Package 'rFIA'

September 29, 2025

Type Package

Title Estimation of Forest Variables using the FIA Database

Version 1.1.2

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

Maintainer Jeffrey Doser <jwdoser@ncsu.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 spatial-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 and model-based estimators to improve population, change, and ratio estimates.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports dplyr (>= 1.0.0), dtplyr (>= 1.0.0), tidyr (>= 1.0.0), stringr, sf, parallel, methods, data.table, bit64, tidyselect (>= 1.0.0), rlang, ggplot2

Suggests knitr, rmarkdown, gganimate, R2jags, coda, testthat

RoxygenNote 7.1.2

URL https://github.com/doserjef/rFIA,
https://www.doserlab.com/files/rFIA

BugReports https://github.com/doserjef/rFIA/issues

NeedsCompilation no

Repository CRAN

Date/Publication 2025-09-29 11:20:02 UTC

Contents

area . . . 3
areaChange . . . 7
biomass . . . 12
carbon . . . 19
clipFIA . . . 24
countiesRI . . . 26
customPSE . . . 27
diversity . . . 30
dwm . . . 35
fiaRI . . . 40
findEVALID . . . 41
fsi . . . 42
getDesignInfo . . . 47
getFIA . . . 50
growMort . . . 52
intersectFIA . . . 57
invasive . . . 59
makeClasses . . . 63
plotFIA . . . 64
readFIA . . . 67
seedling . . . 70
standStruct . . . 74
tpa . . . 79
vegStruct . . . 84
vitalRates . . . 88
volume . . . 94
writeFIA . . . 99

Index **101**

area *Estimate forest land area from FIADB*

Description

Produces estimates of total forest 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 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 = 0.5, treeDomain = NULL, areaDomain = NULL,
     totals = TRUE, variance = FALSE, byPlot = FALSE,
     condList = 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, PLOTGEOM, 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 methods of stand regeneration, specify c(STDORGCD, OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced for regions 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", "timber", "non-forest", "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 plots with at least one tree 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.
condList	logical; if TRUE, returns condition-level summaries intended for subsequent use with customPSE().
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. Area percentages in the domain of interest are represented as the total number of forest plots containing trees of a particular type (live, white pine) / total number of forest plots within the region and area domain. The total populations (e.g., the denominator of the ratio estimate) used to compute these percentages will not change by changing treeDomain. Instead, specifying treeDomain will change the specific plots used to determine the total area (e.g., the numerator of the ratio estimate) that meets the given tree requirements. The total population will change if the user specifies an areaDomain or if changing landType.

Note that when using grpBy with a tree-level parameter (e.g., SPCD), the total area across all groups will NOT be equal to the total area of the given land type. This is because the groups are not mutually exclusive (e.g., a plot can contain more than one species and thus be counted in the area calculations for multiple SPCDs). If specifying grpBy with a PLOT or COND level variable (e.g., forest type [FORTYPCD]), the groups are mutually exclusive and the area across all groups will sum to the total area of the given land type. When specifying grpBy or treeDomain, percentages

are calculated relative to the total amount of land area specified by `landType`. For example, if setting `treeDomain = SPCD == 121` and `landType = 'forest'`, the area percentage returned will represent the estimated percentage of forest land that contains at least one longleaf pine (`SPCD == 121`) tree.

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 `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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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
- **PERC_AREA**: percent of area within the domain of interest
- **AREA_TOTAL**: estimate of total area within domain of interest (acres)
- **nPlots_AREA_NUM**: number of non-zero plots used to compute land area estimates within the domain of interest
- **nPlots_AREA_DEN**: 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, Andrew Finley, Jeffrey W. Doser

References

rFIA website: https://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

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)
# area(db = fiaRI,
#      landType = 'forest',
#      treeDomain = DIA > 20,
#      nCores =2)

