Package ‘FIESTAutils’

March 25, 2025

Type Package

Title Utility Functions for Forest Inventory Estimation and Analysis

Version 1.3.1

Date 2025-03-25

Description A set of tools for data wrangling, spatial data analysis, statistical modeling (including direct, model-assisted, photo-based, and small area tools), and USDA Forest Service data base tools. These tools are aimed to help Foresters, Analysts, and Scientists extract and perform analyses on USDA Forest Service data.

Depends R (>= 4.2.0)

Imports data.table, DBI, gdalraster, graphics, hbsae, JoSAE, mase, methods, nlme, Rcpp, RColorBrewer, RPostgres, RSQLite, sae, sf, sqldf, stats, terra, units, utils

Suggests knitr

License GPL-3

Copyright See file COPYRIGHTS for details.

URL https://github.com/USDAForestService/FIESTAutils

BugReports https://github.com/USDAForestService/FIESTAutils/issues

Encoding UTF-8

LazyData true

LinkingTo Rcpp

RoxygenNote 7.3.2

NeedsCompilation yes

Author Tracey Frescino [aut],
Chris Toney [aut],
Grayson White [aut, cre],
Joshua Yamamoto [aut]

Maintainer Grayson White <graysonwhite13@gmail.com>

Repository CRAN

Date/Publication 2025-03-25 20:40:06 UTC

Contents

`.update_refs` *Updates reference tables*

Description

Updates reference tables

Usage

```
.update_refs(write = FALSE)
```

Arguments

`write`	Logical. Should the internal reference tables be overwritten?

Value

No return value. Called for side effects.

Author(s)

Josh Yamamoto

`database_options` *Database options.*

Description

Returns a list of user-supplied parameters and parameter values for database access.

Usage

```
database_options(
  dbname = NULL,
  host = NULL,
  port = NULL,
  user = NULL,
  password = NULL,
  schema = NULL,
  dbconnopen = TRUE,
  ...
)
```

Arguments

`dbname`	String. Name of database.
`host`	String. Name of database host.
`port`	String. Database port.
`user`	String. User name for database access.
`password`	String. Password for database access.
`schema`	String. Name of schema in database.
`dbconnopen`	Logical. If TRUE, keep database connection open.
`...`	For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for saving data.

Author(s)

Tracey S. Frescino

Examples

```
savedata_options(outfolder = "path", overwrite_dsn = FALSE)
```

datExportData *Spatial - Exports a data frame object.*

Description

Exports a data frame object to a specified output.

Usage

```
datExportData(
  dfobj,
  create_dsn = FALSE,
  index.unique = NULL,
  index = NULL,
  lowernames = FALSE,
  savedata_opts = savedata_options(),
  dbconn = NULL,
  dbconnopen = TRUE
)
```

Arguments

`dfobj` Data.frame class R object. Data frame object to export.

`create_dsn` Boolean.

`index.unique` String. Name of variable(s) in dfobj to make unique index.

`index` String. Name of variable(s) in dfobj to make (non-unique) index.

`lowernames` Logical. If TRUE, convert column names to lowercase before writing to output. dbconnopen Logical. If TRUE, keep database connection open.

`savedata_opts` List. See help(savedata_options()) for a list of options.

`dbconn` Open database connection.

`dbconnopen` Logical. If TRUE, keep database connection open.

Details

Wrapper for sf::st_write function.

Value

An sf spatial object is written to the out_dsn.

Note

If out_fmt='shp':
The ESRI shapefile driver truncates variable names to 10 characters or less. Variable names are changed before export using an internal function (trunc10shp). Name changes are output to the outfolder, 'outshpnm'_newnames.csv.

If sf object has more than 1 record, it cannot be exported to a shapefile.

Author(s)

Tracey S. Frescino

`datSum_options` *Options for summarizing tree data.*

Description

Returns a list of user-supplied parameters and parameter values for summarizing tree data.

Usage

```
datSum_options(
  lbs2tons = TRUE,
  metric = FALSE,
  tround = 5,
  TPA = TRUE,
  adjTPA = 1,
  ACI = FALSE,
  adjtree = FALSE,
  adjvar = "tadjfac",
  keepall = FALSE,
  NAto0 = TRUE,
  ...
)
```

Arguments

`lbs2tons` Logical. If TRUE, converts biomass or carbon variables from pounds to tons (1 pound = 0.0005 short tons). If metric=TRUE, converts to metric tons, else short tons.

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

`tround` Number. The number of digits to round to. If NULL, default=5.

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

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

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

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

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

`keepall` Logical. If TRUE, keeps all plots in dataset with NA values. If FALSE, keeps only summed data when not NA.

`NAto0` Logical. If TRUE, change NA values to 0

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for summarizing tree data.

Author(s)

Tracey S. Frescino

Examples

```
datSum_options(lbs2tons = TRUE, metric = TRUE)
```

`DBtestPostgreSQL` *Database - Test a PostgreSQL database.*

Description

Checks a PostgreSQL database.

