Package ‘rFIA’

September 17, 2020

Type Package

Title Space-Time Estimation of Forest Variables using the FIA Database

Version 0.2.4

Author Hunter Stanke [aut, cre],
Andrew Finley [aut]

Maintainer Hunter Stanke <stankehu@msu.edu>

Description The goal of 'rFIA' is to increase the accessibility and use of the United States Forest Services (USFS) Forest Inventory and Analysis (FIA) Database by providing a user-friendly, open source toolkit to easily query and analyze FIA Data. Designed to accommodate a wide range of potential user objectives, 'rFIA' simplifies the estimation of forest variables from the FIA Database and allows all R users (experts and newcomers alike) to unlock the flexibility inherent to the Enhanced FIA design. Specifically, 'rFIA' improves accessibility to the spatio-temporal estimation capacity of the FIA Database by producing space-time indexed summaries of forest variables within user-defined population boundaries. Direct integration with other popular R packages (e.g., 'dplyr', 'tidyr', and 'sf') facilitates efficient space-time query and data summary, and supports common data representations and API design. The package implements design-based estimation procedures outlined by Bechtold & Patterson (2005) <doi:10.2737/SRS-GTR-80>, and has been validated against estimates and sampling errors produced by FIA 'EVALIDator'. Current development is focused on the implementation of spatially-enabled model-assisted estimators to improve population, change, and ratio estimates.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.1.0)

Imports dplyr, stringr, sp, sf, tidyr (>= 1.0.0), parallel, ggplot2, gganimate, methods, data.table, bit64, progress, tidyselect (>= 1.0.0), purrr, rlang, dtplyr (>= 1.0.0), R2jags, coda

Suggests knitr, rmarkdown

RoxygenNote 7.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2020-09-17 19:00:03 UTC

R topics documented:

`area` *Estimate land area from FIADB*

Description

Produces estimates of total area (acreage) from FIA data. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by species, size class, and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. grpBy = STATECD).

Usage

```
area(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
     byLandType = FALSE, landType = 'forest',  method = 'TI',
     lambda = .5, treeDomain = NULL, areaDomain = NULL,
     totals = FALSE, variance = FALSE, byPlot = FALSE,
     nCores = 1)
```

Arguments

db	FIA.Database or Remote.FIA.Database object produced from readFIA or getFIA. If a Remote.FIA.Database, data will be read in and processed state-by-state to conserve RAM (see details for an example).
grpBy	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with c(), and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify c(ECOSUBCD,OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of polys object.
returnSpatial	logical; if TRUE, merge population estimates with polys and return as sf multipolygon object. When byPlot = TRUE, return plot-level estimates as sf spatial points.
byLandType	logical; if TRUE, return estimates grouped by individual land type classes ("timberland", "non-timberland forest", "non-forest", and "water").
landType	character, one of: "forest", "non-forest", "census water", "non-census water", "water", or "all"; Type of land that estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
method	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
lambda	numeric (0,1); if method = 'EMA', the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use plotFIA with grp set to lambda to produce moving average ribbon plots. See Stanke et al 2020 for examples.
treeDomain	logical predicates defined in terms of the variables in PLOT, TREE, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. DBH greater than 20 inches: DIA > 20, Dominant/Co-dominant crowns only: CCLCD %in% 2:3). Multiple conditions are combined with & (and) or | (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
areaDomain	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: RDDISTCD %in% 1:6, Hard maple/basswood forest type: FORTYPCD == 805). Multiple conditions are combined with & (and) or | (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
totals	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
variance	logical; if TRUE, return estimated variance (VAR) and sample size (N). If FALSE, return 'sampling error' (SE) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.

`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (proportion of plot in domain of interest; `PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **AREA**: estimate of total area wihtin domain of interest (acres)
- **nPlots**: number of non-zero plots used to compute area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: `https://rfia.netlify.app/`

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. `https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf`

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

`biomass`, `readFIA`, `tpa`

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)

## Most recent estimates of forested area in RI
area(db = fiaRI_mr)


## Same as above grouped by land class
area(db = fiaRI_mr, byLandType = TRUE)

## Estimates for area where stems greater than 20 in DBH occur for
##  available inventories (time-series)
area(db = fiaRI,
     landType = 'forest',
     treeDomain = DIA > 20)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
area(db = fiaRI,
     landType = 'forest',
     treeDomain = DIA > 20,
     nCores =2)

## Return estimates at the plot-level
area(db = fiaRI,
     byPlot = TRUE)
```

`biomass` *Estimate volume, biomass, and carbon stocks from the FIADB*

Description

Produces estimates of volume, biomass, and carbon on a per acre basis from FIA data, along with population estimates for each variable. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by species, size class, and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
biomass(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, bySpecies = FALSE,
        bySizeClass = FALSE, landType = 'forest', treeType = 'live',
        method = 'TI', lambda = .5, treeDomain = NULL, areaDomain = NULL,
        totals = FALSE, variance = FALSE, byPlot = FALSE, nCores = 1)
```

Arguments

db	FIA.Database or Remote.FIA.Database object produced from readFIA or getFIA. If a Remote.FIA.Database, data will be read in and processed state-by-state to conserve RAM (see details for an example).
grpBy	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with c(), and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify c(ECOSUBCD,OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of polys object.
returnSpatial	logical; if TRUE, merge population estimates with polys and return as sf multipolygon object. When byPlot = TRUE, return plot-level estimates as sf spatial points.
bySpecies	logical; if TRUE, returns estimates grouped by species.
bySizeClass	logical; if TRUE, returns estimates grouped by size class (2-inch intervals, see makeClasses to compute different size class intervals).
landType	character ("forest" or "timber"); Type of land that estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
treeType	character ("all", "live", "dead", or "gs""); Type of tree that estimates will be produced for. All (default) includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live or dead (leaning less than 45 degrees), respectively. GS (growing-stock) includes live stems greater than 5 in. DBH which contain at least one 8 ft merchantable log.
method	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
lambda	numeric (0,1); if method = 'EMA', the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use plotFIA with grp set to lambda to produce moving average ribbon plots. See Stanke et al 2020 for examples.
treeDomain	logical predicates defined in terms of the variables in PLOT, TREE, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. DBH greater than 20 inches: DIA > 20, Dominant/Co-dominant crowns only: CCLCD %in% c(2,3)). Multiple conditions are combined with & (and) or | (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.

`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020. Specifically, volume, biomass, and carbon mass per acre are computed using a sample-based ratio-of-means estimator of total volume (carbon or biomass) / total land area within the domain of interest.

Net volume estimates (`NETVOL`) include only the volume of wood in the central stem of a sample tree, from a 1-foot stump to a minimum 4-inch top diameter, or to where the central stem breaks into limbs all of which are greater than 4.0 inches in diameter. Does not include rotten, missing, and form cull portions of the main stem. Saw volume estimates (`SAWVOL`) incldue the net volume in the sawlog portion of the tree, from a 1-foot stump to a 9 inches (hardwood) or 7 inches (softwood) top. All volume estimates are reported in cubic feet (cu. ft. / acre). For estimates in board feet, multiply output values by 12.

Biomass (`BIO`) and carbon (`CARB`) estimates are computed seperately for aboveground (AG) and belowground (BG) stocks, and their totals are the summation of above and belowground stocks. All biomass and carbon estimates are reported in oven-dry mass (short tons). Belowground mass for an individual tree includes modeled estimates for coarse roots (> 0.1"). Above ground mass includes all portions of a tree above the root collar, excluding foliage.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and

"YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (`PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **NETVOL_ACRE**: estimate of mean net volume per acre (cu.ft./acre)
- **SAWVOL_ACRE**: estimate of mean merchantable saw volume per acre (cu.ft./acre)
- **BIO_AG_ACRE**: estimate of mean aboveground biomass per acre (tons/acre)

- **BIO_BG_ACRE**: estimate of mean belowground biomass per acre (tons/acre)
- **BIO_ACRE**: estimate of mean total biomass per acre (tons/acre)
- **CARB_AG_ACRE**: estimate of mean aboveground carbon per acre (tons/acre)
- **CARB_BG_ACRE**: estimate of mean belowground carbon per acre (tons/acre)
- **CARB_ACRE**: estimate of mean total carbon per acre (tons/acre)
- **nPlots_VOL**: number of non-zero plots used to compute volume, biomass, and carbon estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://rfia.netlify.app/

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

`tpa`, `vitalRates`, `growMort`

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


## Most recent estimates for growing-stock on timber land by species
biomass(db = fiaRI_mr,
        landType = 'timber',
        treeType = 'gs')


## Same as above, but at the plot-level
biomass(db = fiaRI_mr,
        landType = 'timber',
        treeType = 'gs',
        byPlot = TRUE)

## Estimates for live white pine ( > 12" DBH) on forested mesic sites (all available inventories)
biomass(fiaRI_mr,
        treeType = 'live',
        treeDomain = SPCD == 129 & DIA > 12, # Species code for white pine
        areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
biomass(db = fiaRI_mr,
        grpBy = STAND_AGE)

## Estimates for snags greater than 20 in DBH on forestland for all
##  available inventories (time-series)
biomass(db = fiaRI,
        landType = 'forest',
        treeType = 'dead',
        treeDomain = DIA > 20)

## Most recent estimates for live stems on forest land by species
biomass(db = fiaRI_mr,
        landType = 'forest',
        treeType = 'live',
        bySpecies = TRUE)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
biomass(db = fiaRI_mr,
        landType = 'forest',
        treeType = 'live',
        bySpecies = TRUE,
        nCores = 2)


## Most recent estimates for all stems on forest land grouped by user-defined areal units
ctSF <- biomass(fiaRI_mr,
                polys = countiesRI,
                returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, BIO_AG_ACRE) # Plot of aboveground biomass per acre
```

carbon *Estimate carbon stocks by IPCC forest carbon pools from the FIADB*

Description

Produces estimates of carbon (tons) on a per acre basis from FIA data, along with population estimates for each variable. Estimates are consistent with those used in the EPA's Greenhouse Gas Inventory Estimates. Can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by species, size class, and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
carbon(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
       byPool = TRUE, byComponent = FALSE, modelSnag = TRUE,
       landType = "forest", method = "TI", lambda = 0.5,
       areaDomain = NULL, totals = FALSE, variance = FALSE,
       byPlot = FALSE, nCores = 1)
```

Arguments

db	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
grpBy	variables from PLOT or COND tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
returnSpatial	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
byPool	logical; if TRUE, return estimates grouped by IPCC forest carbon pools (i.e., aboveground live, belowground live, dead wood, litter, and soil organic).
byComponent	logical; if TRUE, return estimates grouped by IPCC forest carbon components (i.e., aboveground live overstory, aboveground live understory, aboveground live overstory, belowground live overstory, standing dead wood, down dead wood, litter, and soil organic).
modelSnag	logical; if TRUE, return modeled estimates of standing dead wood (i.e., snag) carbon (not a direct sum of actual dead wood observatiosn). Otherwise use observations (P2) of standing dead wood carbon in estimation.