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

---

areaChange *Estimate land area change from the FIADB*

Description

Produces estimates of annual net and component change in land 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 land type 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

```
areaChange(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
```

```
byLandType = FALSE, landType = "forest", method = "TI",
lambda = 0.5, treeDomain = NULL, areaDomain = NULL,
variance = FALSE, byPlot = FALSE,
condList = FALSE, chngType = 'net', 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, PLOTGEOM, or COND tables to group estimates by (NOT quoted). Multiple grouping variables should be combined with c(), and grouping will occur hierarchically. For example, to produce separate estimates for each ownership group within methods of stand regeneration, specify c(STDORGCD, OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate 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", "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 weighting 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.

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.
condList	logical; if TRUE, returns condition-level summaries intended for subsequent use with customPSE().
chngType	character, one of "net" or "component"; if "net", produce estimates of net change in land area, and if "component", produce estimates of component change in land area (i.e., showing all shifts across classified attributes).
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.

Estimates are returned in terms of *net* annual change in land area by default, however users may choose to estimate components of land area change by setting chngType = 'component'. For example, imagine we are interested in estimating the change in forestland area across a region. During our sampling period, 4000 acres of forestland was *diverted* to non-forest and an additional 6000 acres of non-forest *reverted* to forestland. rFIA considers these shifts in land classifications change *components*, and hence these point estimates would be returned when chngType = 'component'. However, we are often interested in *net* change in land area, rather than individual components. Here net change in forestland area is +2000 acres (6000-4000) and represents the net result of diversion and reversion processes in the region over our study period.

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 `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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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
- **PERC_CHNG**: estimate of annual percent change in land area within domain of interest (% of previous)
- **AREA_CHNG**: estimate of annual change in land area within domain of interest (acres)
- **PREV_AREA**: estimate of total land area within domain of interest at first measurement (acres)
- **nPlots**: number of non-zero plots used to compute area change estimates

Importantly, when `chngType = 'component'`, individual change components will be returned. If no grouping variables are specified in `grpBy`, results will be grouped by variables named **STATUS1**

and **STATUS2**, indicating the land classification at first and second measurements, respectively. Otherwise, if `grpBy` is specified, change components will be estimated for all shifts in land area across classified attributes represented by the variables (first and second measurements again denoted by the suffix 1 and 2). This is also the case for when additional criteria are specified to the tree, condition, or area domains using `treeDomain` and/or `areaDomain`.

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, Andrew Finley, Jeffrey W. Doser

References

rFIA website: https://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

Examples

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

# Most recents subset
fiaRI_mr <- clipFIA(fiaRI)

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


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

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

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

---

biomass *Estimate tree biomass and carbon stocks from the FIADB*

Description

Produces estimates of tree 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, byComponent = FALSE,
        landType = "forest", treeType = "live", method = "TI",
        lambda = 0.5, treeDomain = NULL, areaDomain = NULL,
        totals = FALSE, variance = FALSE, byPlot = FALSE,
        treeList = FALSE, component = "AG",
        bioMethod = "NSVB", 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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).

byComponent — logical; if TRUE, returns estimates grouped by the following biomass components: stem, stem bark, branches, foliage, stump, stump bark, merchantable bole, merchantable bole bark, sawlog, sawlog bark, and belowground roots.

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 which estimates will be produced for. All includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live (default) 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.

treeList — logical; if TRUE, returns tree-level summaries intended for subsequent use with `customPSE()`.

component	character, combination of: "TOTAL" (sum of all components), "AG" (aboveground components excluding foliage), "STEM" (total stem of timber species from ground line to the tree tip), "STEM_BARK", "BRANCH", (branch/limbs of timber species), "FOLIAGE" (foliage for live trees at least 1.0 inches dbh/drc), "STUMP", "STUMP_BARK", "BOLE" (merchantable bole), "BOLE_BARK", "SAWLOG", "SAWLOG_BARK", "ROOT" (beloground portion of tree including coarse roots with a root diameter at elast 0.1 inch); biomass component to use in estimation. Note that "TOTAL" includes foliage for biomass estimates but does not include foliage for carbon estimates. See Details below for more detailed descriptions of the components.
bioMethod	character; tree-level biomass estimation procedures to use. As of rFIA v1.1.0, the only biomass estimation procedure supported is the National Scale Volume and Biomass ("NSVB") models. See Estimation Details for references.
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, tree biomass and carbon per acre are computed using a sample-based ratio-of-means estimator of total biomass / total land area within the domain of interest.

A sum of aboveground biomass components, excluding foliage, is estimated by default (component = 'AG'). However, users may specify unique combinations of biomass components if they wish to do so. For example, to estimate aboveground biomass, including foliage, specify component = c('AG', 'FOLIAGE') in the call to biomass. To estimate all biomass components simultaneously (i.e., grouped by copmonent), specify byComponent = TRUE. All biomass components are computed using the National Scale Volume and Biomass (NSVB) approach adopted by FIA in September 2023. See Westfall et al. 2024 for more detailed information on NSVB procedures. The following biomass components can be estimated using biomass() (note the components are not mutually exclusive):

- *TOTAL*: total biomass, which is equivalent to the sum of the following components: "ROOT", "STEM", "STEM_BARK", "BRANCH", and "FOLIAGE". *NOTE*: total carbon estimates do not include estimates of carbon in foliage.
- *AG*: aboveground biomass/carbon, not including foliage. This is equivalent to the sum of the following components: "STEM", "STEM_BARK", and "BRANCH".
- *STEM*: oven-dry biomass/carbon of wood in the total stem of timber species (trees where diameter is measured at breast height) with dbh at least 1.0 inches, from ground line to the tree tip. Calculated for live and standing dead trees.
- *STEM_BARK*: oven-dry biomass/carbon of bark in the total stem of timber species with dbh at least 1.0 inches, from ground line to the tree tip. Calculated for live and standing dead trees.
- *BRANCH*: oven-dry biomass/carbon of wood and bark in the branches/limbs of timber species with at least 1.0 inches. This only includes branches; it does not include any portion of the total stem. Calculated for live and standing dead trees. For live trees, this value is reduced for broken tops. For standing dead trees, this value is reduced for broken tops as well as decay.

- *FOLIAGE*: oven-dry biomass of foliage for live trees with dbh/drc at least 1.0 inches. *NOTE*: foliar carbon is not calculated and is instead set to 0.
- *STUMP*: oven-dry biomass/carbon of wood in the stump of timber species with dbh at least 5.0 inches. The stump is that portion of the tree from the ground line to the bottom of the merchantable bole (i.e., below 1 foot). Calculated for live and standing dead trees.
- *STUMP_BARK*:oven-dry biomass/carbon of bark in the stump of timber species with dbh at least 5.0 inches. The stump is that portion of the tree from the ground line to the bottom of the merchantable bole (i.e., below 1 foot). Calculated for live and standing dead trees.
- *BOLE*: oven-dry biomass/carbon of wood in the merchantable bole of timber species with dbh at least 5.0 inches, from a 1-foot stump to a minimum 4-inch top diameter. Calculated for live and standing dead trees.
- *BOLE_BARK*: oven-dry biomass/carbon of bark in the merchantable bole of timber species with dbh at least 5.0 inches, from a 1-foot stump to a minimum 4-inch top diameter. Calculated for live and standing dead trees.
- *SAWLOG*: the oven-dry biomass/carbon of wood in the sawlog portion of timber species of sawtimber size from a 1-foot stump to a minimum top diameter or to where the central stem breaks into limbs, all of which are less than the minimum top diamter. Minimum dbh is 9.0 inches for softwoods and 11.0 inches for hardwoods. The minimum top diameter is 7.0 inches for softwoods and 9.0 inches for hardwoods.
- *SAWLOG_BARK*: the oven-dry biomass/carbon of bark in the sawlog portion of timber species of sawtimber size from a 1-foot stump to a minimum top diameter or to where the central stem breaks into limbs, all of which are less than the minimum top diamter. Minimum dbh is 9.0 inches for softwoods and 11.0 inches for hardwoods. The minimum top diameter is 7.0 inches for softwoods and 9.0 inches for hardwoods.
- *ROOT*: oven-dry biomass of the below ground portion of a tree, including coarse roots with a root diameter of at least 0.1 inches. This is a modeled estimate, calculated for live and standing dead trees with dbh/drc at least 1.0 inches. This component, unlike all other compoments, is estimated using the Component Ratio Method (CRM).

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 `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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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
- **BIO_ACRE**: estimate of mean tree biomass per acre (short tons/acre)
- **CARB_ACRE**: estimate of mean tree carbon per acre (short tons/acre)
- **nPlots_TREE**: number of non-zero plots used to compute 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, Andrew Finley, Jeffrey W. Doser

References

rFIA website: https://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

Westfall, James A., Coulston, John W., Gray, Andrew N., Shaw, John D., Radtke, Philip J., Walker, David M., Weiskittel, Aaron R., MacFarlane, David W., Affleck, David L.R., Zhao, Dehai, Temesgen, Hailemariam, Poudel, Krishna P., Frank, Jereme M., Prisley, Stephen P., Wang, Yingfang, Sánchez Meador, Andrew J., Auty, David, Domke, Grant M. 2024. A national-scale tree volume, biomass, and carbon modeling system for the United States. Gen. Tech. Rep. WO-104. Washington, DC: U.S. Department of Agriculture, Forest Service. 37 p. https://research.fs.usda.gov/treesearch/66998.

Examples

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

# Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


# Most recent estimates of aboveground biomass (excluding foliage)
# for growing-stock trees on timber land
biomass(db = fiaRI_mr,
        landType = 'timber',
        treeType = 'gs')

# Same as above but include foliage
biomass(db = fiaRI_mr,
        landType = 'timber',
        treeType = 'gs',
```

```
        component = c('AG', 'FOLIAGE'))


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

# Belowground (i.e., coarse roots) and stump biomass only
biomass(db = fiaRI_mr,
        component = c('ROOT', 'STUMP'))

# Estimate all biomass components simultaneosly
biomass(db = fiaRI_mr,
        byComponent = 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_ACRE) # Plot of aboveground biomass per acre
```