Usage

```
DBtestPostgreSQL(
  dbname = NULL,
  host = NULL,
  port = NULL,
  user = NULL,
  password = NULL,
  dbconnopen = FALSE,
  showlist = TRUE,
  ...
)
```

Arguments

`dbname`	String. Name of the database on the host.
`host`	String. Host name.
`port`	String. Port number.
`user`	String. User name.
`password`	String. Password.
`dbconnopen`	Logical. If TRUE, the database connection is returned and not closed.
`showlist`	Logical. If TRUE, prints list of tables in database.
`...`	Additional authentication arguments passed to DBI::dbConnect

Value

An S4 object that inherits from DBIConnection via the DBI package if dbconnopen = TRUE, or NULL otherwise. For more information, see 'help(DBI::dbConnect)'.

Author(s)

Tracey S. Frescino

DBtestSQLite *Database - Checks access to a SQLite database.*

Description

Checks a SQLite database.

Usage

```
DBtestSQLite(
  SQLitefn = NULL,
  gpkg = FALSE,
  dbconnopen = FALSE,
  outfolder = NULL,
  showlist = TRUE,
  returnpath = TRUE,
  createnew = TRUE,
  stopifnull = FALSE,
  overwrite = TRUE
)
```

Arguments

`SQLitefn`	String. Name of SQLite database (*.sqlite).
`gpkg`	Logical. If TRUE, Sqlite geopackage database.
`dbconnopen`	Logical. If TRUE, the dbconn connection is not closed.
`outfolder`	String. Optional. Name of output folder. If NULL, export to working directory.
`showlist`	Logical. If TRUE, shows list of tables in database.
`returnpath`	Logical. If TRUE, returns full path to SQLite file name. If FALSE, returns SQLitefn.
`createnew`	If TRUE, creates new SQLite database.
`stopifnull`	Logical. If TRUE, stops if SQLite database doesn't exist.
`overwrite`	Logical. If TRUE, overwrites data.

Value

Character string containing the path to the SQLite database of interest.

Author(s)

Tracey S. Frescino

eval_options *List of population tables.*

Description

Returns a list of user-supplied parameters and parameter values for data evaluation (FIA or custom) extraction to be supplied to *DB functions.

Usage

```
eval_options(
  Cur = FALSE,
  Endyr = NULL,
  Endyr.filter = NULL,
  All = FALSE,
  Type = "VOL",
  evalid = NULL,
  invyrs = NULL,
  measyrs = NULL,
  varCur = "INVYR",
  evalType = NULL,
  ...
)
```

Arguments

`Cur` Logical. If eval=’FIA’: extract plots with most current evaluation. If eval=’custom’: extract the most current sampled plots in the database.

`Endyr` Integer (YYYY). If eval=’FIA’, defines end year for extracting one or more FIA evaluation. If eval=’custom’, defines end year for extracting the most current sampled plots until.

`Endyr.filter` Filter. If endyr != NULL, a filter to identify when to use measEndyr, such as areas or plots identified as being disturbed in a particular year. In this example, plots sampled after the disturbance will be excluded.

`All` Logical. If eval=’FIA’: includes all evaluations in database (annual inventory only). If eval=’custom’: includes all years in database (annual inventory only).

`Type` String vector. Evaluation types (’ALL’,’CURR’,’VOL’,’P2VEG’, DWM’,’INV’,’CHNG’,’GRM’,’REGEN If eval=’FIA’, Type is equivalent to plots for FIA Evaluations where ’ALL’ includes nonsampled plots; ’CURR’ and ’VOL’ include plots used for area or tree estimates, respectively; Type = ’GRM’ includes plots used for growth, removals, mortality; and Type = ’CHNG’ includes plots used for change estimates (See FIA database manual for regioin availability and/or differences (https://www.fia.fs.usda.gov/library/database-documentation/index.php) If eval=’custom’, the associated tables are extracted for each Type. Multiple Types are accepted.

`evalid` Integer. Only eval='FIA': extract data for a specific evaluation period. See notes for more information about FIA Evaluations.

`invyrs` Integer vector. eval='custom': defines specific inventory years of data (e.g., 2010:2015). See FIA manual for definition of INVYR.

`measyrs` Integer vector. eval='custom': defines specific measurement years of data (e.g., 2010:2015).

`varCur` String. Name of variable to use for most current plot ('MEASYEAR', 'INVYR').

`evalType` Deprecated. Use Type instead.

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Tracey S. Frescino

Examples

```
eval_options(invyrs = 2015:2018)
```

`GDT_NAMES` *Reference tables - gdal data types.*

Description

Table with gdal data type names.

Format

A vector of 12 data type values.

Source

gdal values.

`kindcd3old` *Reference table - List of RMRS plots that have fallen out of inventory because they were not found or they were in the wrong place.*

Description

Table with variable codes (VALUE) and descriptions (MEANING).

Format

A dataframe

Source

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

`multest_options` *Multest output options.*

Description

Returns a list of user-supplied parameters and parameter values for outputting multest with custom aesthetics.

Usage

```
multest_options(
  multest_estimators = "all",
  multest_fmt = "csv",
  multest_outfolder = NULL,
  multest_dsn = NULL,
  multest_layer = NULL,
  multest.append = FALSE,
  multest.AOIonly = FALSE,
  ...
)
```

Arguments

`multest_estimators`
: String vector. If multest = TRUE, vector of estimators to include in multest output ('JU.GREG','JU.EBLUP','JFH','hbsaeU','hbsaeA'). See ref_estimators for descriptions. Use multest_estimators = 'all' to output all estimators.

