Package: metagear (via r-universe)

September 6, 2024

Version 0.7

Date 2021-02-15

Title Comprehensive Research Synthesis Tools for Systematic Reviews and Meta-Analysis

Description Functionalities for facilitating systematic reviews, data extractions, and meta-analyses. It includes a GUI (graphical user interface) to help screen the abstracts and titles of bibliographic data; tools to assign screening effort across multiple collaborators/reviewers and to assess inter- reviewer reliability; tools to help automate the download and retrieval of journal PDF articles from online databases; figure and image extractions from PDFs; web scraping of citations; automated and manual data extraction from scatter-plot and bar-plot images; PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) flow diagrams; simple imputation tools to fill gaps in incomplete or missing study parameters; generation of random effects sizes for Hedges' d, log response ratio, odds ratio, and correlation coefficients for Monte Carlo experiments; covariance equations for modelling dependencies among multiple effect sizes (e.g., effect sizes with a common control); and finally summaries that replicate analyses and outputs from widely used but no longer updated meta-analysis software (i.e., metawin). Funding for this package was supported by National Science Foundation (NSF) grants DBI-1262545 and DEB-1451031. CITE: Lajeunesse, M.J. (2016) Facilitating systematic reviews, data extraction and meta-analysis with the metagear package for R. Methods in Ecology and Evolution 7, 323-330 <doi:10.1111/2041-210X.12472>.

Type Package

Depends R (>= 3.3.2)

Imports Matrix, MASS, metafor (>= 1.9-9), stringr

SystemRequirements Tcl/Tk toolkit (X11 Quarts for Mac)

VignetteBuilder R.rsp

License GPL (>= 2)

URL http://lajeunesse.myweb.usf.edu/ https://github.com/mjlajeunesse/
https://www.youtube.com/c/LajeunesseLab/

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Suggests EBImage, ape, hexView, RCurl, testthat, R.rsp

Repository https://mjlajeunesse.r-universe.dev

RemoteUrl https://github.com/mjlajeunesse/metagear

RemoteRef HEAD

RemoteSha 37354985d3e48c8e574bf7b1592bdb8122d8cc6d

Contents

`metagear-package` *Research synthesis tools to facilitate systematic reviews, data extraction, and meta-analysis.*

Description

metagear is a comprehensive, multifunctional toolbox with capabilities aimed to cover much of the research synthesis taxonomy: from applying a systematic review approach to objectively assemble and screen the literature, to extracting data from studies, and to finally summarize and analyze these data with the statistics of meta-analysis. More information about **metagear** can be found at `http://lajeunesse.myweb.usf.edu/`.

Details

What to cite?

Lajeunesse, M.J. (2016) Facilitating systematic reviews, data extraction and meta-analysis with the metagear package for R. *Methods in Ecology and Evolution* 7: 323-330. [download here]

Installation and Dependencies.

metagear has one external dependency that need to be installed and loaded prior to use in R. This is the EBImage R package (Pau et al. 2010) available only from the Bioconductor repository: `https://www.bioconductor.org/`.

To properly install **metagear**, start with the following R script that loads the Bioconductor resources needed to install the EBImage (also accept all of its dependencies):

```
install.packages("BiocManager");
BiocManager::install("EBImage"))
library(metagear)
```

Finally for Mac OS users, installation is sometimes not straighforward as the abstract_screener() requires the Tcl/Tk GUI toolkit to be installed. You can get this toolkit by making sure the latest X11 application (xQuartz) is installed from here: `https://www.xquartz.org/`.

Author(s)

Marc J. Lajeunesse (University of South Florida, Tampa USA)

References

Pau, G., Fuchs, F., Sklyar, O., Boutros, M. and Huber, W. (2010) EBImage: an R package for image processing with applications to cellular phenotypes. Bioinformatics 26: 979-981.

`abstract_screener` *A GUI screener to quickly code candidate studies for inclusion/exclusion into a systematic review or meta-analysis.*

Description

A GUI screener to help scan and evaluate the title and abstract of studies to be included in a systematic review or meta-analysis.

Usage

```
abstract_screener(
  file = file.choose(),
  aReviewer = NULL,
  reviewerColumnName = "REVIEWERS",
  unscreenedColumnName = "INCLUDE",
  unscreenedValue = "not vetted",
  abstractColumnName = "ABSTRACT",
  titleColumnName = "TITLE",
  browserSearch = "https://www.google.com/search?q=",
  fontSize = 13,
  windowWidth = 70,
  windowHeight = 16,
  theButtons = c("YES", "maybe", "NO"),
  keyBindingToButtons = c("y", "m", "n"),
  buttonSize = 10,
  highlightColor = "powderblue",
  highlightKeywords = NA
)
```