---

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

Description

Produces estimates of carbon (*metric tonnes*) 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 IPCC forest carbon pools, IPCC forest carbon components, 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, landType = "forest",
       method = "TI", lambda = 0.5, areaDomain = NULL,
       totals = FALSE, variance = FALSE, byPlot = FALSE,
       condList = 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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, belowground live overstory, belowground live understory, standing dead wood, down dead wood, litter, and soil organic).
landType	character ("forest", "timber", 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). When "forest" or "timber", ratios represent average forest carbon density *on forest or timberland*, i.e., non-forested conditions are excluded. When "all", ratios represent average forest carbon density across all land uses, including non-forest.
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 weighting 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.
condList	logical; if TRUE, returns condition-level summaries intended for subsequent use with `customPSE()`.
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 / 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. Importantly, estimates are reported in *metric tonnes* - this is a key distinction relative to other `rFIA` functions, which report estimates in Imperial units. See the following for more info: http://www.epa.gov/climatechange/ghgemissions/usinventoryreport/archive.html

IPCC forest carbon pools are aboveground live biomass, belowground live biomass, dead wood, litter, and soil organic carbon. Aboveground live biomass, belowground live biomass, and dead wood consist of both actual tree measurements and modeled attributes, while litter and soil organic carbon are based completely on modeled attributes. IPCC forest carbon components are defined as follows:

- *AG_OVER_LIVE*: measured attribute. Carbon in the aboveground portion of live trees with at least 1.0 inch dbh/drc. Calculated from the FIADB TREE.CARBON_AG attribute for live trees.
- *AG_UNDER_LIVE*: modeled attribute. Carbon in the aboveground portion of seedlings and woody shrubs. Not a direct sum of P2 or P3 measurements. Calculated from FIADB COND.CARBON_UNDERSTORY attribute.
- *BG_OVER_LIVE*: measured attribute. Carbon in the belowground portion of a tree, including coarse roots with a root diameter of at least 0.1 inch, for live trees with 1.0 inch dbh/drc. Calculated from the FIADB TREE.CARBON_BG attribute for live trees.
- *BG_UNDER_LIVE*: modeled attribute. Carbon in the belowground portion of seedlings and woody shrubs. Not a direct sum of P2 or P3 measurements. Calculated from COND.CARBON_UNDERSTORY_BG attribute.
- *DOWN_DEAD*: modeled attribute. Carbon of woody material greater than 3 inches in diameter on the ground, and stumps and their roots greater than 3 inches in diameter. Not a direct sum of P2 or P3 measurements. Calculated from FIADB COND.CARBON_DOWN_DEAD attribute.
- *STAND_DEAD*: measured attribute. Carbon in the aboveground and belowground portion of standing dead trees with at least 1.0 inch dbh/drc. Belowground measurements include coarse roots with a root diameter of at least 0.1 inch. Calculated from the FIADB TREE.CARBON_AG and TREE.CARBON_BG columns for dead trees. *NOTE*: prior to rFIA v0.1.0, the argument `modelSnag` allowed users to specify whether this was calculated based on modeled attributes or actual tree measurements. This argument has been removed and we no longer provide the option for generating carbon in snags from modeled attributes and instead the estimates come directly from tree measurements in FIADB.
- *LITTER*: modeled attribute. Carbon of organic material on the floor of the forest, including fine woody debris, humus, and fine roots in the organic forest floor layer above mineral soi. This is based on litter carbon observations on FIA plots, but it is not a direct sum of P2 or P3 measurements. Calculated from FIADB COND.CARBON_LITTER attribute.
- *SOIL_ORG*: modeled attribute. Carbon in fine organic material below the soil surface to a depth of 1 meter. Does not include roots. This is based on soil organic carbon observations on FIA plots, but it is not a direct sum of P2 or P3 measurements. Calculated from FIADB COND.CARBON_SOIL_ORG attribute.

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 their 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 `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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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 (metric tonnes/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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

Westfall, James A., Coulston, John W., Gray, Andrew N., Shaw, John D., Radtke, Philip J., Walker, David M., Weiskittel, Aaron R., MacFarlane, David W., Affleck, David L.R., Zhao, Dehai, Temesgen, Hailemariam, Poudel, Krishna P., Frank, Jereme M., Prisley, Stephen P., Wang, Yingfang, Sánchez Meador, Andrew J., Auty, David, Domke, Grant M. 2024. A national-scale tree volume, biomass, and carbon modeling system for the United States. Gen. Tech. Rep. WO-104. Washington, DC: U.S. Department of Agriculture, Forest Service. 37 p. https://research.fs.usda.gov/treesearch/66998.

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://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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

sf data frame 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. NOTE: the countiesRI object was updated in v1.1.0 to be of class sf.

Usage

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

Format

Formal class sf

Examples

```
data(countiesRI)
```

customPSE *Post-stratified estimator for custom variables in FIA Data*

Description

Produces estimates of population totals, ratios, and associated variances for custom variables using FIA's post-stratified inventories. Accepts tree- and condition-level summaries from rFIA estimator functions as input, with potential modifications to associated variables (e.g., custom allometrics can be applied to estimate tree biomass/carbon). See our website for example use cases.