`multest_fmt`
: String. Format for multest output tables ('csv', 'sqlite', 'gpkg').

`multest_outfolder`
: String. Outfolder for multest. If NULL, same as outfolder.

`multest_dsn`
: String. Name of database if multest_fmt = c('sqlite', 'gpkg').

`multest_layer`
: String. Name of database layer if multest_fmt = c('sqlite', 'gpkg').

`multest.append`
: Logical. If TRUE, appends multest dataframe to output.

`multest.AOIonly`
: Logical. If TRUE, appends multest dataframe (AOI=1) to output.

`...`
: For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for outputting multest.

Author(s)

Grayson W. White

Examples

```
multest_options(multest.append = TRUE)
```

popFilters *Population data filters.*

Description

Returns a list of user-supplied parameters and parameter values for population data filters.

Usage

```
popFilters(
  evalid = NULL,
  states = NULL,
  evalCur = FALSE,
  evalEndyr = NULL,
  measCur = FALSE,
  measEndyr = NULL,
  invyrs = NULL,
  measyrs = NULL,
  intensity = NULL,
  ACI = FALSE,
  AOIonly = FALSE,
  pfilter = NULL,
  ...
)
```

Arguments

`evalid` Numeric. FIA Evaluation identifier for subsetting plots for population.

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

`evalCur` Logical. If TRUE, the most current FIA Evaluation is extracted for state(s).

`evalEndyr` Number. The end year of the FIA Evaluation of interest. Selects only sampled plots and conditions for the evaluation period. If more than one state, create a named list object with evalEndyr labeled for each state (e.g., list(Utah=2014, Colorado=2013).

`measCur` Logical. If TRUE, the most current sampled plots available for state(s).

`measEndyr` Number. The most current sampled plots measured before or during end given..

`invyrs` Integer vector. Inventory year(s) (e.g., c(2000, 2001, 2002)).

`measyrs` Integer vector. Measurement year(s) (e.g., c(2000, 2001, 2002)).

`intensity` Integer code. Code(s) indicating intensity to use for population.

`ACI` Logical. If TRUE, including All Condition Inventory (ACI) plots.

`AOIonly` Logical. If TRUE, and there is an AOI (1/0) attribute in the population data, only AOI=1 are used for estimation.

`pfilter` String. Logical filter for plot or pltassgn.

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for population data filters.

Author(s)

Grayson W. White

---

popTableIDs *List of population table unique IDs.*

Description

Returns a list of user-supplied parameters and parameter values for data table unique IDs to be supplied to *pop functions.

Usage

```
popTableIDs(
  cond = "PLT_CN",
  plt = "CN",
  tree = "PLT_CN",
  seed = "PLT_CN",
  seedling = "PLT_CN",
  vsubpspp = "PLT_CN",
  p2veg_subplot_spp = "PLT_CN",
  vsubpstr = "PLT_CN",
  p2veg_subp_structure = "PLT_CN",
  invsubp = "PLT_CN",
  invasive_subplot_spp = "PLT_CN",
  subplot = "PLT_CN",
  subp_cond = "PLT_CN",
  dwm = "PLT_CN",
  cond_dwm_calc = "PLT_CN",
  sccm = "PLT_CN",
  subp_cond_chng_mtrx = "PLT_CN",
  grm = "PLT_CN",
  tree_grm_component = "PLT_CN",
  begin = "PLT_CN",
  tree_grm_begin = "PLT_CN",
  midpt = "PLT_CN",
  tree_grm_midpt = "PLT_CN",
  plot = "CN",
  pltu = "CN",
  plotu = "CN",
  condu = "PLT_CN",
  ...
)
```

Arguments

`cond`	String. Unique identifier of plot in cond.
`plt`	String. Unique identifier of plot in plt.
`tree`	String. Unique identifier of plot in tree and seed.
`seed`	String.
`seedling`	String.
`vsubpspp`	String.
`p2veg_subplot_spp`	String.
`vsubpstr`	String.
`p2veg_subp_structure`	String.
`invsubp`	String.
`invasive_subplot_spp`	String.
`subplot`	String.
`subp_cond`	String.
`dwm`	String.
`cond_dwm_calc`	String.
`sccm`	String.
`subp_cond_chng_mtrx`	String.
`grm`	String.
`tree_grm_component`	String.
`begin`	String.
`tree_grm_begin`	String.
`midpt`	String.
`tree_grm_midpt`	String.
`plot`	String.
`pltu`	String.
`plotu`	String.
`condu`	String.
`...`	For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied unique identifier of a plot in population tables.

Author(s)

Grayson W. White

---

popTables *List of population tables.*

Description

Returns a list of user-supplied parameters and parameter values for data tables to be supplied to *pop functions.