`landType`	character ("forest" or "timber"); Type of land that estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020. Specifically, carbon mass per acre is computed using a sample-based ratio-of-means estimator of total volume (carbon or biomass) / total land area within the domain of interest.

Estimation of carbon stocks draws on measured (e.g., tree carbon) and modeled attributes (e.g., soil organic carbon). This function is intended to produce estimates consistent with those in the EPA's Greenhouse Gas Inventory Estimates. See the following for more info: http://www.epa.gov/climatechange/ghgemissions/usin

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outermost edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (`PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **CARB_ACRE**: estimate of mean total carbon per acre (tons/acre)

- **nPlots_TREE**: number of non-zero plots used to compute carbon estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: `https://rfia.netlify.app/`

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. `https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf`

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

`biomass`, `dwm`

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


## Most recent estimates of carbon by IPCC pool
carbon(db = fiaRI_mr)


## Same as above, at the plot-level
carbon(db = fiaRI_mr,
       byPlot = TRUE)

## Most recent estimates of carbon by IPCC component
carbon(db = fiaRI_mr, byComponent = TRUE)
```

```
## Most recent estimates of total carbon (i.e., all pools)
carbon(db = fiaRI_mr, byPool = FALSE)

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
carbon(db = fiaRI_mr,
       grpBy = STAND_AGE)


## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
carbon(db = fiaRI_mr,
       grpBy = STAND_AGE,
       nCores = 2)


## Most recent estimates for all stems on forest land grouped by user-defined areal units
ctSF <- carbon(fiaRI_mr,
               byPool = FALSE,
               polys = countiesRI,
               totals = TRUE,
               returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, CARB_TOTAL) # Plot of aboveground biomass per acre
```

---

clipFIA *Spatial and temporal queries for FIADB*

Description

Performs space-time queries on Forest Inventory and Analysis Database (FIADB). Subset database to include only data associated with particular inventory years (i.e., most recent), and/or only data within a user-defined region.

Usage

```
clipFIA(db, mostRecent = TRUE, mask = NULL, matchEval = FALSE,
        evalid = NULL, designCD = NULL, nCores = 1)
```

Arguments

db	FIA.Database or Remote.FIA.Database object produced from readFIA or getFIA. If a Remote.FIA.Database, data will be read in and processed state-by-state to conserve RAM (see details for an example).
mostRecent	logical; if TRUE, returns only data for most recent inventory.

`mask`	sp or sf Polygon/MultiPolgyon object; defines the boundaries of spatial intersection with FIA tables.
`matchEval`	logical; if TRUE, returns subset of data for which there are matching reporting years across states. Only useful if db contains mulitple state subsets of the FIA database.
`evalid`	character; unique value which identifies an inventory year and inventory type for a state. If you would like to subset data for an inventory year other than the most recent, use `findEVALID` to look locate this value (see Examples below).
`designCD`	character vector; plot designs to include. Default includes standard national plot design with other similar sampling designs. See FIA Database User Guide Appendix 1 for descriptions of plot designs (see References).
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Not required to run other **rFIA** functions, but may help conserve free memory and reduce processing time if user is interested in producing estimates for a specific inventory year or within a region not explicitly described in the database (w/in user defined polygons).

Spatial intersections do not adhere strictly to absolute plot locations, all plots which fall within an estimation unit (often a county) which intersects with a user defined region will be returned. The plots which fall slightly outside of the region do NOT bias estimates (removed from computations), but as FIA often employs stratified random sampling estimators, all plots within intersecting estimation units must be present to proudce unbiased variance estimates.

If specifying spatio-temporal intersections on a `"Remote.FIA.Database"`, evaluation will occur state-by-state once called by an estimator function.

Value

List object containing spatially intersected FIADB tables.

Author(s)

Hunter Stanke and Andrew Finley

References

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

See Also

`findEVALID`

Examples

```
## Load data from rFIA package
data(fiaRI)
```

```
## Most recent inventory
clipFIA(fiaRI, mostRecent = TRUE)


## Only plots w/in estimation units w/in a user defined polygon
clipFIA(fiaRI, mask = countiesRI[1,], mostRecent = FALSE)
```

---

countiesRI *County boundaries of Rhode Island*

Description

`sp` `SpatialPolygonsDataFrame` representing county boundaries in the state of Rhode Island. Specify `countiesRI` as the `polys` argument with `fiaRI` as the `db` argument in any `rFIA` function to produce estimates summarized by these areal units within the state of Rhode Island.

Usage

```
data("countiesRI")
```

Format

Formal class SpatialPolygonsDataFrame

Examples

```
data(countiesRI)
```

---

diversity *Estimate diversity from FIADB*

Description

Produces estimates of diversity from FIA data. Returns shannon's index, shannon's equitability, and richness for alpha (mean/SE of stands), beta, and gamma diversity. Default behavior estimates species diversity, using `TPA` as a state variable and `SPCD` to groups of individuals. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by size class and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
diversity(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, bySizeClass = FALSE,
          landType = 'forest', treeType = 'live', method = 'TI', lambda = .5,
          stateVar = TPA_UNADJ, grpVar = SPCD, treeDomain = NULL,
          areaDomain = NULL, byPlot = FALSE, totals = FALSE, variance = FALSE,
          nCores = 1)
```

Arguments

`db`	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
`grpBy`	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
`bySizeClass`	logical; if TRUE, returns estimates grouped by size class (default 2-inch intervals, see `makeClasses` to compute other size class intervals).
`landType`	character ('forest' or 'timber'); Type of land which estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`treeType`	character ("all", "live", "dead", or "gs"); Type of tree that estimates will be produced for. All (default) includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live or dead (leaning less than 45 degrees), respectively. GS (growing-stock) includes live stems greater than 5 in. DBH which contain at least one 8 ft merchantable log.
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`stateVar`	variable from TREE table to use as state variable (NOT quoted). Default, `TPA_UNADJ`. Try, `DRYBIO_AG` for aboveground biomass, `pi*(DIA/2)^2` for basal area, or others.

`grpVar`	factor, variable from TREE table to define individual groups (NOT quoted). Default, `SPCD`. Try, `SPGRPCD` for species group, `makeClasses(db$TREE$DIA,interval = 2)` for diameter class, or others.
`treeDomain`	logical predicates defined in terms of the variables in PLOT, TREE, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. DBH greater than 20 inches: `DIA > 20`, Dominant/Co-dominant crowns only: `CCLCD %in% c(2,3))`. Multiple conditions are combined with `&` (and) or `|` (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020. Procedures for computing diversity indices are outlined in Hill (1973) and Shannon (1948).

Alpha-level indices are computed as the mean diversity of a stand. Specifically, alpha diversity is estimated using a sample-based ratio-of-means estimator of stand diversity (e.g. Richness) * land area of stand / total land area within the domain of interest. Thus estimates of alpha diversity within a stand are weighted by the area that stand represents. Gamma-level diversity is computed as a regional index, pooling all plot data together. Beta diversity is computed as gamma diversity - alpha diversity, and thus represents the excess of regional diversity with respect to local diversity.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outermost edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (`PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **H_a**: mean Shannon's Diversity Index, alpha (stand) level
- **H_b**: Shannon's Diversity Index, beta (landscape) level

- **H_g**: Shannon's Diversity Index, gamma (regional) level
- **Eh_a**: mean Shannon's Equitability Index, alpha (stand) level
- **Eh_b**: Shannon's Equitability Index, beta (landscape) level
- **Eh_g**: Shannon's Equitability Index, alpha (stand) level
- **S_a**: mean Species Richness, alpha (stand) level
- **S_b**: Species Richness, beta (landscape) level
- **S_g**: Species Richness, gamma (regional) level
- **nStands**: number of stands with non-zero plots used to compute alpha diversity estimates

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://rfia.netlify.app/

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

Analysis of ecological communities. (2002). United States: M G M SOFTWARE DESIGN (OR).

Hill, M. O. (1973). Diversity and Evenness: A Unifying Notation and Its Consequences. Ecology, 54(2), 427-432. doi:10.2307/1934352.

Shannon, C. E. (1948). A Mathematical Theory of Communication. Bell System Technical Journal, 27(3), 379-423. doi:10.1002/j.1538-7305.1948.tb01338.x.