Usage

```
customPSE(db, x, xVars, xGrpBy = NULL, xTransform = NULL, y = NULL,
          yVars = NULL, yGrpBy = NULL, yTransform = NULL, method = "TI",
          lambda = 0.5, totals = TRUE, variance = TRUE)
```

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).
x	data.frame; tree- or condition-list containing numerator variable(s). See details for more info on producing acceptable tree- and condition-lists using rFIA estimator functions.
xVars	name of variable(s) in x to be treated as numerator variables, unquoted. Multiple variables should be combined with c(), e.g., xVars = c(TPA, BAA).
xGrpBy	names of variables in x to be treated as grouping variables for the numerator, unquoted. Multiple variables should be combined with c(), e.g., xVars = c(SPCD, OWNGRPCD).
xTransform	function to be applied to plot-level summaries of numerator variables, e.g., sqrt or function(x){log(x + 0.001)}.
y	data.frame; tree- or condition-list containing denominator variable. See details for more info on producing acceptable tree- and condition-lists using rFIA estimator functions.
yVars	name of variable in y to be treated as denominator variables, unquoted. Maximum one denominator variable allowed at this time.
yGrpBy	names of variables in y to be treated as grouping variables for the numerator, unquoted. Multiple variables should be combined with c(), e.g., xVars = c(SPCD, OWNGRPCD).
yTransform	function to be applied to plot-level summaries of denominator variable, e.g., sqrt or function(x){log(x + 0.001)}.
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.
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.

Details

Workflow and intended use cases

customPSE is intended to be used in combination with standard rFIA estimator functions, like tpa(), area(), and volume(), among others. Standard rFIA estimator functions generate tree- and or condition-lists for standard variables of interest (see treeList and condList arguments in estimator functions). Users may make modifications to these standard variables, for example a variable representing tree crown area may be added to a tree-list produced by tpa() (via some suite of allometrics). Users may then hand their modified tree-list to customPSE to estimate the total and proportion of forested land area in their domain of interest that is covered by tree crowns.

customPSE may be used to estimate population totals for multiple variables simultaneously (total number of trees in a region), and given a denominator variable, the associated population ratios (trees per forested acre, where forested land area is the denominator). Estimation follows the procedures documented in Bechtold and Patterson (2005) and Stanke et al 2020.

Three general forms of ratio estimates may be produced: tree-tree, tree-area, and area-area ratios. For example, if tree height is specified as the numerator variable (adjusted for sampling area by multiplying by TPA), and TPA is specified as the denominator variable, a tree-tree ratio will be produced that represents the height of the average tree within a region of interest. Similarly, if stand age is specified as the numerator (adjusted for sampling area by proportionate area of the forested condition on the plot, i.e., PROP_FOREST), and the proportion area of the plot that is forested is specified as the denominator, an area-area ratio will be produced that represents average stand age within the region of interest. Tree-area ratios are more familiar, such as trees per acre, tree biomass per acre, etc, where a tree variable is specified as the numerator, and proportion of plot area occupied by forestland is the denominator. See our website for detailed examples of each of these ratio estimates.

Input requirements

Estimation of tree variables require the following columns be present in x and/or y: PLT_CN, EVAL_TYP, SUBP, TREE, and TREE_BASIS. Similarly, estimation of area variables require the following columns be present in x and/or y: PLT_CN, EVAL_TYP, CONDID, and AREA_BASIS. Each of these required variables will be returned in tree- and condition-lists generated by standard rFIA estimator functions.

IMPORTANT: Only one of TREE_BASIS or AREA_BASIS may be present x or y, as the presence of these columns are used to determine if variables to be estimated are tree variables or area variables. Some standard rFIA estimator functions will produce tree-lists with both TREE_BASIS and AREA_BASIS listed in output, as the tree-list will contain tree variables (e.g., TPA, BAA) as

well as area variables (e.g., PROP_FOREST, proportion of plot represented by the forested condition where each tree is growing)). To produce a tree-area ratio with such an output, AREA_BASIS must be removed from the data.frame specified in x, and TREE_BASIS must be removed from that specified in y.

Value

- **YEAR**: reporting year associated with estimates
- ***_RATIO**: population ratio estimate, where * will be replaced with the name of each numerator variable
- ***_TOTAL**: population total estimate, where * will be replaced with the name of each numerator/ denominator variable
- ***_RATIO_VAR**: estimated variance of the population ratio
- ***_TOTAL_VAR**: estimated variance of the population total
- **nPlots_x**: number of non-zero plots used to compute numerator estimates
- **nPlots_y**: number of non-zero plots used to compute denominator estimates
- **N**: total number of plots (including zeros) associated with each inventory

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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

Examples

```
# See our website for a more thorough suite of examples
data(fiaRI)

# Get tree-list from tpa
```

```
tree.list <- tpa(fiaRI,
                 treeList = TRUE)

# Estimate trees per acre and basal area per acre
customPSE(db = fiaRI,
          # Numerator variables
          x = dplyr::select(tree.list, -c(AREA_BASIS)),
          xVars = c(TPA, BAA),
          # Denominator variables
          y = dplyr::select(tree.list, -c(TREE_BASIS)),
          yVars = PROP_FOREST)

# Same as above, but rename variables for a clean output
# customPSE(db = fiaRI,
#           x = dplyr::select(tree.list, -c(AREA_BASIS)),
#           # Variables can be renamed using c()
#           xVars = c(NUM = TPA,
#                     BA = BAA),
#           y = dplyr::select(tree.list, -c(TREE_BASIS)),
#           # Variables can be renamed using c()
#           yVars = c(FOREST_AREA = PROP_FOREST))
#
# # Ensure the above matches expected output
# tpa(fiaRI,
#     totals = TRUE,
#     variance = TRUE)
```

`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_UNADJ` (trees per acre) as a state variable and `SPCD` (species) 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 = 0.5,
          stateVar = TPA_UNADJ, grpVar = SPCD, treeDomain = NULL,
          areaDomain = NULL, byPlot = FALSE, condList = 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, PLOTGEOM, 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 methods of stand regeneration, specify c(STDORGCD, OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produced 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 which estimates will be produced for. All includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live (default) 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.
condList	logical; if TRUE, returns condition-level summaries intended for subsequent use with customPSE.
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 ..." error), 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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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, gamma (regional) 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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