Usage

```
popTables(
  cond = "COND",
  plt = "PLOT",
  tree = "TREE",
  seed = "SEEDLING",
  seedling = "SEEDLING",
  vsubpspp = "P2VEG_SUBPLOT_SPP",
  p2veg_subplot_spp = "P2VEG_SUBPLOT_SPP",
  vsubpstr = "P2VEG_SUBP_STRUCTURE",
  p2veg_subp_structure = "P2VEG_SUBP_STRUCTURE",
  invsubp = "INVASIVE_SUBPLOT_SPP",
  invasive_subplot_spp = "INVASIVE_SUBPLOT_SPP",
  subplot = "SUBPLOT",
  subp_cond = "SUBP_COND",
  dwm = "COND_DWM_CALC",
  cond_dwm_calc = "COND_DWM_CALC",
  sccm = "SUBP_COND_CHNG_MTRX",
  subp_cond_chng_mtrx = "SUBP_COND_CHNG_MTRX",
  grm = "TREE_GRM_COMPONENT",
  tree_grm_component = "TREE_GRM_COMPONENT",
  begin = "TREE_GRM_BEGIN",
  tree_grm_begin = "TREE_GRM_BEGIN",
  midpt = "TREE_GRM_MIDPT",
  tree_grm_midpt = "TREE_GRM_MIDPT",
  plot = "plot",
  pltu = "pltu",
  plotu = "plotu",
  condu = "condu",
  ...
)
```

Arguments

`cond` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Condition-level data with one record for each condition, including or excluding nonsampled conditions. Plot variables and strata/estimation unit variable(s) may be included if plt and pltassgn=NULL. See details for necessary variables to include.

`plt` DF/DT, Optional. R object, sf R object, comma-delimited file(*.csv), layer or spatial layer in dsn, or shapefile(*.shp). Plot-level data with one record for each plot, including or excluding nonsampled conditions. If nonsampled plots are included, PLOT_STATUS_CD variable must be in table or a filter defined in plt.nonsamp.filter.

`tree` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level data with one record for each tree. Tree data are aggregated to condition-level. See details for necessary variables to include.

`seed` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Seedling data with one record for each seedling count.

`seedling` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Seedling data with one record for each seedling count.

`vsubpspp` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Vegetation species-level data with one record for each species (P2VEG_SUBPLOT_SPP).

`p2veg_subplot_spp` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Vegetation species-level data with one record for each species (P2VEG_SUBPLOT_SPP).

`vsubpstr` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Vegetation species-structure data with one record for each species (P2VEG_SUBP_STRUCTURE).

`p2veg_subp_structure` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Vegetation species-structure data with one record for each species (P2VEG_SUBP_STRUCTURE).

`invsubp` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Invasive species data with one record for each species (INVASIVE_SUBPLOT_SPP).

`invasive_subplot_spp` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Invasive species data with one record for each species (INVASIVE_SUBPLOT_SPP).

`subplot` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Subplot-level data with one record for each species (SUBPLOT).

`subp_cond` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Subplot condition-level data with one record for each species (SUBP_COND).

`dwm` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Calculated down woody material (COND_DWM_CALC).

`cond_dwm_calc` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Calculated down woody material (COND_DWM_CALC).

`sccm` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Subplot-level data (SUBP_COND_CHNG_MTRX).

`subp_cond_chng_mtrx` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Subplot-level data (SUBP_COND_CHNG_MTRX).

`grm` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level growth, removal, and mortality data (TREE_GRM_COMPONENT).

`tree_grm_component` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level growth, removal, and mortality data (TREE_GRM_COMPONENT).

`begin` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level growth, removal, and mortality data (TREE_GRM_BEGIN).

`tree_grm_begin` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level growth, removal, and mortality data (TREE_GRM_BEGIN).

`midpt` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level growth, removal, and mortality data (TREE_GRM_MIDPT).

`tree_grm_midpt` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Tree-level growth, removal, and mortality data (TREE_GRM_MIDPT).

`plot` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Plot data unioned with remeasured plot data.

`pltu` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Plot data unioned with remeasured plot data.

`plotu` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Plot data unioned with remeasured plot data.

`condu` DF/DT, R object, comma-delimited file(*.csv), or layer in dsn. Cond data unioned with remeasured cond data.

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Grayson W. White

`ref_codes` *Reference tables - Code definitions.*

Description

Table with variable codes (VALUE) and descriptions (MEANING).

Format

A dataframe with 7 columns, VARIABLE, VALUE, MEANING, COLORHEX, GROUP, GROUPNM, GROUPHEX.

Source

FIA look-up tables.

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_codes_archive` *Reference tables - Code definitions (Archive).*

Description

Table with variable codes (VALUE) and descriptions (MEANING).

Format

A dataframe with 7 columns, VARIABLE, VALUE, MEANING, COLORHEX, GROUP, GROUPNM, GROUPHEX.

Source

FIA look-up tables.

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_cond` *Reference table - Metadata for cond default variables output from DBgetPlots()*

Description

Data frame with variable names and descriptions

Format

A data frame with 61 rows and 3 columns VARIABLE - Variable in cond data frame DESCRIPTION - Description of variable in cond data frame TABLE - Table in database where variable originates or if derived

Source

FIA look-up table

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_conversion` *Reference table - for conversion factors.*

Description

Table with conversion factors from English to metric units.

Format

A dataframe with 6 columns: TYPE, ENGLISH, ENGLISH_ABBR, METRIC, METRIC_ABBR, CONVERSION.

Source

Conversion table.

`ref_diacl2in` *Reference table - diameter 2-inch class codes (DIA).*

Description

Table with min (MIN), max (MAX), and 2-inch class diameter codes (MEANING).

Format

A dataframe with 3 columns, MIN, MAX, and MEANING.

