earth)
EPSG:4269 - Longitude/Latitude (NAD83) - Common by U.S. Federal Agencies

The sf::st_transform (GDAL) method is used for map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

ESRI Shapefile Driver
If exportsp=TRUE:
The writeOGR (rgdal) function is called. If out_fmt="shp", the ESRI Shapefile driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If sf object has more than 1 record, it will be returned but not exported.

Author(s)
Tracey S. Frescino

Examples

```r
# Set up `SpatialPoints` object
WYspplt <- spMakeSpatialPoints(xyplt = WYplt,
    xy.uniqueid = "CN",
    xvar = "LON_PUBLIC",
    yvar = "LAT_PUBLIC",
    prj = "longlat",
    datum = "NAD83")

# Check CRS
sf::st_crs(WYspplt)

# Set up projection
```
spUnionPoly

spUnionPoly

Spatial - Generate a unioned sf object with polygons and attributes from two sf polygon objects.

Description

Generate a unioned sf object with polygons and attributes from two sf polygon objects.

Usage

spUnionPoly(
  polyv1,
  polyv1_dsn = NULL,
  polyv2,
  polyv2_dsn = NULL,
  showext = FALSE,
  areacalc = FALSE,
  areavar = "ACRES_GIS",
  exportsp = FALSE,
  savedata_opts = NULL,
  ...
)

Arguments

polyv1 sf R object or String. Polygon data to union. Can be a spatial polygon object, full pathname to a shapefile, or name of a layer within a database.

polyv1_dsn String. Data source name (dsn; e.g., sqlite or shapefile pathname) of layer to union. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if polyv1 is sf object.

polyv2 sf R object or String. Polygon data to union. Can be a spatial polygon object, full pathname to a shapefile, or name of a layer within a database.

polyv2_dsn String. Data source name (dsn; e.g., sqlite or shapefile pathname) of layer to union. The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html). Optional if polyv2 is sf object.

showext Logical. If TRUE, layer extents are displayed in plot window.

areacalc Logical. If TRUE, calculate area of unioned polygons and append to attribute table (See details).
### spUnionPoly

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>areavar</td>
<td>String. Name of area variable.</td>
</tr>
<tr>
<td>exportsp</td>
<td>Logical. If TRUE, the spatial unioned object is exported to outfolder (see spExportSpatial for details).</td>
</tr>
<tr>
<td>savedata_opts</td>
<td>List. See help(savedata_options()) for a list of options. Only used when export = TRUE. If out_layer = NULL, default = ‘polyunion’.</td>
</tr>
</tbody>
</table>

#### Details

- *If variable = NULL, then it will prompt user for input.*
- Uses raster function union to merge two polygons and crop, if clip=TRUE. Generates a new ID for each polygon and appends attributes from both polygons.
- areacalc
  - If areacalc = TRUE and the unioned spatial object is not in a projected coordinate system (i.e., longlat), the object will be reprojected to the Albers Equal Area projection before area is calculated.

#### Value

- sf object of unioned polygon. If polyv1 and polyv2 have different projections, the projection of returned object will have the same projection as poly1 (See note about on-the-fly projection conversion).
- If exportsp=TRUE, the sf object will be written to outfolder (See note).

#### Note

**On-the-fly projection conversion**

The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in the both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

**ESRI Shapefile Driver**

If exportsp=TRUE:

The writeOGR (rgdal) function is called. If out_fmt="shp", the ESRI Shapefile driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). If sf object has more than 1 record, it will be returned but not exported.

#### Author(s)

Tracey S. Frescino

#### Examples

```r
## Not run:
# Set up data from 'FIESTA' and 'raster'
WYbhfn <- system.file("extdata",
```
WYbh <- spImportSpatial(WYbhfn)
USAco <- raster::getData("GADM", country = "USA", level = 2)

# Generate unioned `sf` object
polyUnion <- spUnionPoly(polyv1 = USAco[USAco$NAME_1 == "Wyoming",],
                         polyv2 = WYbh,
                         areacalc = TRUE)

# Plot the result
plot(sf::st_geometry(polyUnion))

## End(Not run)

---

**spZonalRast**

`Spatial - Extracts summary statistics by polygon (i.e., zone) for a raster.`

### Description

Extracts summary statistics by polygon, or zone for a raster (single or multi-band).