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 size class
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 live 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 can be 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 = 0.5, areaDomain = NULL, totals = FALSE,
    variance = FALSE, byPlot = FALSE, condList = FALSE,
    byFuelType = 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, PLOTGEOM, 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 methods of stand regeneration, specify c(STDORGCD, OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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.
condList	logical; if TRUE, returns condition-level summaries intended for subsequent use with customPSE.
byFuelType	logical; if TRUE, returns estimates grouped by fuel type (e.g., 1HR, 10HR, 100HR, 1000HR fuels).
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 by default 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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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
- **FUEL_TYPE**: fuel type associated with each row
- **VOL_ACRE**: estimate of mean volume per acre of dwm (cu.ft/acre)
- **BIO_ACRE**: estimate of mean biomass per acre of dwm (short tons/acre)
- **CARB_ACRE**: estimate of mean carbon mass per acre of dwm (short 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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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://research.fs.usda.gov/treesearch/13615

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)

# 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)

# Estimates of all forestland by county and return
# return spatial object
```

```
dwmSF <- dwm(fiaRI_mr,
             polys = countiesRI,
             returnSpatial = TRUE)
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 of Rhode Island. NOTE: the fiaRI object was updated in v1.1.0 to reflect changes in the FIA Database that took place since creation of the original object.

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: 20.1 Mb Tables: COND_DWM_CALC COND INVASIVE_SUBPLOT_SPP P2VEG_SUBP_STRUCTURE PLOT PLOTGEOM POP_ESTN_UNIT POP_EVAL_GRP POP_EVAL_TYP POP_EVAL POP_PLOT_STRATUM_ASSGN POP_STRATUM SEEDLING SUBP_COND_CHNG_MTRX SUBP_COND SUBPLOT SURVEY TREE_GRM_BEGIN TREE_GRM_COMPONENT TREE_GRM_MIDPT TREE

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	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', 'Wisconsin'))
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 Type Description Table (REF_POP_EVAL_TYP_DESCR) in FIADB P2 User Guide (link in 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://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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 annual change in relative live tree density from the FIADB using the Forest Stability Index (FSI). See Stanke et al. 2021 (doi: 10.1038/s41467-020-20678-z) for a complete description of the the Forest Stability Index.

Usage

```
fsi(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
    bySpecies = FALSE, bySizeClass = FALSE,
    landType = "forest", treeType = "live", method = "TI",
    lambda = 0.5, treeDomain = NULL, areaDomain = NULL,
    totals = TRUE, variance = TRUE, byPlot = FALSE,
    useSeries = FALSE, mostRecent = FALSE, scaleBy = NULL,
    betas = NULL, returnBetas = 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, PLOTGEOM, 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 separate estimates for each ownership group within methods of stand regeneration, specify c(STDORGCD, OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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. Live includes all stems greater than 1 in. DBH which are live (leaning less than 45 degrees). 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 weighting 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.
useSeries	logical; If TRUE, use multiple remeasurements to estimate annual change in relative density on each plot, when available.
mostRecent	logical; If TRUE, only return results for the most recent inventory in each state. Only useful when useSeries=TRUE, as in this case, using clipFIA to select the most recent inventory will drop all but the most recent remeasurement.

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().
betas	data.frame; coefficients of maximum size-density models returned in a previous call to fsi when returnBetas = TRUE. See examples.
returnBetas	logical; If true, returns estimated coefficients of maximum size-density models along with results. These coefficients can then be handed to the beta argument (see above) in subsequent runs. This speeds up processing and ensures the same coefficients are used to model maximum-size density curves between function calls. See Value below for more details.
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.

Please see Stanke et al. 2021 (doi: 10.1038/s41467-020-20678-z) for a complete description of the Forest Stability Index (FSI). In short, the FSI is a direct measure of temporal change in the relative density of live trees, where relative density is defined as the ratio of observed tree density to maximum potential tree density. Maximum potential tree density is modeled as a power function of average tree size - in the current implementation average tree basal area is used. Users may allow both the "slopes" and intercepts of this power function to vary by classified groups, like forest community type using the scaleBy argument. Users may return the estimated parameters of maximum size-density models by specifying returnBetas = TRUE.

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 their 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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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

When `returnBetas = TRUE`, a list will be returned. This list will contain a dataframe named "results", containing estimates of the FSI, and another named "betas", containing estimated parameters of the maximum size-density model. When `returnBetas = FALSE`, a data.frame corresponding with "results" will be returned.

Results 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
- **FSI**: estimate of forest stability index (i.e., annual change in relative live tree density)
- **PERC_FSI**: estimate of % forest stability index (i.e., % annual change in relative live tree density)
- **FSI_STATUS**: indication of the forest stability index (i.e., decline, stable, or expand)
- **FSI_INT**: width of 95% confidence interval of mean FSI

- **PREV_RD**: estimate of relative live tree density at initial measurement of all plots (i.e., observed density / maximum potential density)
- **PREV_RD**: estimate of relative live tree density at final measurement of all plots (i.e., observed density / maximum potential density)
- **TPA_RATE**: standardized estimate of annual change in TPA (proportionate change)
- **BA_RATE**: standardized estimate of annual change in BA (proportionate change)

Betas Within betas, all variable names ending in "upper" or "lower" represent the upper and lower bounds of the 95% Bayesian credible interval of their respective variables. All variable names beginning with "fixed" represent the fixed effects in random slope/intercept models (i.e., the global average).

- **grps**: unique identifier associated with the group (i.e., unique combination of variables listed in `scaleBy`).
- **alpha**: posterior median of scaling factor that describes the maximum tree density at average tree basal area of one sq. ft.
- **rate**: posterior median of negative exponent controlling the decay in maximum tree density with increasing average tree size.
- **n**: number of observations with the group with an approximately normal diameter distribution and no evidence of recent disturbance.

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

Stanke, H., Finley, A. O., Domke, G. M., Weed, A. S., & MacFarlane, D. W. (2021). Over half of western United States' most abundant tree species in decline. Nature Communications, 12(1), 451. doi: 10.1038/s41467-020-20678-z

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

Examples

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

# Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


## Most recent estimates for all live trees in RI
## Allowing maximum size-density relationship to
## vary by forest community type
# fsi(db = fiaRI_mr,
#     scaleBy = FORTYPCD)
#
# ## Same as above at the plot-level
# fsi(db = fiaRI_mr,
#     scaleBy = FORTYPCD,
#     byPlot = TRUE)
#
#
# ## Same as above, but return the estimated coefficients of the
# ## maximum size-density model
# results <- fsi(db = fiaRI_mr,
#                scaleBy = FORTYPCD,
#                returnBetas = TRUE)
# ## Our results are stored in a list, where "results" gives us the
# ## estimates of the FSI, and "betas" gives us the estimated
# ## model coefficients
# results$results # FSI estimates
# results$betas # model coefficients
#
#
# ## Estimates for live white pine ( > 12" DBH) on
# ## forested mesic sites (all available inventories)
# ## Here we instead allow maximum size-density relationships
# ## to vary by site productivity class
# fsi(fiaRI_mr,
#     scaleBy = SITECLCD,
#     treeType = 'live',
#     treeDomain = SPCD == 129 & DIA > 12, # Species code for white pine
#     areaDomain = PHYSCLCD %in% 21:29) # Mesic Physiographic classes
```