See Also

tpa, standStruct, invasive

Examples

```
## Load data from rFIA package
data(fiaRI)
data(countiesRI)

## Make a most recent subset
fiaRI_mr <- clipFIA(fiaRI)

## Most recent estimates for live stems on forest land
diversity(db = fiaRI_mr,
          landType = 'forest',
          treeType = 'live')


## Same as above at the plot-level
diversity(db = fiaRI_mr,
          landType = 'forest',
          treeType = 'live',
          byPlot = TRUE)

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
diversity(db = fiaRI_mr,
          grpBy = STAND_AGE)

## Estimates for live white pine ( > 12" DBH) on forested mesic sites (all available inventories)
diversity(fiaRI,
          treeType = 'live',
          treeDomain = DIA > 12,
          areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes

## Most recent estimates for growing-stock on timber land by species
diversity(db = fiaRI_mr,
          landType = 'timber',
          treeType = 'gs',
          bySizeClass = TRUE)

## Same as above, implemented in parallel
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
diversity(db = fiaRI_mr,
          landType = 'timber',
          treeType = 'gs',
          bySizeClass = TRUE,
          nCores = 2)

## Most recent estimates for all stems on forest land grouped by user-defined areal units
ctSF <- diversity(clipFIA(fiaRI, mostRecent = TRUE),
                  polys = countiesRI,
                  returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, H_a) # Plot of mean Shannons Index of stands
```

---

dwm — *Estimate volume, biomass, and carbon stocks of down woody material (fuels) from FIADB*

Description

Produces estimates of down woody material stocks on a per acre basis from the Forest Inventory and Analysis Database (FIADB), along with population totals for each variable. Estimates are returned by fuel class (duff, litter, 1HR, 10HR, 100HR, 1000HR, piles) for application in fuels management. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
dwm(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, landType = 'forest',
    method = 'TI', lambda = .5, areaDomain = NULL, totals = FALSE,
    variance = FALSE, byPlot = FALSE, tidy = TRUE, nCores = 1)
```

Arguments

db	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
grpBy	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
returnSpatial	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
landType	character ("forest" or "timber""); Type of land that estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
method	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
lambda	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
areaDomain	Logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.

`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`tidy`	logical; if TRUE, returns estimates grouped by fuel type, rather than including individual columns for each fuel type (For use in tidyverse packages, e.g. `ggplot2`, `dplyr`). Not recommended when returning spatial objects (`returnSpatial = TRUE`), for consistency with shapefile data structures.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020. Specifically, per acre estimates are computed using a sample-based ratio-of-means estimator of total volume (biomass or carbon) / total land area within the domain of interest.

As defined by FIA, down woody material includes dead organic materials (resulting from plant mortality and leaf turnover) and fuel complexes of live shrubs and herbs. To maintain relevance for forest fuels management, we report estimates grouped by fuel lag-time classes. Specifically, we report estimates for 1HR fuels (small, fine woody debris), 10HR fuels (medium, fine woody debris), 100HR fuels (large, fine woody debris), 1000HR fuels (coarse woody debris), and slash piles, in addition to duff (O horizon; all unidentifiable organic material above mineral soil, beneath litter) and litter (identifiable plant material which is downed and smaller than 10HR fuel class (1HR class includes standing herbaceous material). See Woodall and Monleon (2007) for definitions of fuel lag-time classes and for details on sampling and estimation procedures.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (`PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **VOL_ACRE**: estimate of mean volume per acre of dwm (cu.ft/acre)
- **BIO_ACRE**: estimate of mean biomass per acre of dwm (tons/acre)
- **CARB_ACRE**: estimate of mean carbon mass per acre of dwm (tons/acre)
- **nPlots**: number of non-zero plots used to compute estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: `https://rfia.netlify.app/`

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. `https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf`

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

Woodall, C.; Monleon, V.J., eds. 2007. Sampling Protocol, Estimation, and Analysis Procedures for the Down Woody Materials Indicator of the FIA Program. Gen. Tech. Rep. NRS - 22. ewtown Square, PA: U.S. Department of Agriculture, Forest Service, Northern Research Station. `https://www.nrs.fs.fed.us/pubs/gtr/gtr_nrs22.pdf`

See Also

`tpa`, `biomass`

Examples

```
## Load data from rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)

## Most recent estimates
dwm(fiaRI_mr)


## Same as above at the plot-level
## Most recent estimates
dwm(fiaRI_mr, byPlot = TRUE)

## Estimates of all forestland, over time
dwm(fiaRI)

## Same as above, but output contains seperate column for each structural stage,
##   rathern than grouping variable
dwm(fiaRI,
    tidy = FALSE)

## Estimates of all forestland on mesic sites (most recent)
dwm(fiaRI_mr,
    areaDomain = PHYSCLCD %in% 21:29)

## Estimates of all forestland by owner group (most recent subset)
dwm(fiaRI_mr,
    grpBy = OWNGRPCD)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
dwm(fiaRI_mr,
    tidy = FALSE,
    nCores = 2)

## Estimates of all forestland by county and return
#   return spatial object
dwmSF <- dwm(fiaRI_mr,
             polys = countiesRI,
             returnSpatial = TRUE,
             tidy = FALSE)
plot(dwmSF)
plotFIA(dwmSF, BIO_ACRE) # TOTAL BIOMASS / ACRE (tons)
```

fiaRI *FIADB for Rhode Island 2013 - 2018*

Description

Subset of the Forest Inventory and Analysis Database for the state of Rhode Island. Reporting years range from 2013 - 2018. Specify `fiaRI` as the db argument in any `rFIA` function to produce estimates for the state fo Rhode Island.

Download other subsets of the FIA Database from the FIA Datamart: https://apps.fs.usda.gov/fia/datamart/datamart.html. Once downloaded, unzip the directory, and read into R using `readFIA`.

Usage

```
data("fiaRI")
```

Format

—- FIA Database Object —– Reporting Years: 2013 2014 2015 2016 2017 2018 States: RHODE ISLAND Total Plots: 769 Memory Used: 19.5 Mb Tables: COND COND_DWM_CALC INVASIVE_SUBPLOT_SPP PLOT POP_ESTN_UNIT POP_EVAL POP_EVAL_GRP POP_EVAL_TYP POP_PLOT_STRATUM_ASSGN POP_STRATUM SUBPLOT TREE TREE_GRM_COMPONENT TREE_GRM_ESTN SUBP_COND_CHNG_MTRX

Examples

```
data(fiaRI)
summary(fiaRI)
print(fiaRI)
```

`findEVALID` *Find EVALIDs used in the FIADB*

Description

Lookup Evaluation IDs (EVALIDs) associated with reporting years and evaluation types used in the Forest Inventory and Analysis Database. NOT required to run other **rFIA** functions. Only use if you are interested in subsetting an `FIA.Database` object for a specific reporting year or evaluation type using `clipFIA`.

Usage

```
findEVALID(db, mostRecent = FALSE, state = NULL, year = NULL, type = NULL)
```

Arguments

`db`	list; FIA Database object produced from `readFIA`.
`mostRecent`	logical; if TRUE, returns EVALIDs associated with most recent inventory.
`state`	character vector containing full names of states of interest (e.g. `c('Michigan','Minnesota','Wisconsi`
`year`	numeric vector containing years of interest (e.g. `c(2015,2016,2017)`)
`type`	character ('ALL', 'CURR', 'VOL', 'GROW', 'MORT', 'REMV', 'CHANGE', 'DWM', 'REGEN'). See Reference Population Evaluation Table Description Type Table in FIADB P2 User Guide (References) for descriptions of evaluation types.

Details

EVALIDs in the FIA Database are used to reference data points associated with particular inventory years and evaluation types within a state (e.g. 2017 Current Volume in Michigan). They are often extraordinarily confusing for those not familiar for the FIA Database. With this in mind, **rFIA** has been designed to eliminate users dependence on identifying and specifying appropriate EVALIDs to produce desired estimates, and we therefore do not recommend users attempt to identify EVALIDs independently.

Any `state` or `year` specified must be present in db to return associated EVALIDS.

Value

A numeric vector containing the EVALIDs associated with states, years, or evaluation types specified.

Author(s)

Hunter Stanke and Andrew Finley

References

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

See Also

clipFIA

Examples

```
## Lookup all EVALIDs in an FIA.Database object
findEVALID(fiaRI)

## Find the most recent EVALIDs
findEVALID(fiaRI, mostRecent = FALSE)
```

---

fsi — *Estimate the Forest Stability Index from the FIADB*

Description

Estimate forest population performance from the FIADB using the Forest Stability Index. This function is experimental, please check back soon for updated documentation.

Usage

```
fsi(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
    bySpecies = FALSE, bySizeClass = FALSE,
    landType = "forest", treeType = "live", method = "sma",
    lambda = 0.5, treeDomain = NULL, areaDomain = NULL,
    totals = TRUE, variance = TRUE, byPlot = FALSE,
    useSeries = FALSE, scaleBy = NULL, betas = NULL,
    returnBetas = FALSE, nCores = 1)
```

Arguments

db	`FIA.Database` object produced from `readFIA`; Function requires that PLOT, TREE, COND, POP_PLOT_STRATUM_ASSGN, POP_ESTN_UNIT, POP_EVAL, POP_STRATUM, POP_EVAL_TYP, POP_EVAL_GRP tables exist in FIA.Database object.
grpBy	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.

`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit.
`returnSpatial`	logical; if TRUE, return sf spatial object (polys must also be specified).
`bySpecies`	logical; if TRUE, returns estimates grouped by species.
`bySizeClass`	logical; if TRUE, returns estimates grouped by size class (2-inch intervals, see `makeClasses` to compute different size class intervals).
`landType`	character ('forest' or 'timber'); Type of land which estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`treeType`	character ('all', 'live', 'dead', or 'gs'); Type of tree which estimates will be produced for. All (default) includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live or dead (leaning less than 45 degrees), respectively. GS (growing-stock) includes live stems greater than 5 in. DBH which contain at least one 8 ft merchantable log.
`method`	character; Method used for annual panel combination (see details). One of: 'TI' (temporally indifferent), 'annual' (annual panels), 'SMA' (simple moving average), 'LMA' (linear moving average), or 'EMA' (exponential moving average)
`lambda`	numeric (0,1); if `method == 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots.
`treeDomain`	logical predicates defined in terms of the variables in PLOT, TREE, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. DBH greater than 20 inches: `DIA > 20`, Dominant/Co-dominant crowns only: `CCLCD %in% c(2,3))`. Multiple conditions are combined with `&` (and) or `|` (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return population estimates (e.g. total area, total biomass) along with ratio estimates (e.g. mean biomass per acre).
`variance`	logical; if TRUE, return estimates of uncertainty as variance instead of sampling error (default). Sampling error cannot be used to estimate confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations (population totals not returned).
`useSeries`	logical; If TRUE, use multiple remeasurements to estimate change in TPA & BA on each plot, when available.

`scaleBy`	variables from PLOT or COND tables to use as 'random effects' in model of size-density relationships.Multiple variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(FORTYPCD,SITECLCD)`.
`betas`	more soon.
`returnBetas`	more soon.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Please check back soon for more details

Value

Dataframe or SF object (if `returnSpatial` = `TRUE`). If `byPlot` = `TRUE`, totals are returned for each plot. All variables with names ending in `SE` represent the estimate of sampling error (%) of the variable. All variables with names ending in `TOTAL` represent the population total of the variable.

- **YEAR**: reporting year associated with estimates
- **SI**: estimate of forest stability index
- **SI_STATUS**: indication of the forest stability index (i.e., decline, stable, or expand)
- **SI_INT**: width of 95% confidence interval of mean FSI
- **TPA_RATE**: standardized estimate of annual change in TPA
- **BAA_RATE**: standardized estimate of annual change in BAA

Note

All sampling error estimates are returned as percentages, and represent ~68% confidence (1 standard deviation). To compute sampling error percent with 95% confidence, multiply by 1.96.

Author(s)

Hunter Stanke and Andrew Finley

References

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. `https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf`

See Also

`tpa`, `vitalRates`, `growMort`

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)

## FSI for all forestland 2018 in RI
fsi(fiaRI_mr, method = 'sma')
```