### Usage

```r
spZonalRast(
  polyv,
  polyv_dsn = NULL,
  polyv.att = NULL,
  rastfn,
  rastfolder = NULL,
  bands = NULL,
  zonalstat,
  pixelfun = NULL,
  outname = NULL,
  showext = FALSE,
  rastlut = NULL,
  rast.NODATA = NULL,
  na.rm = TRUE,
  savedata = FALSE,
  savedata_opts = NULL
)
```

### Arguments

- **polyv**: sf R object or String. Polygon data to identify zones. Can be a spatial polygon object, full pathname to a shapefile, or name of a layer within a database.
polyv_dsn String. Data source name (dsn; e.g., sqlite or shapefile pathname) of zonal layer.
The dsn varies by driver. See gdal OGR vector formats (https://www.gdal.org/ogr_formats.html).
Optional if polyv is sf object.

polyv.att String. Name of attribute in polyv to identify zones for summarizing raster
statistics.

rastfn String or Raster. File name(s) with extensions, or raster object(s). Note: raster
objects must be written to file.

rastfolder String. Name of the folder with raster layers. Optional. Useful if all raster layers
are in same folder.

bands Numeric vector. If rast is a multi-layer raster and only 1 or some layers are
desired, specify layer number(s) in a vector format. If NULL, all layers are
summed.

zonalstat String vector. Zonal statistic(s) to return for rasters with continuous data ("mean",
"sum", "majority", "minority", "variety", "npxels") or rasters with discrete data
("count", "proportion").

pixelfun Function. A function to apply to the individual pixel values before calculating
sum and mean. The function should accept a single numeric argument (pixel
value) and return a single numeric argument.

outname String. Variable name for output. The output names will use outname as a prefix
to summary statistics (i.e., 'outname'.mean, 'outname'.sum).

showext Logical. If TRUE, layer extents are displayed in plot window.

rastlut Data frame. A look up table to recode raster values. Must be 2 columns: Column
1 with raster values and column 2 with recode values.

rast.NODATA Numeric. NODATA value or other values to ignore. These values will be re-
moved from output zonal table. See notes below.

na.rm Logical. If TRUE, Null values are removed before zonal statistic calculations.

savedata Logical. If TRUE, the zonal data are saved to outfolder.

savedata_opts List. See help(savedata_options()) for a list of options. Only used when savedata
= TRUE. If out_layer = NULL, default = 'zonalext'.

Details

Use spZonalRast() to prompt for input.

If the projection of polyv is different than the projection of rast, the polyv SpatialPolygons object is
converted to the projection of rast (See note about on-the-fly projection conversion).

Value

zonalext Data frame. Zonal statistics by polygon attribute (attribute).

outname String vector. Names of zonal statistic variables generated in zonalext data
frame.

rasterfile String vector. Names of raster file(s) associated with zonal statistic.

If savedata=TRUE, outdat data frame is saved to outfolder (Default name: zonalext_.date_.csv).
Note

rast.NODATA

NODATA values are raster pixel values that have no data of interest, including pixels within the extent of the layer, but outside the area of interest. Sometimes these pixels have been defined previously. The defined NODATA pixels are imported to R as NULL values. When not previously defined, the pixels outside the area of interest will be the minimum or maximum value depending on the data type (e.g., 16-bit signed: min=-32,768; max=32,768) or byte size (1 byte: min=0; max=255). These NODATA values will be added to the zonal statistic calculations if not specified in rast.NODATA.

On-the-fly projection conversion

The spTransform (rgdal) method is used for on-the-fly map projection conversion and datum transformation using PROJ.4 arguments. Datum transformation only occurs if the +datum tag is present in both the from and to PROJ.4 strings. The +towgs84 tag is used when no datum transformation is needed. PROJ.4 transformations assume NAD83 and WGS84 are identical unless other transformation parameters are specified. Be aware, providing inaccurate or incomplete CRS information may lead to erroneous data shifts when reprojecting. See spTransform help documentation for more details.

Author(s)

Tracey S. Frescino

Examples

# Set up data from `FIESTA`
WYbhdistfn <- system.file("extdata",
    "sp_data/WYbighorn_districtbnd.shp",
    package = "FIESTA")
demfn <- system.file("extdata",
    "sp_data/WYbighorn_dem_250m.img",
    package = "FIESTA")

# Import spatial data with `spImportSpatial`
WYbhdist <- spImportSpatial(WYbhdistfn)

# Extract mean and sum in `WYbhdist`
spZonalRast(polv = WYbhdist,
    polv.att = "DISTRICTNA",
    rastfn = demfn,
    zonalstat = c("mean", "sum"))

stunitco

SpatialPolygonsDataFrame with FIA state, unit, county codes and names

Description

SpatialPolygonsDataFrame with FIA state, unit, county codes and names
Usage

stunitco

Format

An object of class sf (inherits from data.frame) with 3233 rows and 8 columns.