`getDesignInfo` *Extract design features for FIA inventories*

Description

Extracts design information for post-stratified FIA inventories, intended to aid the development of alternative model-based estimators of forest variables. Design information is currently limited to estimation unit land area (AREA_USED) and strata weights (proportion of estimation unit represented by each stratum). This is sufficient to acknowledge design features in an aspatial model, however, inclusion probabilities and strata boundaries will be necessary to incorporate spatial predictors.

Usage

```
getDesignInfo(db,
              type = c("ALL", "CURR", "VOL", "GROW", "MORT",
                       "REMV", "CHNG", "DWM", "REGEN"),
              mostRecent = TRUE,
              evalid = NULL)
```

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.
type	character ('ALL', 'CURR', 'VOL', 'GROW', 'MORT', 'REMV', 'CHNG', 'DWM', 'REGEN'). See Reference Population Evaluation Type Description Table (REF_POP_EVAL_TYP_DESC in FIADB P2 User Guide (link in references) for descriptions of evaluation types.
mostRecent	logical; if TRUE, returns EVALIDs associated with most recent inventory.
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 locate this value.

Details

The FIA database is not limited for use with standard design-based estimators. A plethora of recent work has highlighted the improvements that model-assisted and model-based estimators can provide relative to the standard post-stratified estimators implemented in rFIA. However, when implementing alternative model-assisted or model-based estimators it is important to accommodate the design information for post-stratified FIA inventories. getDesignInfo() allows you to quickly extract design-based information for use when implementing alternative estimators with FIA data. See the Stanke et al. 2021 reference below for a paper describing use of rFIA for implementing model-based estimators to improve small area estimation of forest parameters.

Value

Generates a data frame with the columns described below, containing the design information associated with one or multiple FIA inventories.

For reference, there are often multiple non-overlapping estimation units within a state, and multiple, non-overlapping, and exhaustive strata within each estimation unit. Estimation unit and strata

boundaries vary with inventories (i.e., by reporting year and by inventory type), though multiple inventories may draw from data collected at a single plot visit. Hence, ESTN_UNIT_CN and STRATUM_CN are specific to EVALIDs (or alternatively, the combination of STATE, YEAR, and EVAL_TYP, for all states except Texas). However, a single PLT_CN may be associated with multiple ESTN_UNIT_CNs and STRATUM_CNs.

- **STATECD**: unique identifier for states.
- **YEAR**: reporting year associated with estimates (END_INVYR from POP_EVAL).
- **EVAL_TYP**: an identifier describing the type of evaluation. For example, "EXPDWM" represents sampled plots used for down woody material estimates.
- **EVALID**: unique identifier that represents the population used to produce a type of estimate.
- **ESTN_UNIT_CN**: unique identifier for estimation unit.
- **AREA_USED**: area of estimation unit used for population estimation.
- **STRATUM_CN**: unique identifier for strata.
- **STRATUM_WGT**: proportion of estimation unit area (AREA_USED) that is occupied by a particular stratum. Defined on the range (0,1].
- **pltID**: unique identifier for plot *location*.
- **PLT_CN**: unique identifier for plot *visit*.

Author(s)

Hunter Stanke and Andrew Finley

References

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

See Also

`findEVALID`

Examples

```
# Load the Rhode Island subset included w/ rFIA
data(fiaRI)

# Extract the design information associated with the most recent current
# volume inventory in the state (2018 for this subset)
wgts <- getDesignInfo(db = fiaRI, mostRecent = TRUE, type = 'VOL')
```

getFIA *Download FIA Data and Load into R*

Description

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. Requires an internet connection to access and download tables from the FIA Datamart. **R stops downloads after 60 seconds by default. To prevent this (allow download for 1 hour), run** `options(timeout=3600)`**.**

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, TREE_GRM_MIDPT, TREE_GRM_BEGIN, SUBP_COND_CHNG_MTRX, SEEDLING, SURVEY, SUBP_COND, P2VEG_SUBP_STRUCTURE. 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 Website Questions: https://research.fs.usda.gov/programs/fia#data-and-tools Direct email: SM.FS.fia@usda.gov E.g., Problems accessing the state CSV files.

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

See Also

`readFIA`

Examples

```
# Allow downloads that take up to 1 hour (defaults to 60 seconds)
options(timeout=3600)

# 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 growth, recruitment, mortality, and harvest rates from FIADB*

Description

Produces estimates of annual growth, 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`).

Usage

```
growMort(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE, bySpecies = FALSE,
         bySizeClass = FALSE, landType = 'forest', treeType = 'all',
         method = 'TI', lambda = 0.5, stateVar = 'TPA', treeDomain = NULL,
         areaDomain = NULL, totals = FALSE, variance = FALSE,
         byPlot = FALSE, treeList = 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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" or "gs"); Type of tree that estimates will be produced for. All (default) includes all stems, live and dead, greater than 5 in. DBH (or those that grow to 5in by the end of time period 2). 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, SNDVOL, SAWVOL, SAWVOL_BF (board feet).

`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.

`treeList` logical; if TRUE, returns tree-level summaries intended for subsequent use with `customPSE()`.