Source

Imported from comma-delimited file.

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_domain` *Reference table - for generating tables.*

Description

Table with row/column domain (VARNM) and their pretty names for table output (TABLENM).

Format

A dataframe with 2 columns, VARNM and TABLENM.

Source

FIA look-up table.

`ref_estimators` *Reference table - FIESTA estimators.*

Description

Table with list of estimators currently in FIESTA.

Format

A dataframe with 7 columns: ESTIMATOR, SHORTNAME, PACKAGE, OUTNAME, ESTIMATOR_TYPE, ESTIMATOR_DATA, DESCRIPTION, CITATION

Source

Comma-delimited file.

`ref_estvar` *Reference table - for generating estimates*

Description

Data frame with variable names and descriptions

Format

A data frame to use a reference for estimation variables and filters.

ref_evaltyp *Reference table - for generating tables.*

Description

Table with row/column domain (VARNM) and their pretty names for table output (TABLENM).

Format

A dataframe with 3 columns, EVAL_TYP_CD, EVAL_TYP, DESCRIPTION.

Source

FIA look-up table.

ref_plt *Reference table - Metadata for plt default variables output from DBgetPlots()*

Description

Data frame with variable names and descriptions.

Format

A data frame with 43 rows and 3 columns VARIABLE - Variable in plt data frame DESCRIPTION - Description of variable in plt data frame TABLE - Table in database where variable originates or if derived

Source

FIA look-up table

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_popType` *Reference table - popType codes.*

Description

Table with population type (popType) and associated evaluation code (EVAL_TYP_CD).

Format

A dataframe with 2 columns, VARNM and TITLE.

Source

Comma-delimited file.

`ref_shp` *Reference table - Metadata for shp_* default variables output from DBgetPlots()*

Description

Data frame with variable names and descriptions

Format

A dataframe with 63 rows and 4 columns VARIABLE - Variable in plt data frame DESCRIPTION - Description of variable in plt data frame TABLE - Table in database where variable originates or if derived SHPEXPORT - Name of variable for exported shapefile (<= 10 characters)

Source

FIA look-up table

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_species` *Reference table - Code definitions.*

Description

Table with species information downloaded from datamart FIADB_REFERENCES, subset from REF_SPECIES TABLE.

Format

A dataframe with 14 columns: SPCD, COMMON_NAME, GENUS, SPECIES, SPECIES_SYMBOL, E_SPGRCD, W_SPGRPCD, C_SPGRPCD, P_SPGRPCD, MAJOR_SPGRPCD, JENKINS_TOTAL_B1, JENKINS_TOTAL_B2, DRYWT_TO_GREENWT_CONERSION, SCIENTIFIC_NAME (GENUS + SPECIES).

Source

Imported from comma-delimited file.

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_statecd` *Reference table - state codes (STATECD).*

Description

Table with state codes (VALUE), name (MEANING), abbreviation (ABBR), and UNIT.

Format

A dataframe with 4 columns, VALUE, MEANING, ABBR, UNIT.

Source

Imported from comma-delimited file.

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

`ref_titles` *Reference table - Variable titles.*

Description

Table with variable name (VARNM) and associated title (TITLE).

Format

A dataframe with 2 columns, VARNM and TITLE.

Source

Comma-delimited file.

`ref_tree` *Reference table - Metadata for tree default variables output from DBgetPlots()*

Description

Data frame with variable names and descriptions

Format

A data frame with 72 rows and 3 columns VARIABLE - Variable in tree data frame DESCRIPTION - Description of variable in tree data frame TABLE - Table in database where variable originates

Source

FIA look-up table

References

O'Connell, B.M.; LaPoint, E.B.; Turner, J.A.; Ridley, T.; Boyer, D.; Wilson, A.M.; Waddell, K.L.; Christensen, G.; Conkling, B.L. 2012. The Forest Inventory and Analysis Database: Database Description and Users Manual Version 5.1.2 for Phase 2. U.S. Department of Agriculture. (http://fia.fs.fed.us/library/database-documentation/current/ver5-2012/FIADB_user manual_5-1-2_p2_07_2012.pdf)

ref_units *Reference table - for variable units.*

Description

Table with units for TREE variables. The WOODLAND column was added to identify which variables include woodland species.The kg2tons column was added to identify which variables are commonly converted from kilograms to tons in estimation process.

Format

A dataframe with 4 columns: VARIABLE, UNITS, METRICUNITS, WOODLAND, kg2tons.

Source

Units table.

savedata_options *Data saving options.*

Description

Returns a list of user-supplied parameters and parameter values for saving data.

Usage

```
savedata_options(
  outfolder = NULL,
  out_fmt = "csv",
  outsp_fmt = "shp",
  outobj_fmt = "rds",
  out_dsn = NULL,
  out_layer = "outdat",
  outfn.pre = NULL,
  outfn.date = FALSE,
  addtitle = TRUE,
  raw_fmt = "csv",
  raw_dsn = NULL,
  overwrite_dsn = FALSE,
  overwrite_layer = TRUE,
  append_layer = FALSE,
  add_layer = TRUE,
  layer.pre = NULL,
  outconn = NULL,
  ...
)
```

Arguments

`outfolder` String. The outfolder to write files to. If NULL, files are written to working directory, or if gui=TRUE, a window to browse.