`getFIA` *Download FIA Data and Load into R*

Description

Easiest and most efficient way to access FIA Data in R. Downloads FIA Data from the FIA Datamart, loads the data into R environment, and optionally saves all downloaded tables as .csv files to local directory. Capable of merging multiple state downloads of the FIA database for regional analysis. Requires an internet connection to access and download tables from the FIA Datamart.

Usage

```
getFIA(states, dir = NULL, common = TRUE, tables = NULL,
       load = TRUE, nCores = 1)
```

Arguments

`states`	character; state/ US territory abbreviations (e.g. 'AL', 'MI', etc.) indicating which state subsets to download. Choose to download multiple states by passing character vector of state abbreviations (e.g. `states = c('RI','CT','MA')`). If multiple states specified, tables will be saved as individual state subsets (for future use with `readFIA`, although loaded in R as a merged, regional database.
`dir`	character (optional); directory where FIA tables will be saved after download. If NULL, tables will not be saved on disk and only loaded into R environment.
`common`	logical; if TRUE, only import most commonly used tables, including all required for `rFIA` functions (see Details for list of tables).
`tables`	character vector (optional); names of specific tables to be downloaded for each state specified (e.g. 'PLOT', 'TREE', 'COND', 'TREE_GRM_COMPONENT').
`load`	logical; should downloaded data be loaded into R immediately? If all data is too large to fit in memory, use `load = FALSE` and load the data as a `"Remote.FIA.Database"` with `readFIA`.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

If common = TRUE, the following tables will be loaded: COND, COND_DWM_CALC, INVASIVE_SUBPLOT_SPP, PLOT, POP_ESTN_UNIT, POP_EVAL, POP_EVAL_GRP, POP_EVAL_TYP, POP_PLOT_STRATUM_ASSGN, POP_STRATUM, SUBPLOT, TREE, TREE_GRM_COMPONENT. These tables currently support all functionality with rFIA, and it is recommended that only these tables be imported to conserve RAM and reduce processing time.

If you wish to merge multiple state downloads of FIA data (e.g. Michigan and Indiana state downloads), simply specify multiple state abbreviations to the states argument. Upon import, corresponding tables (e.g. MI_PLOT and IN_PLOT) will be merged, and analysis can be completed for the entire region or within spatial units which transcend state boundaries (e.g. Ecoregion subsections).

If you choose to save downloaded tables to a local directory after download (simply specify dir), these tables can be easily reloaded into R using readFIA (do not need to redownload files).

Easy, efficient parallelization is implemented with the parallel package. Users must only specify the nCores argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (nCores = 1).

Value

List object containing FIA Datatables. List elements represent individual FIA Datatables stored as data.frame objects.

If multiple subsets of the FIA database are downloaded (e.g. states = c('MI','IN')), corresponding tables will be merged (e.g. PLOT table returned contains plots in both Michigan and Indiana).

Author(s)

Hunter Stanke and Andrew Finley

References

FIA DataMart: https://apps.fs.usda.gov/fia/datamart/datamart.html

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

See Also

readFIA

Examples

```
## Download the common tables for Rhode Island, load into R, and save to local directory
## Replace tempDir() with the path to your directory (where data will be saved)
db <- getFIA(states = 'RI', dir = tempdir())
```

growMort — *Estimate recruitment, mortality, and harvest rates from FIADB*

Description

Produces estimates of annual regeneration, recruitment, natural mortality, and harvest rates from the Forest Inventory and Analysis Database (FIADB), along with population estimates for each variable. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by species, size class, and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`). Easy options to implement parallel processing.

Usage

```
growMort(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, bySpecies = FALSE,
         bySizeClass = FALSE, landType = 'forest', treeType = 'all',
         method = 'TI', lambda = .5, stateVar = 'TPA', treeDomain = NULL,
         areaDomain = NULL, totals = FALSE, variance = FALSE,
         byPlot = FALSE, nCores = 1)
```

Arguments

`db`	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
`grpBy`	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
`bySpecies`	logical; if TRUE, returns estimates grouped by species.
`bySizeClass`	logical; if TRUE, returns estimates grouped by size class (2-inch intervals, see `makeClasses` to compute different size class intervals).
`landType`	character ("forest" or "timber""); Type of land that estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).

`treeType`	character ("all", "live", "dead", or "gs"); Type of tree that estimates will be produced for. All (default) includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live or dead (leaning less than 45 degrees), respectively. GS (growing-stock) includes live stems greater than 5 in. DBH which contain at least one 8 ft merchantable log.
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`stateVar`	character; State variable for reporting GRM estimates. One of: TPA, BAA, BIO_AG, BIO_BG, BIO, CARB_AG, CARB_BG, CARB, NETVOL, SAWVOL.
`treeDomain`	logical predicates defined in terms of the variables in PLOT, TREE, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. DBH greater than 20 inches: `DIA > 20`, Dominant/Co-dominant crowns only: `CCLCD %in% c(2,3))`. Multiple conditions are combined with `&` (and) or `|` (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return population estimates (e.g. total area, total mortality) along with ratio estimates (e.g. mean mortality trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Average annual rates are computed using a sample-based ratio of means estimator of total trees subject to an event (e.g. recruitment, mortality) annually / total area. Similarly, the proportion of individuals subject to each event annually is computed as the total trees subject to the event between time 1 and time 2 / total live trees at time 2. All estimates are returned as average annual rates. Only conditions which were forested in time 1 and in time 2 are included in estimates (excluding converted stands).

Recruitment events are defined as when a live stem that is less than 5 inches DBH at time 1, grows to or beyond 5 inches DBH by time 2. This does NOT include stems that grow beyond the 5-inch diameter criteria and are then subject to mortality prior to remeasurement. Natural mortality is defined as when a live stem is subject to non-harvest mortality between successive measurement periods. Finally, harvest is defined as when a live stem is cut and removed between successive measurements.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (`PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **RECR_TPA**: estimate of mean annual recruitment as trees per acre
- **MORT_TPA**: estimate of mean annual mortality as trees per acre
- **REMV_TPA**: estimate of mean annual removals (harvest) as trees per acre
- **RECR_PERC**: estimate of mean percent of individuals subject to recruitment annually
- **MORT_PERC**: estimate of mean percent of individuals subject to mortality annually
- **REMV_PERC**: estimate of mean percent of individuals subject to removal (harvest) annually
- **nPlots_TREE**: number of non-zero plots used to compute total tree estimates
- **nPlots_RECR**: number of non-zero plots used to compute recruitment estimates
- **nPlots_MORT**: number of non-zero plots used to compute mortality estimates
- **nPlots_REMV**: number of non-zero plots used to compute removal estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://rfia.netlify.app/

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. `https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf`

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

`tpa`, `vitalRates`

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


## Most recent estimates for growing-stock on timber land by species
growMort(db = fiaRI_mr,
         landType = 'timber',
         treeType = 'gs')

## Same as above at the plot-level
growMort(db = fiaRI_mr,
         landType = 'timber',
         treeType = 'gs',
         byPlot = TRUE)

## Estimates for white pine ( > 12" DBH) on forested mesic sites
growMort(fiaRI_mr,
         treeType = 'all',
         treeDomain = SPCD == 129 & DIA > 12, # Species code for white pine
         areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
growMort(db = fiaRI_mr,
         grpBy = STAND_AGE)

## Most recent estimates for stems on forest land by species
growMort(db = fiaRI_mr,
         landType = 'forest',
         bySpecies = TRUE)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
```

```
growMort(db = fiaRI_mr,
         landType = 'forest',
         bySpecies = TRUE,
         nCores = 2)


## Most recent estimates for all stems on forest land grouped by user-defined areal units
ctSF <- growMort(fiaRI_mr,
                 polys = countiesRI,
                 returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, MORT_TPA) # Plot of Mortality TPA with color scale
```

---

invasive *Estimate invasive species coverage from FIADB*

Description

Produces estimates of areal coverage of invasive species from the Forest Inventory and Analysis Database. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. All estimates are returned by species although can be grouped by other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
invasive(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, landType = "forest",
         method = 'TI', lambda = .5, areaDomain = NULL, totals = FALSE,
         variance = FALSE, byPlot = FALSE, nCores = 1)
```

Arguments

`db`	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
`grpBy`	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.