Source


WYcond

FIA data. Condition-level data from FIA public database.

Description

FIA condition-level data for the state of Wyoming, inventory years 2011-2012 and was downloaded Sept 8, 2017 from FIADB_1.7.0.00.

Format

A dataframe with 26 columns and 6710 rows.

Source

FIA Database.

References

Description

FIA plot-level data for the state of Wyoming, inventory years 2011-2012 and was downloaded Sept 8, 2017 from FIADB_1.7.0.00.

Format

A data frame with 24 columns and 6336 rows.

Details

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CN</td>
<td>Unique FIADB identifier</td>
</tr>
<tr>
<td>PREV_PLT_CN</td>
<td>Previous unique FIADB identifier</td>
</tr>
<tr>
<td>INVYR</td>
<td>Inventory year</td>
</tr>
<tr>
<td>STATECD</td>
<td>State code (FIPS)</td>
</tr>
<tr>
<td>CYCLE</td>
<td>Inventory cycle number</td>
</tr>
<tr>
<td>SUBCYCLE</td>
<td>Inventory subcycle number (Do not use subcycle 99 for estimation)</td>
</tr>
<tr>
<td>UNITCD</td>
<td>Survey unit code</td>
</tr>
<tr>
<td>COUNTYCD</td>
<td>County code</td>
</tr>
<tr>
<td>PLOT</td>
<td>Phase 2 plot number (Public)</td>
</tr>
<tr>
<td>LON_PUBLIC</td>
<td>Longitude - fuzzed/swapped (Decimal degrees; NAD83)</td>
</tr>
<tr>
<td>LAT_PUBLIC</td>
<td>Latitude - fuzzed/swapped (Decimal degrees; NAD83)</td>
</tr>
<tr>
<td>PLOT_NONSAMPLE_REASN_CD</td>
<td>Plot nonsampled reason</td>
</tr>
<tr>
<td>SAMP_METHOD_CD</td>
<td>Sample method code</td>
</tr>
<tr>
<td>SUBP_EXAMINE_CD</td>
<td>Subplots examined code</td>
</tr>
<tr>
<td>MANUAL</td>
<td>Manual version number</td>
</tr>
<tr>
<td>INTENSITY</td>
<td>Intensity</td>
</tr>
<tr>
<td>MEASYEAR</td>
<td>Measurement year</td>
</tr>
<tr>
<td>MEASMON</td>
<td>Measurement month</td>
</tr>
<tr>
<td>MEASDAY</td>
<td>Measurement day</td>
</tr>
<tr>
<td>REMPER</td>
<td>Remeasurement period</td>
</tr>
<tr>
<td>DESIGNCD</td>
<td>Plot design</td>
</tr>
<tr>
<td>P2PANEL</td>
<td>Phase 2 panel number</td>
</tr>
<tr>
<td>SUBPANEL</td>
<td>Subpanel number</td>
</tr>
<tr>
<td>ELEV</td>
<td>Elevation (ft)</td>
</tr>
<tr>
<td>KINDCD</td>
<td>Sample kind</td>
</tr>
<tr>
<td>MORT_TYP_CD</td>
<td>Type of annual mortality volume (1:Current annual; 2:Periodic annual)</td>
</tr>
<tr>
<td>GROW_TYP_CD</td>
<td>Type of annual volume growth (1:Current annual; 2:Periodic annual)</td>
</tr>
<tr>
<td>NF_PLOT_NONSAMPLE_REASN_CD</td>
<td>Nonforest sampling status</td>
</tr>
<tr>
<td>P2VEG_SAMPLING_STATUS_CD</td>
<td>P2 vegetation sampling status</td>
</tr>
<tr>
<td>PLOT_STATUS_CD</td>
<td>Plot sampling status</td>
</tr>
<tr>
<td>NF_PLOT_STATUS_CD</td>
<td>Nonforest plot sampling status</td>
</tr>
<tr>
<td>NBRCND</td>
<td>DERIVED: Number of conditions for plot</td>
</tr>
</tbody>
</table>
Source

FIA national database (FIADB_1.6.0.02), downloaded September 18, 2016.