`out_fmt` String. Format for output tables (’csv’, ’sqlite’, ’gpkg’, ’gdb’).

`outsp_fmt` String. Format for output spatial (’shp’, sqlite’, ’gpkg’, ’gdb’).

`outobj_fmt` String. Format for output spatial (’rda’, ’rds’, ’llo’).

`out_dsn` String. Data source name for output. If extension is not included, out_fmt is used. Use full path if outfolder=NULL.

`out_layer` outlayer.

`outfn.pre` String. If savedata=TRUE, prefix for output files. If rawdata=TRUE, prefix for rawdata files (if raw_fmt = ’csv’) or raw_dsn (if raw_fmt != ’csv’).

`outfn.date` Logical. If TRUE, add current date to out_dsn.

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

`raw_fmt` String. Format for output rawdata tables (’sqlite’, ’gpkg’, ’csv’, ’gdb’).

`raw_dsn` String. Data source name for rawdata output. If extension is not included, out_fmt is used. Use full path if outfolder=NULL.

`overwrite_dsn` Logical. If TRUE, overwrites raw_dsn, if exists.

`overwrite_layer` Logical. If TRUE, overwrites the output. If rawdata=TRUE, overwrites out_layer in rawdata folder (if raw_fmt = ’csv’) or out_layers in raw_dsn (if raw_fmt != ’csv’).

`append_layer` Logical. If TRUE, and appends data to existing *.csv files (if *_fmt = ’csv’) or *_dsn layers (if *_fmt != ’csv".

`add_layer` Logical. If TRUE, adds layer to an existing out_dsn (if out_fmt != c(’csv’,’shp’)).

`layer.pre` Layer prefix.

`outconn` Open database connection to save to.

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for saving data.

Author(s)

Grayson W. White

Examples

```
savedata_options(outfolder = "path", overwrite_dsn = FALSE)
```

spMakeSpatial_options *Make SpatialPoints options*

Description

Returns a list of user-supplied parameters and parameter values for making SpatialPoints.

Usage

```
spMakeSpatial_options(
  xvar = NULL,
  yvar = NULL,
  xy.crs = 4269,
  prj = NULL,
  datum = NULL,
  zone = NULL,
  zoneS = FALSE,
  aea.param = "USGS",
  ...
)
```

Arguments

`xvar` String. Name of variable in xyplt defining x coordinate.

`yvar` String. Name of variable in xyplt defining y coordinate.

`xy.crs` PROJ.4 String or CRS object or Integer EPSG code defining Coordinate Reference System. (e.g., EPSG:4269-Geodetic coordinate system for North America, NAD83).

`prj` String. Projection, or coordinate system of the X/Y coordinates ("longlat", "utm", "aea"). If other, include PROJ.4 string in prj4str.

`datum` String. Datum of projection ("WGS84", "NAD83", "NAD27").

`zone` Integer. If prj="utm", the UTM zone.

`zoneS` Logical. If prj="utm", if the UTM zone is in the Southern hemisphere.

`aea.param` String. If prj="aea", the associated lat/lon parameters (USGS: " +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0 +y_0=0"). If other, include PROJ.4 string in prj4str.

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Grayson W. White

Examples

```
spMakeSpatial_options()
```

`strata_options` *Strata options.*

Description

Returns a list of user-supplied parameters and parameter values for strata.

Usage

```
strata_options(
  getwt = FALSE,
  getwtvar = "P1POINTCNT",
  strwtvar = "strwt",
  stratcombine = TRUE,
  minplotnum.strat = 2,
  pivot = FALSE,
  nonresp = FALSE,
  ...
)
```

Arguments

`getwt` Logical. If TRUE, calculates strata weights from stratatlut getwtvar. If FALSE, strwtvar variable must be in stratalut.

`getwtvar` String. If getwt=TRUE, name of variable in stratalut to calculate weights (Default = 'P1POINTCNT').

`strwtvar` String. If getwt=FALSE, name of variable in stratalut with calculated weights (Default = 'strwt').

`stratcombine` Logical. If TRUE, and strata=TRUE, automatically combines strata categories if less than minplotnum.strat plots in any one stratum. See notes for more info.

`minplotnum.strat` Integer. Minimum number of plots for a stratum within an estimation unit.

`pivot` Logical. If TRUE, pivot stratalut.

`nonresp` Deprecated.

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Grayson W. White

Examples

```
strata_options(getwt = FALSE)
```

---

stunitco *SpatialPolygonsDataFrame with FIA state, unit, county codes and names*

Description

Polygon feature class with state and county boundaries defined by Census Bureau, including Federal Information Processing Standards (FIPS) codes. The FIA Survey Unit code and name attributes (UNITCD, UNITNM) were appended to dataset, with joining columns of STATECD and COUNTYCD.

Format

A SpatialPolygonsDataFrame with 3233 features and 8 attributes RS - FIA Research Station name RSCD - FIA Research Station code STATECD - FIPS state code STATENM - FIPS state name STATEAB - FIPS state abbreviation UNITCD - FIA survey unit code UNITNM - FIA survey unit name COUNTYCD - FIPS county code COUNTYNM - FIPS county name

Details

Derived from cb_2018_us_county_5m. STATEFP was converted to numeric and named STATECD COUNTYFP was converted to numeric and named COUNTYCD Lookup table for FIA Research Station (REF_RESEARCH_STATION) was downloaded from FIA DataMart on 20191105 (FIADB_1.6.1.00) and joined by STATECD. A lookup table for UNITCD was created from plot data using unique STATECD, COUNTYCD, UNITCD and joined to table.