`landType`	character ("forest" or "timber"); Type of land that estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Specifically, percent areal coverage is computed using a sample-based ratio-of-means estimator of total invasive coverage area / total land area within the domain of interest. Estimates of areal coverage of individual invasive species should NOT be summed to produce estimates of areal coverage by ALL invasive species, as areal coverage by species is not mutually exclusive (multiple species my occur in the same area). Current FIA data collection protocols do not allow for the unbiased estimation of areal coverage by all invasive species.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outermost edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (proportion of plot in domain of interest; `PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **SYMBOL**: unique species ID from NRCS Plant Reference Guide
- **SCIENTIFIC_NAME**: scientific name of the species
- **COMMON_NAME**: common name of the species
- **COVER_PCT**: estimate of percent areal coverage of the species
- **COVER_AREA**: estimate of areal coverage of the species (acres)
- **AREA**: estimate of total land area (acres)
- **nPlots_INV**: number of non-zero plots used to compute invasive coverage estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://rfia.netlify.app/

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

`dwm`, `vegStruct`

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)
```

```
## Most recent estimates on forest land
invasive(db = fiaRI_mr,
         landType = 'forest')

## Most recent estimates on forest land
invasive(db = fiaRI_mr,
         landType = 'forest',
         byPlot = TRUE)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
invasive(db = fiaRI_mr,
         landType = 'forest',
         nCores = 2)

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
invasive(db = fiaRI_mr,
         grpBy = STAND_AGE)

## Estimates on forested mesic sites (all available inventories)
invasive(fiaRI,
         areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes

## Most recent estimates on forest land grouped by user-defined areal units
ctSF <- invasive(fiaRI_mr,
                 polys = countiesRI,
                 returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF[ctSF$SYMBOL == 'ROMU',], COVER_PCT) # Plot Multiflora rose coverage
```

makeClasses *Convert numeric variables to class intervals (factor)*

Description

Convert continuous numeric variables to class intervals with output as factor or numeric classes. Simplified implementation of `cut`. Example uses include computing diameter or height classes for summarization with **rFIA** functions (e.g. `tpa`, `biomass`).

Usage

```
makeClasses(x, interval = NULL, lower = NULL, upper = NULL,
            brks = NULL, numLabs = FALSE)
```

Arguments

`x`	numeric vector to be converted to factor (class intervals).
`interval`	numeric; interval of desired output classes. e.g. specify `x = DIA` and `interval = 2` for 2-inch diameter class intervals.
`lower`	lower bound of output classes, included in lowest class. e.g. [`lower`, ...).
`upper`	upper bound of output classes, NOT included in highest class. e.g. [..., `upper`).
`brks`	numeric vector of desired breakpoints (bounds) of class intervals.
`numLabs`	logical; if TRUE, return class intervals as numeric vector with values representing the lower bounds of each interval. If FALSE, return factor with labels of form `'[b1,b2)'`.

Value

Factor or integer vector. Factor values represent class intervals with `[b1,b2)` notation, values of integer vectors represent the lower bounds of class intervals (e.g. `b1`).

Author(s)

Hunter Stanke and Andrew Finley

See Also

`clipFIA`

Examples

```
## Load data from the rFIA package
data(fiaRI)

## Compute Diameter Classes on 1-inch intervals for each tree in TREE table ----
# Factor w/ interval labels
makeClasses(fiaRI$TREE$DIA, interval = 1)
# Numeric w/ lower bound of each class as returned value
makeClasses(fiaRI$TREE$DIA, interval = 1, numLabs = TRUE)

## Compute Stand Age Classes on 20 year intervals for each
## condition in COND table ----
# NOTE: Unrecorded stand age recorded as -999, replace negative values with NA
fiaRI$COND$STDAGE[fiaRI$COND$STDAGE < 0] <- NA
makeClasses(fiaRI$COND$STDAGE, interval = 25)

## Compute Stand Stocking Classes (10%) for all live (ALSTK),
## and growing stock (GSSTK) in COND table ----
makeClasses(fiaRI$COND$ALSTK, interval = 10) # All Live
makeClasses(fiaRI$COND$GSSTK, interval = 10) # Growing Stock

## Compute % Slope Classes (20%) for each condition in COND table ----
makeClasses(fiaRI$COND$SLOPE, interval = 20)
```

plotFIA *Static and animated plots of FIA summaries*

Description

Default behavior for non-spatial summaries produces time-series plots, and for spatial summaries (class `sf`) produces choropleth maps. For non-spatial summaries, the user may specify the `grp` parameter to produce plots with multiple lines, colored by a grouping variable. Additionally, users may specify an x-axis to produce plots other than time series (e.g. BAA (`y`) by size class (`x`) colored by species (`grp`)).

Usage

```
plotFIA(data, y = NULL, grp = NULL, x = NULL, animate = FALSE, facet = FALSE,
        se = FALSE,n.max = NULL, plot.title = NULL, y.lab = NULL, x.lab = NULL,
        legend.title = NULL, legend.labs = waiver(), limits = c(NA, NA),
        color.option = 'viridis', line.color = "gray30", line.width =1,
        min.year = 2005, direction = 1, alpha = .9, transform = "identity",
        text.size = 1, text.font = '', lab.width = 1, legend.height = 1,
        legend.width = 1, device = "png", savePath = NULL, fileName = NULL)
```

Arguments

`data`	dataframe, `sf` object, or `FIA.Database` object; FIA summary produced from other `rFIA` functions (e.g. `tpa`, `biomass`, etc.). Also accepts `FIA.Database`, will return map of plot locations.
`y`	variable contained in `data` which will be used as y-axis or to fill polygons (spatial). NOT quoted.
`grp`	variable contained in `data` which will be used as a grouping variable. Not meaningful for spatial summaries. NOT quoted.
`x`	variable contained in `data` which will be used as a x-axis in place of time. If NULL, time-series plot will be produced. Not meaningful for spatial summaries. NOT quoted.
`animate`	logical; if TRUE, produces temporally animated plots.
`facet`	logical; if TRUE, produces temporally grouped plots (stationary).
`se`	logical; if TRUE, plots error bars along with estimates. All error bars represent 68% confidence.
`n.max`	numeric; maximum number of groups to plot. If positive, will plot the top `n` groups with respect to `y`, and if negative, will plot the bottom `n`. Not meaningful for spatial summaries.
`plot.title`	character; plot title.
`y.lab`	character; y-axis label. Not meaningful for spatial summaries.
`x.lab`	character; x-axis label. Not meaningful for spatial summaries.

`legend.title`	character; title for legend.
`legend.labs`	character; labels for legend values.
`limits`	numeric vector of length 2; minumum and maximum of continuous scale for legend.
`color.option`	character; one of: "viridis" (default), "magma", "inferno", "plasma", or "cividis".
`line.color`	character; color of plotted line (non-spatial) or polygon outline color (spatial).
`line.width`	numeric; scalar for plotted line width (non-spatial) polygon outline width (spatial). Specify `lineWidth = 0` for no outline.
`min.year`	numeric; earliest year to be included in animation. FIA data is sparse in years prior to 2005 and estimates are unlikely to be available.
`direction`	numeric; sets the order of colors in the scale. If 1, the default, colors are ordered from darkest to lightest. If -1, the order of colors is reversed.
`alpha`	numeric; alpha transparency, a number in [0,1], see argument alpha in `hsv`.
`transform`	character; transformations to apply to plotted variable `y`. Options include: "asn", "atanh", "boxcox", "exp", "identity", "log", "log10", "log1p", "log2", "logit", "reciprocal", "reverse", "sqrt".
`text.size`	numeric; scalar for text size (e.g. text.size = 2 would be twice the default size).
`text.font`	character; font family. Choose from: 'Short', 'Canonical', 'mono', 'Courier', 'sans', 'Helvetica', 'serif', 'Times', 'AvantGarde', 'Bookman', 'Helvetica-Narrow', 'NewCenturySchoolbook', 'Palatino', 'URWGothic', 'URWBookman', 'NimbusMon', 'URWHelvetica', 'NimbusSan', 'NimbusSanCond', 'CenturySch', 'URWPalladio', 'URWTimes', or 'NimbusRom'.
`lab.width`	numeric; scalar for legend title width. This value controls text wrapping in title.
`legend.height`	numeric; scalar for legend height.
`legend.width`	numeric; scalar for legend width.
`device`	character; device to use for image save. Can either be a device function (e.g. png()), or one of "eps", "ps", "tex" (pictex), "pdf", "jpeg", "tiff", "png", "bmp", "svg" or "wmf" (windows only).
`savePath`	character; path to save plot to (combined with fileName).
`fileName`	character; file name to create on disk.

Details

To produce spatial plots, summaries must be returned as spatial objects (e.g. specify `returnSpatial = TRUE` when computing summaries using `tpa`). For animated plots, also requires that multiple reporting years be present in the summary data (animations iterate through time). For a map of plot locations contained in your `FIA.Database`, specify the object as the `data` argument.

For objects produced with `byPlot = TRUE` and `returnSpatial = TRUE` (spatial point patterns), a categorical grouping variable can be specified to `grp`. Point radii will reflect magnitude of `y` and color will reflect categorical groups (`grp`).

If `animate = FALSE` and multiple reporting years are present in the summary, produces plots of the most recent subset.

Specify `savePath` and `fileName` to save plots (animations saved as .gif files).

Author(s)

Hunter Stanke and Andrew Finley

Examples

```
################### SPATIAL PLOTTING ##############################
## Compute abundance estimates for live stems in Rhode Island
## for all available inventory years, summarized by counties and
## return a spatial object
tpaRI <- tpa(fiaRI, polys = countiesRI, returnSpatial = TRUE)

## Not run:
## Produce animated plot
plotFIA(tpaRI, y = TPA, animate = TRUE, legend.title = 'Abundance (TPA)')
## With a square root transform
plotFIA(tpaRI, y = TPA, animate = TRUE, legend.title = 'Abundance (TPA)', transform = 'sqrt')

## Same as above, but for static plots (most recent subset from RI)
tpaMR <- tpa(clipFIA(fiaRI), polys = countiesRI, returnSpatial = TRUE)
## Produce animated plot
plotFIA(tpaMR, y = TPA, animate = FALSE, plot.title = 'Abundance (TPA)')