`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 1. 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 (both natural and harvest) 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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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_***: estimate of mean annual recruitment
- **MORT_***: estimate of mean annual mortality
- **REMV_***: estimate of mean annual removals (harvest)
- **GROW_***: estimate of mean annual growth on survivors
- **CHNG_***: estimate of mean annual net change (i.e., growth + recruitment - mortality - removals)
- **RECR_PERC**: estimate of mean percent of individuals subject to recruitment annually (recruitment / previous total)
- **MORT_PERC**: estimate of mean percent of individuals subject to mortality annually (mortality / previous total)
- **REMV_PERC**: estimate of mean percent of individuals subject to removal (harvest) annually (removals / previous total)
- **GROW_PERC**: estimate of mean annual growth on survivors (%) (growth / previous total)
- **CHNG_PERC**: estimate of mean annual net change (%) (net change / previous total)
- **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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

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
```

`intersectFIA` *Intersect FIA data with spatial polygons*

Description

Performs spatial intersection between FIA data and user-supplied spatial polygons (sp or sf). Polygon attributes appended to PLOT table, and hence can be used as grouping variables in subsequent calls to rFIA estimator functions. Alternative to the `polys` argument in rFIA estimator functions.

Usage

```
intersectFIA(db, polys, 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).
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced for region encompassed by each areal unit. FIA plot locations will be reprojected to match projection of `polys` object.
`nCores`	numeric; number of cores to use for parallel implementation. Check available cores using `detectCores`. Default = 1, serial processing.

Details

All polygon attributes will be joined onto the PLOT table.

Primarily useful if you intend to make multiple calls to rFIA estimator functions, e.g., you need to call both `tpa` and `biomass` and group by spatial polygons in both cases. If using the `polys` argument in each function call, spatial intersection will occur mulitple times, and hence be slower than performing the intersection a single time upfront.

Value

FIA.Database or Remote.FIA.Database, depending on specification of db.

Author(s)

Hunter Stanke and Andrew Finley

References

rFIA website: https://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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

clipFIA

Examples

```
data(fiaRI)
data(countiesRI)

# Perform spatial intersection
db <- intersectFIA(fiaRI,
                   countiesRI)

# Group estimates by variable defined
# in `countiesRI`
tpa(db,
    grpBy = COUNTY)
```

`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 = 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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 may 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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

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
```

`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 95% 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).

Value

A `ggplot` object containing the resulting plot.

Author(s)

Hunter Stanke and Andrew Finley

Examples

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



################### 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)

## Produce animated plot
if(interactive()) {
  plotFIA(tpaRI, y = TPA, animate = TRUE, legend.title = 'Abundance (TPA)')
}
## With a square root transform
if(interactive()) {
 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
```

---

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

Description

Loads FIA Datatables into R from .csv files stored in a local directory (common, easy), or from a database connection (uncommon, difficult). 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.

Usage

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

Arguments

dir	directory where .csv files of FIA tables are stored.
con	database connection (e.g., produced with DBI::dbConnect(). Only required when reading tables from a database connection. Most users will want to specify dir instead, reading FIA tables from .csv files.)
schema	SQL Schema that contains FIA tables. Only required when reading tables from a database connection, which most users should avoid.
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 data.table::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, TREE_GRM_MIDPT, TREE_GRM_BEGIN, SUBP_COND_CHNG_MTRX, SEEDLING, SURVEY, SUBP_COND, P2VEG_SUBP_STRUCTURE. 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. North Carolina and Virginia 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. NC_PLOT and VA_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 tables. List elements represent individual FIA tables 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. North Carolina and Virginia state downloads), corresponding tables will be merged (e.g. PLOT table returned contains plots in both North Carolina and Virginia).

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 downloading 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://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

See Also

`clipFIA`, `getFIA`

Examples

```
# The following examples shows how you
# can take an existing in-memory FIA.Database,
# 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())
```

`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 = 0.5, treeDomain = NULL, areaDomain = NULL,
         totals = FALSE, variance = FALSE, byPlot = FALSE,
         treeList = 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
`polys`	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produced 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 weighting 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.
treeList	logical; if TRUE, returns tree-level summaries intended for subsequent use with customPSE.
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 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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

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 = 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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, 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 as dominant, subdominant, or intermdediate crown-classes are used to classify stand structural stage.

Diameter Classes

- *Pole*: 12.7 - 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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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

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,
            byPlot = TRUE)

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

## 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)
```

---

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, treeList = 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produced 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 includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live (default) 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.

treeList — logical; if TRUE, returns tree-level summaries intended for subsequent use with customPSE.

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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

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 from 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. 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.

polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced for regions 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 weighting 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. Percent coverage estimates are returned separately by canopy layer and growth habit (the general appearance of the plant, including size, shape, growth form, and orientation). Canopy layers

Canopy layers

- 0 - 2.0 feet

- 2.1 - 6.0 feet
- 6.1 - 16.0 feet
- Greater than 16 feet
- Aerial: Canopy cover for all layers

Growth habit

- *Forbs*: herbaceous, broad-leaved plants; includes non-woody-vines and ferns (does not include mosses and cryptobiotic crusts).
- *Graminoids*: grasses and grass-like plants (includes rushes and sedges).
- *Non-tally tree*: tree species not on a particular FIA work unit's tree tally list that are woody plants with a single well-defined, dominant main stem, not supported by other vegetation or structures (not vines), and which are, or expected to become, greater than 13 feet in height. Seedlings, saplings, and mature plants are included.
- *Shrubs/vines*: woody, multiple-stemmed plants of any size, subshrubs (low-growing shrubs under 1.5 feet tall at maturity), and woody vines. Most cacti are included in this category.
- *Tally tree*: all core tree species and any core optional tree species selected by a particular FIA work unit. Only tree species on the FIA Master Tree Species List (or those listed as hybrid, variety, or subspecies) are included. Any plant of that species is included, regardless of its shape and regardless of whether it was tallied on the subplot or microplot during the tree tally. Seedlings, saplings, and mature plants are included.

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).

Forest land must have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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_TOTAL**: estimate of areal coverage of the growth habit within the canopy layer (acres)
- **AREA_TOTAL**: 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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

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. Only stems 5 inches DBH or greater are included in estimates. 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 = 'all',
           method = 'TI', lambda = .5, treeDomain = NULL,
           areaDomain = NULL, totals = FALSE, variance = FALSE,
           byPlot = FALSE, treeList = 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, PLOTGEOM, 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 methods of stand regeneration, specify c(STDORGCD, OWNGRPCD).
polys	sp or sf Polygon/MultiPolgyon object; Areal units to bin data for estimation. Separate estimates will be produced 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", or "gs"); Type of tree which estimates will be produced for. See details for more info.
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 weighting 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.

treeList — logical; if TRUE, returns tree-level summaries intended for subsequent use with `customPSE()`

.

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.

When `treeType = 'all'` (default), estimates are of *net* growth rates (including recruitment and mortality), and hence 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. Alternatively, when `treeType = 'live'`, growth rates are calculated using only trees that were alive at both plot visits and give a more realistic representation of individual tree growth.

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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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 net volume growth of a stem (cu. ft./ yr)
- **NETVOL_GROW_AC**: estimate of mean annual net volume growth per acre (cu. ft./ acre/ yr)
- **SAWVOL_GROW**: estimate of mean annual net sawlog volume growth of a stem (MBF / yr)
- **SAWVOL_GROW_AC**: estimate of mean annual net sawlog volume growth per acre (MBF/ 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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-database

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.

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
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
```