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

Source

Downloaded from the United States Census Bureau on 2019 November 3, format Esri Shapefile (https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.html) Projection: Geographic (GCS_North_American_1983) EPSG: 4269

---

tableIDs *List of FIADB table unique IDs.*

Description

Returns a list of user-supplied parameters and parameter values for data table unique IDs to be supplied to *pop functions.

Usage

```
tableIDs(
  cond = "PLT_CN",
  plt = "CN",
  tree = "PLT_CN",
  seedling = "PLT_CN",
  subplot = "PLT_CN",
  subp_cond = "PLT_CN",
  condid = "CONDID",
  subpid = "SUBP",
  ...
)
```

Arguments

`cond`	String. Unique identifier of plot in cond.
`plt`	String. Unique identifier of plot in plt.
`tree`	String. Unique identifier of plot in tree.
`seedling`	String. Unique identifier of plot in seedling.
`subplot`	String. Unique identifier of plot in subplot.
`subp_cond`	String. Unique identifier of plot in subp_cond.
`condid`	String. Unique identifier of a condition in cond.
`subpid`	String. Unique identifier of a subplot in subplot and subp_cond.
`...`	For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied unique identifier of a plot in tables.

Author(s)

Tracey S. Frescino

table_options *Table aesthetics and output options.*

Description

Returns a list of user-supplied parameters and parameter values for outputting tables with custom aesthetics.

Usage

```
table_options(
  row.FIAname = FALSE,
  col.FIAname = FALSE,
  row.orderby = NULL,
  col.orderby = NULL,
  row.add0 = FALSE,
  col.add0 = FALSE,
  rowlut = NULL,
  collut = NULL,
  row.classify = NULL,
  col.classify = NULL,
  rawonly = FALSE,
  raw.keep0 = FALSE,
  rowgrp = FALSE,
  rowgrpnm = NULL,
  rowgrpord = NULL,
  totals = TRUE,
  allin1 = FALSE,
  metric = FALSE,
  estround = 1,
  pseround = 2,
  estnull = "--",
  psenull = "--",
  row.NAname = "Other",
  col.NAname = row.NAname,
  divideby = NULL,
  spcdname = "COMMON_SCIENTIFIC",
  ...
)
```

Arguments

`row.FIAname` Logical. If TRUE, retrieves default FIA reference names for rowvar located in ref_codes data frame. Names are only available for certain variables (Check sort(unique(ref_codes$VARIABLE)) for available names. If row.FIAname = TRUE and rowvar is in ref_codes, the rowvar name is used for the output table, and the rowvar code is used to sort.

`col.FIAname` Logical. If TRUE, retrieves default FIA reference names for colvar located in ref_codes data frame. Names are only available for certain variables. Check: sort(unique(ref_codes$VARIABLE)) for available names. If col.FIAname = TRUE and rowvar is in ref_codes, the colvar name is used for the output table, and the colvar code is used to sort.

`row.orderby` String. Optional. Name of variable to sort table rows. Both the rowvar and row.orderby variables must be included in the same input data.frame. if NULL, and row.FIAname=FALSE or rowvar is not in ref_codes, the rows are ordered by rowvar.

`col.orderby` String. Optional. Name of variable to sort table columns. Both the colvar and col.orderby variables must be included in the same input data.frame. if NULL, and col.FIAname=FALSE or colvar is not in ref_codes, the columns are ordered by colvar.

`row.add0` Logical. If TRUE, include rows with 0 values to the output table.

`col.add0` Logical. If TRUE, include columns with 0 values to the output table.

`rowlut` Data frame. A lookup table with variable codes and code names to include as rows of output table (See notes for more information and format).

`collut` Data frame. A lookup table with variable codes and code names to include as columns of output table (See notes for more information and format).

`row.classify` Data frame (if categorical) or Vector (if continuous). If clasifying categories, input a dataframe with two columns ('FROM' and 'TO'). If classifying continuous values, input a vector of class breaks for row

`col.classify` Data frame (if categorical) or Vector (if continuous). If clasifying categories, input a dataframe with two columns ('FROM' and 'TO'). If classifying continuous values, input a vector of class breaks for column

`rawonly` Logical. If TRUE, only rawdata are output. If dataset includes many estimation units, and only raw data tables are desired, it is more efficient to output raw data only.

`raw.keep0` Logical. If TRUE, keep 0 values in raw data tables.

`rowgrp` Logical. If TRUE, appends row groups to first column of table. Only available if group category exists in ref_codes table or defined in rowgrpnm (e.g., FORTYPGRPCD, OWNGRPCD).

`rowgrpnm` String. Name of variable for grouping rowvar. Variable must be included in same input table as rowvar.

`rowgrpord` String. Name of variable to sort row group variable. Variable must be included in same input table as rowgrpnm.

`totals` Logical. If TRUE, returns total estimate (mean * AREAUSED).