References


Description

FIA plot-level data for the state of Wyoming, inventory years 2011-2012 and was downloaded Sept 8, 2017 from FIADB_1.7.0.00. It includes strata assignments for FIA plot center location (1:GREEN; 2:BROWN).

Format

A dataframe with 24 columns and 3047 rows.

Source

FIA Database.

References

WYseed

*FIA data. Seedling data from FIA public database.*

**Description**

FIA seedling data for the state of Wyoming, inventory years 2011-2012 and was downloaded May 17, 2017 from FIADB_1.7.0.00.

**Format**

A dataframe with 10 columns and 1039 rows.

**Source**

FIA Database.

**References**


WYstratalut

*FIA data. Strata data from FIA public database.*

**Description**

FIA condition-level data for the state of Wyoming, inventory years 2011-2012 and was downloaded Sept 8, 2017 from FIADB_1.7.0.00. It includes P1POINTCNT and number of FIA plots by strata (1:GREEN; 2:BROWN) and by estimation unit (ESTN_UNIT).

**Format**

A dataframe with 6 columns and 39 rows.

**Source**

FIA Database.

**References**

Wytree  
*FIA data. Tree-level data from FIA public database.*

**Description**

FIA tree-level data for the state of Wyoming, inventory years 2011-2012 and was downloaded Sept 8, 2017 from FIADB_1.7.0.00.

**Format**

A dataframe with 45 columns and 37312 rows.

**Source**

FIA Database.

**References**


Wyunitarea  
*FIA data. Acres data from FIA public database.*

**Description**

FIA acres by estimation unit for the state of Wyoming, inventory years 2011-2012 and was downloaded Sept 8, 2017 from FIADB_1.7.0.00.

**Format**

A data table with 4 columns and 23 rows.

**Source**

FIA Database.

**References**

WYunitzonal

WYunitzonal  Zonal data. Zonal means for auxiliary data in counties in Wyoming.

Description

Zonal means and pixel counts for certain auxiliary data in counties in Wyoming. Includes county code variable to distinguish counties, and state code variable to distinguish states.

Format

A dataframe with 9 columns and 23 rows.
Index

* datasets
  DEFAULT_NODATA, 59
  ecomap, 59
  GDT_NAMES, 60
  kindcd3old, 60
  ref_ALP, 134
  ref_codes, 135
  ref_cond, 135
  ref_conversion, 136
  ref_diac12in, 136
  ref_domain, 137
  ref_estvar, 137
  ref_plt, 138
  ref_shp, 138
  ref_species, 139
  ref_statecd, 139
  ref_titles, 140
  ref_tree, 140
  stunitco, 195
  WYcond, 196
  WYplt, 197
  WYpltassgn, 198
  WYseed, 199
  WYstratalut, 199
  WYtree, 200
  WYunitarea, 200
  WYunitzonal, 201

* data
  datBarplot, 4
  datBarStacked, 7
  datFilter, 11
  datFreq, 13
  datLineplot, 14
  datLUTclass, 17
  datLUTnm, 19
  datPBplotchg, 21
  datPBPnt2pct, 22
  datPivot, 23
  datPlotcnt, 24
  datSumCond, 26
  datSumTree, 28
  datSumTreeDom, 31
  DBgetEvalid, 38
  DBgetPlots, 41
  DBgetSQLite, 51
  DBgetStrata, 52
  DBgetXY, 55
  DBqryCSV, 58
  modGBarea, 61
  modGBp2veg, 66
  modGBpop, 71
  modGBratio, 76
  modGBtree, 83
  modMAarea, 89
  modMApop, 94
  modMAtree, 99
  modPB, 105
  modPBpop, 111
  modSAarea, 117
  modSApop, 121
  modSATree, 125
  spClipPoint, 141
  spClipPoly, 143
  spClipRast, 145
  spExportSpatial, 148
  spExtractPoly, 149
  spExtractRast, 152
  spGetAuxiliary, 155
  spGetEstUnit, 160
  spGetPlots, 163
  spGetSadoms, 168
  spGetStates, 172
  spGetStrata, 173
  spGetXY, 177
  spImportSpatial, 181
  spMakeSpatialPoints, 182
  spPoly2Rast, 184
  spReprojectVector, 189
INDEX

WYplt, 197
WYpltassgn, 198
WYseed, 199
WYstratalut, 199
WYtree, 200
WYunitarea, 200
WYunitzonal, 201