################# NON-SPATIAL PLOTTING #########################
## Same as above, but return a non-spatial object (no spatial grouping)
tpaRI <- tpa(fiaRI)

## Plot TPA over time
plotFIA(tpaRI, TPA)

## BAA over time, grouped by ownership group
tpaRI_own <- tpa(fiaRI, grpBy = OWNGRPCD)
plotFIA(tpaRI_own, y = BAA, grp = OWNGRPCD)

## BAA by size class (not a time series) grouped by species
tpaRI_sc <- tpa(clipFIA(fiaRI), bySpecies = TRUE, bySizeClass = TRUE)
plotFIA(tpaRI_sc, y = BAA, grp = COMMON_NAME, x = sizeClass, n.max = 4)# Only the top 4


## End(Not run)
```

readFIA *Load FIA database into R environment from local directory*

Description

Loads FIA Datatables into R from .csv files stored in a local directory. If you have not previously downloaded FIA Data from the FIA Datamart, use getFIA to download data for your region of interest and load it into R. Capable of merging multiple state downloads of the FIA database for regional analysis. Simply store each set of state data, as downloaded from the FIA Datamart, in the same directory and hand to readFIA.

Usage

```
readFIA(dir, common = TRUE, tables = NULL, states = NULL,
        inMemory = TRUE, nCores = 1, ...)
```

Arguments

`dir`	directory where FIA Datatables are stored.
`common`	logical; if TRUE, only import most commonly used tables, including all required for `rFIA` functions (see Details for list of tables).
`tables`	character vector; names of specific tables to be imported (e.g. 'PLOT', 'TREE', 'COND', 'TREE_GRM_COMPONENT').
`states`	character; state/ US territory abbreviations (e.g. 'AL', 'MI', etc.) indicating which state subsets to read. Data for each state must be in `dir`. Choose to read multiple states by passing character vector of state abbreviations (e.g. `states = c('RI','CT','MA')`). If `states = NULL`, data for all states within `dir` will be read in and merged into a regional database.
`inMemory`	logical; should data be stored in-memory? If FALSE, data will be read in state-by-state when an estimator function is called (e.g., `tpa`). This conserves RAM and allows the user to to produce estimates using very large databases that does not all fit in memory at once.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.
`...`	other arguments to pass to `fread`.

Details

Download subsets of the FIA Database using `getFIA` (recommended), or manually from the FIA Datamart: `https://apps.fs.usda.gov/fia/datamart/datamart.html`. Once downloaded, unzip the directory (if downloaded manually), and read into R using `readFIA`.

If `common = TRUE`, the following tables will be imported: COND, COND_DWM_CALC, INVASIVE_SUBPLOT_SPP, PLOT, POP_ESTN_UNIT, POP_EVAL, POP_EVAL_GRP, POP_EVAL_TYP, POP_PLOT_STRATUM_ASSGN, POP_STRATUM, SUBPLOT, TREE, TREE_GRM_COMPONENT. These tables currently support all functionality with `rFIA`, and it is recommended that only these tables be imported to conserve RAM and reduce processing time.

If you wish to merge multiple state downloads of FIA data (e.g. Michigan and Indiana state downloads), simply place both sets of datatables in the same directory (done for you when using `getFIA`) and import with `readFIA`. Upon import, corresponding tables (e.g. MI_PLOT and IN_PLOT) will be merged, and analysis can be completed for the entire region or within spatial units which transcend state boundaries (e.g. Ecoregion subsections).

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Value

List object containing FIA Datatables. List elements represent individual FIA Datatables stored as `data.frame` objects. Names of list elements reflect names of files from which they were read into R environment (File names should not be changed after download from FIA Datamart).

If multiple subsets of the FIA database are held in the same directory (e.g. Michigan and Indiana state downloads), corresponding tables will be merged (e.g. PLOT table returned contains plots in both Michigan and Indiana).

Note

To download subsets of the FIA Database manually, go online to the FIA Datamart (`https://apps.fs.usda.gov/fia/datamart/datamart.html`) and choose to download .csv files. Here you can choose to download subsets of the full database for individual states, or select to download individual tables. For use with the `rFIA` package, we recommend download of subsets of the full database representing individual states of interest. Files must be unzipped in order to be imported.

Alternatively, use `getFIA` to automate the download, reading, and saving process for you (recommended).

Author(s)

Hunter Stanke and Andrew Finley

References

FIA DataMart: `https://apps.fs.usda.gov/fia/datamart/datamart.html`

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

See Also

`clipFIA`, `getFIA`

Examples

```
## Not run: \donttest{
## The following examples shows how you
## can take an existing in-memory FIA.Datbase,
## save it, and read it back in!


## First download the common tables for Rhode Island,
## load into R, but don't save it anywhere yet
db <- getFIA(states = 'RI')



## Now we write it all out
## Replace tempdir() with the path to your
## directory (where data will be saved)
writeFIA(db, dir = tempdir())
## Then read it back in with readFIA
db <- readFIA(dir = tempdir())

}
## End(Not run)
```

---

seedling *Estimate seedling abundance per acre from FIADB*

Description

Produces seedling (< 1 inch DBH) tree per acre (TPA) estimates from FIA data, along with population totals. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by species and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`). Easy options to implement parallel processing.

Usage

```
seedling(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
         bySpecies = FALSE, landType = "forest", method = 'TI',
         lambda = .5, treeDomain = NULL, areaDomain = NULL,
         totals = FALSE, variance = FALSE,
         byPlot = FALSE, nCores = 1)
```

Arguments

`db`	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
`grpBy`	variables from PLOT, COND, or SEEDLING tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
`bySpecies`	logical; if TRUE, returns estimates grouped by species.

`landType`	character ("forest" or "timber"); Type of land that estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`treeDomain`	logical predicates defined in terms of the variables in PLOT, SEEDLING, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. white pine: `SPCD == 129`). Multiple conditions are combined with `&` (and) or `|` (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Specifically, TPA is computed using a sample-based ratio-of-means estimator of total seedlings / total land area within the domain of interest. Percentages of live TPA in the domain of interest are represented as the total number of trees of a particular type (e.g., white pine) / total number of trees (live, all species) within the region. The total populations used to compute these percentages will not change by changing treeType, but will vary if the user specifies an areaDomain or treeDomain.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (`PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in SE, represent the estimate of sampling error (%) of the variable. When variance = TRUE, variables ending in VAR denote the variance of the variable and N is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **TPA**: estimate of mean trees per acre
- **TPA_PERC**: estimate of mean proportion of live trees falling within the domain of interest, with respect to trees per acre
- **nPlots_SEEDLING**: number of non-zero plots used to compute tpa estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use variance = TRUE for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://rfia.netlify.app/

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

tpa, growMort, biomass

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)
```

```
## Most recent estimates on timber land by species
seedling(db = fiaRI_mr,
    landType = 'timber')


## Same as above at the plot-level
seedling(db = fiaRI_mr,
         landType = 'timber',
         byPlot = TRUE)

## Estimates for white pine on forested mesic sites (all available inventories)
seedling(fiaRI_mr,
    treeDomain = SPCD == 129, # Species code for white pine
    areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
seedling(db = fiaRI_mr,
    grpBy = STAND_AGE)

## Most recent estimates for live stems on forest land by species
seedling(db = fiaRI_mr,
    landType = 'forest',
    bySpecies = TRUE)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
seedling(db = fiaRI_mr,
    landType = 'forest',
    bySpecies = TRUE,
    nCores = 2)


## Most recent estimates for all stems on forest land grouped by user-defined areal units
ctSF <- seedling(fiaRI_mr,
            polys = countiesRI,
            returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, TPA) # Plot of TPA with color scale
```

`standStruct` *Estimate forest structural stage distribution from FIADB*

Description

Estimates the stand structural stage distribution of an area of forest/ timberland from FIA data. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. Easy options to implement parallel processing. Stand structural stage is classified for each stand (condition) using a method similar to that of Frelich and Lorimer (1991) but substitute basal area for exposed crown area (see Details, References). If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
standStruct(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
            landType = 'forest', method = 'TI', lambda = .5,
            areaDomain = NULL, totals = FALSE, variance = FALSE,
            byPlot = FALSE, tidy = TRUE, nCores = 1)
```

Arguments

`db`	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
`grpBy`	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
`landType`	character ("forest" or "timber"); Type of land which estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.

`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`tidy`	logical; if TRUE, returns estimates grouped by structural stage, rather than including individual columns for each stand structural stage (For use in tidyverse packages, e.g. `ggplot2`, `dplyr`). Not recommended when returning spatial objects (`returnSpatial = TRUE`), for consistency with shapefile data structures.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Specifically, the percent land area occupied by forest in each stand structural stage are computed using a sample-based ratio-of-means estimator of total area in structural stage / total land area within the domain of interest. Stand structural stage is classified based on the relative basal area of canopy stems in various size classes (defined below). Only stems which are identified on-site dominant, subdominant, or intermdediate crown-classes are used to classify stand structural stage.

Diameter Classes

- *Pole*: 11 - 25.9 cm
- *Mature*: 26 - 45.9 cm
- *Large*: 46+ cm

Structural Stage Classification

- *Pole Stage*: > 67% BA in pole and mature classes, with more BA in pole than mature.
- *Mature Stage*: > 67% BA in pole and mature classes, with more BA in mature than pole OR > 67% BA in mature and large classes, with more BA in mature.
- *Late-Successional Stage*: > 67% BA in mature and large classes, with more in large
- *Mosiac*: Any plot not meeting above criteria.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (structural stage of dominant stand type; `PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **STAGE**: Stand structural stage.
- **PERC**: % land area in each structural stage.

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: `https://rfia.netlify.app/`

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. `https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf`

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

Frelich, L. E., and Lorimer, C. G. (1991). Natural Disturbance Regimes in Hemlock-Hardwood Forests of the Upper Great Lakes Region. Ecological Monographs, 61(2), 145-164. doi:10.2307/1943005

Goodell, L., and Faber-Langendoen, D. (2007). Development of stand structural stage indices to characterize forest condition in Upstate New York. Forest Ecology and Management, 249(3), 158-170. doi:10.1016/j.foreco.2007.04.052

See Also

`tpa`, `diversity`

Examples