`allin1` Logical. If TRUE, both estimates and percent sample error are output in one table as: estimates (percent sample error).

`metric` Logical. If TRUE, output if returned in metric units.

`estround` Integer. Number of decimal places for estimates.

`pseround` Integer. Number of decimal places for percent sampling error.

`estnull` Number or character. The number or symbol to use to indicate ’not sampled’ for estimate.

`psenull` Number or character. The number or symbol to use to indicate ’not sampled’ for percent standard error.

`row.NAname` String. The name to use for NA values for rows.

`col.NAname` String. String. The name to use for NA values for columns.

`divideby` String. Conversion number for output (’hundred’, ’thousand’, ’million’).

`spcdname` String. Type of name to use for species in tables (’COMMON’, ’SCIENTIFIC’, SYMBOL’, ’COMMON_SCIENTIFIC’, ’NONE’).

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for outputting tables with custom aesthetics.

Author(s)

Grayson W. White

Examples

```
table_options(row.FIAname = TRUE, col.FIAname = TRUE)
```

title_options *Title output options.*

Description

Returns a list of user-supplied parameters and parameter values for outputting title with custom aesthetics.

Usage

```
title_options(
  title.main = NULL,
  title.ref = NULL,
  title.rowvar = NULL,
  title.colvar = NULL,
  title.unitvar = NULL,
  title.estvar = NULL,
  title.estvarn = NULL,
  title.filter = NULL,
  title.units = "acres",
  ...
)
```

Arguments

`title.main` String. TITLE, if savedata=TRUE and/or returntitle=TRUE: the complete title used for table. If title.main=NULL, the title.* parameters are used to generate title string. Note: if title.ref is not NULL, it is added to title.main.

`title.ref` String. TITLE, if savedata=TRUE and/or returntitle=TRUE: the ending text of the table title (e.g., Nevada, 2004-2005). If NULL, = "".

`title.rowvar` String. TITLE, if savedata=TRUE and/or returntitle=TRUE: pretty name for the row domain variable. If NULL, = rowvar.

`title.colvar` String. TITLE, if savedata=TRUE and/or returntitle=TRUE: pretty name for the column domain variable. If NULL, = colvar.

`title.unitvar` String. TITLE, if savedata=TRUE and/or returntitle=TRUE: pretty name for the estimation unit variable. If NULL, = unitvar.

`title.estvar` String. TITLE: if savedata=TRUE and/or returntitle=TRUE: pretty name for the estimate variable. If NULL, title.estvar = estvar.name.

`title.estvarn` String. TITLE: if savedata=TRUE and/or returntitle=TRUE: pretty name for the estimate variable. If NULL, title.estvar = estvar.name.

`title.filter` String. TITLE, if savedata=TRUE and/or returntitle=TRUE: pretty name for filter(s). If title.filter=NULL, a default is generated from cfilter. If title.filter="", no title.filter is used.

`title.units` String.

`...` For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for outputting titles with custom aesthetics.

Author(s)

Grayson W. White

Examples

```
title_options(title.main = "My fancy title", title.estvar = "Estimate title")
```

unit_options *Unit options.*

Description

Returns a list of user-supplied parameters and parameter values for unit.

Usage

```
unit_options(
  unitvar2 = NULL,
  areaunits = "acres",
  minplotnum.unit = 10,
  unit.action = "keep",
  npixelvar = "npixels",
  ...
)
```

Arguments

`unitvar2`	String. Name of a second level estimation unit variable in unitarea and cond or pltassgn with assignment for each plot (e.g., 'STATECD').
`areaunits`	String. Units of areavar in unitarea ('acres', 'hectares').
`minplotnum.unit`	Integer. Minimum number of plots for estimation unit.
`unit.action`	String. What to do if number of plots in an estimation unit is less than minplotnum.unit ('keep', 'remove' 'combine'). If unit.action='keep', estimation units with less that minplotnum.unit will be kept in output tables; if unit.action='remove', the estimation units with less that minplotnum.unit will be removed from the output tables; and if unit.action='combine', combines estimation unit to the following estimation unit, ordered in stratalut or unitzonal.
`npixelvar`	String. Name of variable in unitlut defining number of pixels by estimation unit.
`...`	For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Grayson W. White

Examples

```
unit_options()
```

`xy_options` *List of population tables.*

Description

Returns a list of user-supplied parameters and parameter values for data xyuation (FIA or custom) extraction to be supplied to *DB functions.

Usage

```
xy_options(
  xy.uniqueid = "CN",
  xvar = "LON",
  yvar = "LAT",
  xy.crs = 4269,
  xyjoinid = NULL,
  ...
)
```

Arguments

`xy.uniqueid`	String. Unique identifier of xy.
`xvar`	String. Name of variable in xy defining x coordinate.
`yvar`	String. Name of variable in xy defining y coordinate.
`xy.crs`	PROJ.4 String or CRS object or Integer EPSG code defining Coordinate Reference System.
`xyjoinid`	String. Name of variable in xy to join to plot data. If NULL, xyjoinid = xy.uniqueid.
`...`	For extendibility.

Details

If no parameters, an empty list is returned.

Value

A list of user-supplied parameters and parameter values for strata.

Author(s)

Tracey S. Frescino

Examples

```
xy_options(xvar="LON", yvar="LAT")
```

Index