volume *Estimate merchantable tree volume from the FIADB*

Description

Produces estimates of merchantable tree volume (i.e., merchantable bole volume and sawlog volume) 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

```
volume(db, grpBy = NULL, polys = NULL, returnSpatial = FALSE,
       bySpecies = FALSE, bySizeClass = FALSE, landType = "forest",
       treeType = "live", volType = "NET", method = "TI",
       lambda = 0.5, treeDomain = NULL, areaDomain = NULL,
       totals = FALSE, variance = FALSE, byPlot = FALSE,
       treeList = 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, PLOTGEOM, 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 methods of stand regeneration, specify `c(STDORGCD, OWNGRPCD)`.
polys	`sp` or `sf` Polygon/MultiPolgyon object; Areal units to bin data for estimation. Seperate estimates will be produced 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 which estimates will be produced for. All includes all stems, live and dead, greater than 1 in. DBH. Live/Dead includes all stems greater than 1 in. DBH which are live (default) 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.

volType — character, one of: "NET", "SOUND", or "GROSS"; merchantable volume definition to use in estimation. See details for more info.

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.

treeList — logical; if TRUE, returns tree-level summaries intended for subsequent use with `customPSE`

.

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, tree volume per acre is computed using a sample-based ratio-of-means estimator of total volume / total land area within the domain of interest.

Estimates of total merchantable volume are in units of cubic feet (CF), and estimates of sawlog volume in terms of cubic feet and thousand board feet (MBF; International 1/4 inch rule). FIA's net volume definition is used by default (`volType = "NET"`): "net volume of wood in the central stem of a sample tree 5.0 inches d.b.h., 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 <4.0 inches in diameter... Does not include rotten, missing, and form cull (volume loss due to rotten, missing, and form cull defect has been deducted)". Users opt to use two alternative definitions: sound volume (`volType = "SOUND"`) or gross volume (`volType = "GROSS"`). Sound volume is identical to net volume except that sound includes volume from portions of the stem that can be considered "form cull" under the net volume definition (e.g., sweep). In contrast, gross volume is identical to the net volume definition except that gross includes volume from portions of the stem that are rotten, missing, and considered form cull.

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 have at least 10-percent canopy cover by live tally 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 forest and non-forested lands that meet the mimium tree canopy cover and forest areas adjacent to urban and built-up lands. The minimum area for classification of forest land is 1 acre in size and 120 feet wide measured stem-to-stem from the outer-most edge. Roadside, streamside, and shelterbelt strips of trees must have a width of at least 120 feet and continuous length of at least 363 feet to qualify as forest land. Tree-covered areas in agricultural production settings, such as fruit orchards, or tree-covered areas in urban settings, such as city parks, are not considered forest land.

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
- **BOLE_CF_ACRE**: estimate of mean merchantable bole volume per acre (cu.ft./acre)
- **SAW_CF_ACRE**: estimate of mean merchantable sawtimber volume per acre (cu.ft./acre)
- **SAW_MBF_ACRE**: estimate of mean merchantable sawtimber volume per acre (thousand board feet/acre; International 1/4 inch rule)
- **nPlots_TREE**: number of non-zero plots used to compute volume 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://doserlab.com/files/rfia/

FIA Database User Guide: https://research.fs.usda.gov/understory/forest-inventory-and-analysis-databas

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.

Examples

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

# Most recents subset
fiaRI_mr <- clipFIA(fiaRI)


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


# Same as above, but using the gross volume definition
volume(db = fiaRI_mr,
       landType = 'timber',
       treeType = 'gs',
       volType = 'gross')

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

# Estimates for live white pine ( > 12" DBH) on forested mesic sites (all available inventories)
volume(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)
volume(db = fiaRI_mr,
```

```
       grpBy = STAND_AGE)

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

# Most recent estimates for live stems on forest land by species
volume(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
# volume(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 <- volume(fiaRI_mr,
               polys = countiesRI,
               returnSpatial = TRUE)
plot(ctSF) # Plot multiple variables simultaneously
plotFIA(ctSF, SAW_MBF_ACRE) # Plot of saw volume, in board feet
```

`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 data.table::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).

Value

No return value, called to export FIA.Database object to a local directory.

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

∗ **datasets**
countiesRI, 26
fiaRI, 40

area, 3
areaChange, 7

biomass, 12

carbon, 19
clipFIA, 24, *41, 42*, *58*, *64*, *69*
countiesRI, 26
customPSE, 27
cut, *63*

detectCores, *4*, *25*, *32*, *37*, *44*, *50*, *57*, *60*, *71*, *76*, *80*, *85*, *90*, *95*, *100*
diversity, 30
dwm, 35

fiaRI, 40
findEVALID, *25, 26*, 41, *48*
fsi, 42

getDesignInfo, 47
getFIA, *25*, *31*, *36*, 50, *59*, *69, 70*, *75*, *79*, *84*, *94*, *100*
growMort, 52

intersectFIA, 57
invasive, 59

makeClasses, *31*, *43*, 63, *80*, *94*

parallel, *5*, *10*, *16*, *22*, *33*, *38*, *45*, *51*, *54*, *61*, *68*, *72*, *77*, *81*, *87*, *91*, *96*, *100*
plotFIA, 64

readFIA, *25*, *31*, *36*, *40*, *51*, *59*, 67, *70*, *75*, *79*, *84*, *94*, *100*

seedling, 70
standStruct, 74

tpa, *63*, *66*, 79

vegStruct, 84
vitalRates, 88
volume, 94

writeFIA, 99