```
## Load data from rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)

## Calculate structural stage distribution of all forestland
standStruct(fiaRI_mr)


## Same as above at plot-level (classify stands)
standStruct(fiaRI_mr)

## Same as above, but output contains seperate column for each structural stage,
##   rathern than grouping variable
```

```
standStruct(fiaRI_mr, tidy = FALSE)

## Calculate structural stage distribution of all forestland by owner group, over time
standStruct(fiaRI_mr,
            grpBy = OWNGRPCD)

## Same as above, but implemented in parallel
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
standStruct(fiaRI_mr,
            grpBy = OWNGRPCD,
            nCores = 2)

## Calculate structural stage distribution of all forestland on xeric sites, over time
standStruct(fiaRI_mr,
            areaDomain = PHYSCLCD %in% c(11:19))

## Calculate structural stage distribution of all forestland, over time
standStruct(fiaRI)

## Calculate structural stage distribution of all forestland by county and return
#   return spatial object
sdSF <- standStruct(fiaRI_mr,
            polys = countiesRI,
            returnSpatial = TRUE,
            tidy = FALSE)
plot(sdSF)
plotFIA(sdSF, POLE_PERC)
```

`tpa` *Estimate trees per acre and basal area per acre from FIADB*

Description

Produces tree per acre (TPA) and basal area per acre (BAA) estimates from FIA data, along with population totals for each variable. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by species, size class, and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
tpa(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, bySpecies = FALSE,
    bySizeClass = FALSE, landType = 'forest', treeType = 'live',
     method = 'TI', lambda = .5, treeDomain = NULL, areaDomain = NULL,
     totals = FALSE, variance = FALSE, byPlot = FALSE, nCores = 1)
```

Arguments

`db`	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
`grpBy`	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD, OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
`bySpecies`	logical; if TRUE, returns estimates grouped by species.
`bySizeClass`	logical; if TRUE, returns estimates grouped by size class (2-inch intervals, see `makeClasses` to compute different size class intervals).
`landType`	character ("forest" or "timber"); Type of land which estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`treeType`	character ("all", "live", "dead", or "gs""); Type of tree which estimates will be produced for. All (default) includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live or dead (leaning less than 45 degrees), respectively. GS (growing-stock) includes live stems greater than 5 in. DBH which contain at least one 8 ft merchantable log.
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`treeDomain`	logical predicates defined in terms of the variables in PLOT, TREE, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. DBH greater than 20 inches: `DIA > 20`, Dominant/Co-dominant crowns only: `CCLCD %in% c(2,3))`. Multiple conditions are combined with `&` (and) or `|` (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|`

	(or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Specifically, TPA and BAA are computed using a sample-based ratio-of-means estimator of total trees (BA) / total land area within the domain of interest. Percentages of TPA and BAA in the domain of interest are represented as the total number of trees of a particular type (live, white pine) / total number of trees (live and dead, all species) within the region. The total populations used to compute these percentages will not change by changing treeType, but will vary if the user specifies an areaDomain or treeDomain.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a Remote.FIA.Database. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the parallel package. Users must only specify the nCores argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (nCores = 1).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if returnSpatial = TRUE). If byPlot = TRUE, values are returned for each plot (PLOT_STATUS_CD = 1 when forest exists at the plot location). All variables with names ending in SE, represent the estimate of sampling error (%) of the variable. When variance = TRUE, variables ending in VAR denote the variance of the variable and N is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **TPA**: estimate of mean trees per acre
- **BAA**: estimate of mean basal area (sq. ft.) per acre
- **TPA_PERC**: estimate of mean proportion of trees falling within the domain of interest, with respect to trees per acre
- **BAA_PERC**: estimate of mean proportion of trees falling within the domain of interest, with respect to basal area per acre
- **nPlots_TREE**: number of non-zero plots used to compute tree and basal area estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use variance = TRUE for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://rfia.netlify.app/

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

biomass, growMort, seedling

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


## Most recent estimates for growing-stock on timber land by species
tpa(db = fiaRI_mr,
    landType = 'timber',
    treeType = 'gs')

## Same as above at the plot-level
tpa(db = fiaRI_mr,
    landType = 'timber',
    treeType = 'gs',
    byPlot = TRUE)

## Estimates for live white pine ( > 12" DBH) on forested mesic sites (all available inventories)
tpa(fiaRI_mr,
    treeType = 'live',
    treeDomain = SPCD == 129 & DIA > 12, # Species code for white pine
    areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
tpa(db = fiaRI_mr,
    grpBy = STAND_AGE)

## Estimates for snags greater than 20 in DBH on forestland for all
##  available inventories (time-series)
tpa(db = fiaRI,
    landType = 'forest',
    treeType = 'dead',
    treeDomain = DIA > 20)

## Most recent estimates for live stems on forest land by species
tpa(db = fiaRI_mr,
    landType = 'forest',
    treeType = 'live',
    bySpecies = TRUE)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
tpa(db = fiaRI_mr,
    landType = 'forest',
    treeType = 'live',
    bySpecies = TRUE,
    nCores = 2)


## Most recent estimates for all stems on forest land grouped by user-defined areal units
ctSF <- tpa(fiaRI_mr,
            polys = countiesRI,
            returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, TPA) # Plot of TPA with color scale
```

---

vegStruct *Estimate vegation cover by canopy layer with the FIADB*

Description

Produces estimates of vegetation cover by canopy layer and species growth form from the Forest Inventory and Analysis Database. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. All estimates are returned by species although can be grouped by other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`). Easy options to implement parallel processing.

Usage

```
vegStruct(db, grpBy = NULL, polys = NULL,
            returnSpatial = FALSE, landType = "forest",
            method = "TI", lambda = 0.5,
            areaDomain = NULL, totals = FALSE,
            variance = FALSE, byPlot = FALSE,
            nCores = 1)
```

Arguments

`db`	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).
`grpBy`	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c(ECOSUBCD,OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
`landType`	character ("forest" or "timber"); Type of land which estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).
`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.

`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020. Specifically, percent areal coverage is computed using a sample-based ratio-of-means estimator of total coverage area / total land area within the domain of interest.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (proportion of plot in domain of interest; `PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **LAYER**: canopy layer
- **GROWTH_HABIT**: species growth habit
- **COVER_PCT**: estimate of percent areal coverage of the growth habit within the canopy layer
- **COVER_AREA**: estimate of areal coverage of the growth habit within the canopy layer (acres)
- **AREA**: estimate of total land area (acres)
- **nPlots_VEG**: number of non-zero plots used to compute areal coverage estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://rfia.netlify.app/

FIA Database User Guide: https://www.fia.fs.fed.us/library/database-documentation/

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

invasive, dwm

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


## Estimates across RI for the most recent inventory year
vegStruct(db = fiaRI_mr)


## Return estimates at the plot-level
vegStruct(db = fiaRI,
          byPlot = TRUE)
```

---

vitalRates *Estimate tree growth rates from FIADB*

Description

Computes estimates of average annual DBH, basal area, height, and net volume growth rates for individual stems, along with average annual basal area and net volume growth per acre. Estimates can be produced for regions defined within the FIA Database (e.g. counties), at the plot level, or within user-defined areal units. Options to group estimates by species, size class, and other variables defined in the FIADB. If multiple reporting years (EVALIDs) are included in the data, estimates will be output as a time series. If multiple states are represented by the data, estimates will be output for the full region (all area combined), unless specified otherwise (e.g. `grpBy = STATECD`).

Usage

```
vitalRates(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, bySpecies = FALSE,
          bySizeClass = FALSE, landType = 'forest', treeType = 'live',
          method = 'TI', lambda = .5, treeDomain = NULL,
          areaDomain = NULL, totals = FALSE, variance = FALSE,
          byPlot = FALSE, nCores = 1)
```

Arguments

db	`FIA.Database` or `Remote.FIA.Database` object produced from `readFIA` or `getFIA`. If a `Remote.FIA.Database`, data will be read in and processed state-by-state to conserve RAM (see details for an example).

`grpBy`	variables from PLOT, COND, or TREE tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with `c()`, and grouping will occur heirarchically. For example, to produce seperate estimates for each ownership group within ecoregion subsections, specify `c('ECOSUBCD', 'OWNGRPCD')`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produces for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`returnSpatial`	logical; if TRUE, merge population estimates with `polys` and return as `sf` multipolygon object. When `byPlot = TRUE`, return plot-level estimates as `sf` spatial points.
`bySpecies`	logical; if TRUE, returns estimates grouped by species.
`bySizeClass`	logical; if TRUE, returns estimates grouped by size class (2-inch intervals, see `makeClasses` to compute different size class intervals).
`landType`	character ("forest" or "timber"); Type of land which estimates will be produced for. Timberland is a subset of forestland (default) which has high site potential and non-reserve status (see details).
`treeType`	character ("live" or "gs"); Type of tree which estimates will be produced for. All (default) includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live or dead (leaning less than 45 degrees), respectively. GS (growing-stock) includes live stems greater than 5 in. DBH which contain at least one 8 ft merchantable log.
`method`	character; design-based estimator to use. One of: "TI" (temporally indifferent, default), "annual" (annual), "SMA"" (simple moving average), "LMA" (linear moving average), or "EMA" (exponential moving average). See Stanke et al 2020 for a complete description of these estimators.
`lambda`	numeric (0,1); if `method = 'EMA'`, the decay parameter used to define weighting scheme for annual panels. Low values place higher weight on more recent panels, and vice versa. Specify a vector of values to compute estimates using mulitple wieghting schemes, and use `plotFIA` with `grp` set to `lambda` to produce moving average ribbon plots. See Stanke et al 2020 for examples.
`treeDomain`	logical predicates defined in terms of the variables in PLOT, TREE, and/or COND tables. Used to define the type of trees for which estimates will be produced (e.g. DBH greater than 20 inches: `DIA > 20`, Dominant/Co-dominant crowns only: `CCLCD %in% c(2,3))`. Multiple conditions are combined with `&` (and) or `|` (or). Only trees where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`areaDomain`	logical predicates defined in terms of the variables in PLOT and/or COND tables. Used to define the area for which estimates will be produced (e.g. within 1 mile of improved road: `RDDISTCD %in% c(1:6)`, Hard maple/basswood forest type: `FORTYPCD == 805)`. Multiple conditions are combined with `&` (and) or `|` (or). Only plots within areas where the condition evaluates to TRUE are used in producing estimates. Should NOT be quoted.
`totals`	logical; if TRUE, return total population estimates (e.g. total area) along with ratio estimates (e.g. mean trees per acre).

`variance`	logical; if TRUE, return estimated variance (`VAR`) and sample size (`N`). If FALSE, return 'sampling error' (`SE`) as returned by EVALIDator. Note: sampling error cannot be used to construct confidence intervals.
`byPlot`	logical; if TRUE, returns estimates for individual plot locations instead of population estimates.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

Estimation Details

Estimation of forest variables follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Average annual diameter, basal area, height, and net volume growth of a stem is computed using a sample-based ratio of means estimator of total diameter (basal area, height, net volume) growth / total trees, and average annual basal area and net volume growth per acre is computed as total basal area (net volume) growth / total area. All estimates are returned as average annual rates. Only conditions which were forest in time 1 and in time 2 are included in estimates (excluding converted stands). Only stems 5 inches DBH or greater are included in estimates.

As estimates are of net growth rates, they may attain a negative value. Negative growth estimates most likely indicate a substantial change in an attribute of the tree or area between time 1 and time 2, which caused the attribute to decrease. Implementation of the growth accounting method allows us to more accurately represent shifts in forest attributes (biomass) between classified groups (size classes) over time.

Users may specify alternatives to the 'Temporally Indifferent' estimator using the `method` argument. Alternative design-based estimators include the annual estimator ("ANNUAL"; annual panels, or estimates from plots measured in the same year), simple moving average ("SMA"; combines annual panels with equal weight), linear moving average ("LMA"; combine annual panels with weights that decay *linearly* with time since measurement), and exponential moving average ("EMA"; combine annual panels with weights that decay *exponentially* with time since measurement). The "best" estimator depends entirely on user-objectives, see Stanke et al 2020 for a complete description of these estimators and tradeoffs between precision and temporal specificity.

When `byPlot = FALSE` (i.e., population estimates are returned), the "YEAR" column in the resulting dataframe indicates the final year of the inventory cycle that estimates are produced for. For example, an estimate of current forest area (e.g., 2018) may draw on data collected from 2008-2018, and "YEAR" will be listed as 2018 (consistent with EVALIDator). However, when `byPlot = TRUE` (i.e., plot-level estimates returned), the "YEAR" column denotes the year that each plot was measured (MEASYEAR), which may differ slightly from its associated inventory year (INVYR).

Stratified random sampling techniques are most often employed to compute estimates in recent inventories, although double sampling and simple random sampling may be employed for early inventories. Estimates are adjusted for non-response bias by assuming attributes of non-response plot locations to be equal to the mean of other plots included within thier respective stratum or population.

Working with "Big Data"

If FIA data are too large to hold in memory (e.g., R throws the "cannot allocate vector of size ..." errors), use larger-than-RAM options. See documentation of `link{readFIA}` for examples of how to set up a `Remote.FIA.Database`. As a reference, we have used rFIA's larger-than-RAM methods to estimate forest variables using the entire FIA Database (~50GB) on a standard desktop computer with 16GB of RAM. Check out our website for more details and examples.

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Definition of forestland

Forest land must be at least 10-percent stocked by trees of any size, including land that formerly had such tree cover and that will be naturally or artificially regenerated. Forest land includes transition zones, such as areas between heavily forested and nonforested lands that are at least 10-percent stocked with trees and forest areas adjacent to urban and builtup lands. The minimum area for classification of forest land is 1 acre and 120 feet wide measured stem-to-stem from the outer-most edge. Unimproved roads and trails, streams, and clearings in forest areas are classified as forest if less than 120 feet wide. Timber land is a subset of forest land that is producing or is capable of producing crops of industrial wood and not withdrawn from timber utilization by statute or administrative regulation. (Note: Areas qualifying as timberland are capable of producing at least 20 cubic feet per acre per year of industrial wood in natural stands. Currently inaccessible and inoperable areas are NOT included).

Value

Dataframe or SF object (if `returnSpatial = TRUE`). If `byPlot = TRUE`, values are returned for each plot (`PLOT_STATUS_CD = 1` when forest exists at the plot location). All variables with names ending in `SE`, represent the estimate of sampling error (%) of the variable. When `variance = TRUE`, variables ending in `VAR` denote the variance of the variable and `N` is the total sample size (i.e., including non-zero plots).

- **YEAR**: reporting year associated with estimates
- **DIA_GROW**: estimate of mean annual diameter growth of a stem (inches/ yr)
- **BA_GROW**: estimate of mean annual basal area growth of a stem (sq. ft./ yr)
- **BAA_GROW**: estimate of mean annual basal area growth per acre (sq. ft./ acre/ yr)
- **NETVOL_GROW**: estimate of mean annual sound net volume growth of a stem (cu. ft./ yr)
- **NETVOL_GROW_AC**: estimate of mean annual sound net volume growth per acre (cu. ft./ acre/ yr)
- **BIO_GROW**: estimate of mean annual aboveground biomass growth of a stem (short tons/ yr)
- **BIO_GROW_AC**: estimate of mean annual aboveground biomass growth per acre (short tons/ acre/ yr)
- **nPlots_TREE**: number of non-zero plots used to compute tree and basal area estimates
- **nPlots_AREA**: number of non-zero plots used to compute land area estimates

Note

All sampling error estimates (SE) are returned as the "percent coefficient of variation" (standard deviation / mean * 100) for consistency with EVALIDator. IMPORTANT: sampling error cannot be used to construct confidence intervals. Please use `variance = TRUE` for that (i.e., return variance and sample size instead of sampling error).

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: `https://rfia.netlify.app/`

FIA Database User Guide: `https://www.fia.fs.fed.us/library/database-documentation/`

Bechtold, W.A.; Patterson, P.L., eds. 2005. The Enhanced Forest Inventory and Analysis Program - National Sampling Design and Estimation Procedures. Gen. Tech. Rep. SRS - 80. Asheville, NC: U.S. Department of Agriculture, Forest Service, Southern Research Station. 85 p. `https://www.srs.fs.usda.gov/pubs/gtr/gtr_srs080/gtr_srs080.pdf`

Stanke, H., Finley, A. O., Weed, A. S., Walters, B. F., & Domke, G. M. (2020). rFIA: An R package for estimation of forest attributes with the US Forest Inventory and Analysis database. Environmental Modelling & Software, 127, 104664.

See Also

`growMort`, `tpa`

Examples

```
## Load data from the rFIA package
data(fiaRI)
data(countiesRI)

## Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


## Most recent estimates for growing-stock on timber land by species
vitalRates(db = fiaRI_mr,
           landType = 'timber',
           treeType = 'gs')


## Same as above but at the plot-level
vitalRates(db = fiaRI_mr,
           landType = 'timber',
           treeType = 'gs',
           byPlot = TRUE)

## Estimates for white pine ( > 12" DBH) on forested mesic sites
vitalRates(fiaRI_mr,
           treeType = 'live',
           treeDomain = SPCD == 129 & DIA > 12, # Species code for white pine
           areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes

## Most recent estimates grouped by stand age on forest land
# Make a categorical variable which represents stand age (grouped by 10 yr intervals)
fiaRI_mr$COND$STAND_AGE <- makeClasses(fiaRI_mr$COND$STDAGE, interval = 10)
vitalRates(db = fiaRI_mr,
           grpBy = STAND_AGE)

## Most recent estimates for live stems on forest land by species
vitalRates(db = fiaRI_mr,
           landType = 'forest',
           bySpecies = TRUE)

## Same as above, but implemented in parallel (much quicker)
parallel::detectCores(logical = FALSE) # 4 cores available, we will take 2
vitalRates(db = fiaRI_mr,
           landType = 'forest',
           bySpecies = TRUE,
           nCores = 2)


## Most recent estimates for all stems on forest land grouped by user-defined areal units
ctSF <- vitalRates(fiaRI_mr,
                   polys = countiesRI,
                   returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, BIO_GROW) # Plot of individual tree biomass growth rates
```

writeFIA *Write FIA tables to local directory*

Description

Write FIA.Database object to local directory as a series of .csv files representing each table. Most useful for writing merged states and temporal/spatial subsets of the database. Once written as .csv, files can be reloaded into R with readFIA.

Usage

```
writeFIA(db, dir, byState = FALSE, nCores = 1, ...)
```

Arguments

db	FIA.Database object produced from readFIA or getFIA.
dir	directory where FIA Datatables will be stored.

`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.
`byState`	logical; should tables be written out by state? Must be TRUE if planning to load data as an out-of-memory database in the future (see `readFIA`).
`...`	other arguments to pass to `fwrite`.

Details

Easy, efficient parallelization is implemented with the `parallel` package. Users must only specify the `nCores` argument with a value greater than 1 in order to implement parallel processing on their machines. Parallel implementation is achieved using a snow type cluster on any Windows OS, and with multicore forking on any Unix OS (Linux, Mac). Implementing parallel processing may substantially decrease decrease free memory during processing, particularly on Windows OS. Thus, users should be cautious when running in parallel, and consider implementing serial processing for this task if computational resources are limited (`nCores = 1`).

Author(s)

Hunter Stanke and Andrew Finley

See Also

`readFIA`, `getFIA`

Examples

```
## Write the 'fiaRI' object to a temporary directory
## Replace temp_dir with the path to your directory (where data will be saved)
temp_dir = tempdir()
writeFIA(fiaRI, dir = temp_dir)
```

Index