Arguments

`file`	The file name and location of a .csv file containing the abstracts and titles. The .csv file should have been initialized with `effort_initialize` and populated with screeners (reviewers) using `effort_distribute`.
`aReviewer`	The name (a string) of the reviewer to screen abstracts. It is used when there are multiple reviewers assigned to screen abstracts. The default column label is "REVIEWERS" as initialized with `effort_distribute`.
`reviewerColumnName`	The name of the column heading in the .csv file that contains the reviewer names that will screen abstracts. The default column label is "REVIEWERS".
`unscreenedColumnName`	The name of the column heading in the .csv file that contains the screening outcomes (i.e. vetting outcomes by a reviewer). Unscreened references are by default labeled as "not vetted". The reviewer then can code to "YES" (is a relevant study), "NO" is not relevant and should be excluded, or "MAYBE" if the title/abstract is missing or does not contains enough information to fully assess inclusivity. The default label of this column is "INCLUDE".
`unscreenedValue`	Changes the default coding (a string) of "not vetted" that designates whether an abstract remains to be screened or vetted.
`abstractColumnName`	The name of the column heading in the .csv file that contains the abstracts. The default label of this column is "ABSTRACT".
`titleColumnName`	The name of the column heading in the .csv file that contains the titles. The default label of this column is "TITLE".
`browserSearch`	Change the url for the browser title search; the default is Google.
`fontSize`	Change the font gWidgets::size of the title and abstract text.
`windowWidth`	Change the default width of the GUI window.
`windowHeight`	Change the default height of the GUI window.
`theButtons`	A vector of coding buttons included on the screener. The default is YES, maybe, and NO. Buttons can be removed as added by changing this vector. For example, theButtons = c("YES", "NO") to remove the maybe-button, or theButtons = c("YES", "maybe", NO", "model") to add a "model" button that tags studies specifically as "model".
`keyBindingToButtons`	A vector of specific keyboard bindings to buttons. They are keyboard shortcuts to buttons and the default binding is y for YES-button, m for maybe-button, and n for NO-button. If theButtons parameter is modified then these keybindings should also be modified.
`buttonSize`	Change the default gWidgets::size of buttons.
`highlightColor`	The color of keywords highlighted in title and abstract. The default is blue, but for classic yellow use "palegoldenrod".
`highlightKeywords`	A string or list of keywords that will be highlighted in title and abstract.

Note

Installation and troubleshooting

For Mac OS users, installation is sometimes not straighforward as this screener requires the Tcl/Tk GUI toolkit to be installed. You can get this toolkit by making sure the latest X11 application (xQuartz) is installed, see here: https://www.xquartz.org/. More information on installation is found in `metagear`'s vignette.

How to use the screener

The GUI itself will appear as a single window with the first title/abstract listed in the .csv file. If abstracts have already been screened/coded, it will begin at the nearest reference labeled as "not vetted". The SEARCH WEB button opens the default browser and searches Google with the title of the reference. The YES, MAYBE, NO buttons, which also have keyboard shortcuts y and n, are used to code the inclusion/exclusion of the reference. Once clicked/coded the next reference is loaded. The SAVE button is used to save the coding progress of screening tasks. It will save coding progress directly to the loaded .csv file. **Closing the GUI, and not saving, will result in the loss of screening efforts relative to last save.**

There is also an ISSUE FIXES menu bar with quick corrections to screening errors. These include ISSUE FIXES: REFRESH TITLE AND ABSTRACT TEXT which reloads the text of the current abstract in case portions were deleted when copying and pasting sections, ISSUE FIXES: STATUS OF CURRENT ABSTRACT which provides information on whether or not the abstract was previously screened, and ISSUE FIXES: RETURN TO PREVIOUS ABSTRACT that backtracks to the previous abstract if a selection error occurred (note a warning will appear of there is a change to its inclusion/exclusion coding).

Examples

```
## Not run:

data(example_references_metagear)
effort_distribute(example_references_metagear,
                  initialize = TRUE,
                  reviewers = "marc",
                  save_split = TRUE)
abstract_screener("effort_marc.csv",
                  aReviewer = "marc",
                  highlightKeywords = "and")

## End(Not run)
```

`browse_DOI` *Opens a web page associated with a DOI (digital object identifier).*

Description

Uses the DOI name of a study reference to locate the e-journal website, or reference/citation website in Web of Science, Google Scholar, or CrossRef. Opens in default web-browser.

Usage

```
browse_DOI(theDOI, host = "DOI")
```

Arguments

`theDOI`	A string that identifies an electronic document on the web.
`host`	A string that defines the domain link used to open the DOI. The default, `"DOI"`, will open to the web page associated with the DOI (e.g., publisher website). Other options include `"WOS"` that will open the DOI in Web of Science, `"GS"` in Google Scholar, and `"CRF"` in Crossref.

Examples

```
## Not run:

browse_DOI("10.1086/603628")

## End(Not run)
```

`chachi` *A small tribute to Chachi.*

Description

Rest easy little bud, 200?-2016.

Usage

```
chachi()
```

covariance_commonControl *Generates a sampling variance-covariance matrix for modeling dependencies among effect sizes due to sharing a common control.*

Description

Generates K by K sampling variance-covariance (VCV) matrix that models the dependencies that arise due to using the same control group study parameters when estimating multiple effect sizes. This VCV matrix can then be used in meta-analysis. Currently only supports VCV calculation for log response ratios (see Lajeunesse 2011).

Usage

```
covariance_commonControl(
  aDataFrame,
  control_ID,
  X_t,
  SD_t,
  N_t,
  X_c,
  SD_c,
  N_c,
  metric = "RR"
)
```

Arguments

`aDataFrame`	A data frame containing columns with all study parameters used to estimate effect sizes (e.g., means, SD, N's for treatment and control groups). Must also contain a column that codes which effect sizes share a common control. See example below.
`control_ID`	Label of the column that codes groups of effect sizes that share the mean, SD, and N of a control group.
`X_t`	Column label for the means of (t)reatment group used to estimate the effect size.
`SD_t`	Column label for the standard deviations (SD) of the treatment group used to estimate the effect size.
`N_t`	Column label for the sample size (N) of the treatment group used to estimate the effect size.
`X_c`	Column label for the means of (c)ontrol group used to estimate the effect size.
`SD_c`	Column label for the standard deviations (SD) of the control group used to estimate the effect size.
`N_c`	Column label for the sample size (N) of the control group used to estimate the effect size.
`metric`	Option to designate which effect size metric for which the common control VCV matrix is to be estimated. Default is "RR" for log response ratio.

Value

A K by K sampling variance-covariance matrix and a data frame aligned with the block diagonal design of the sampling matrix.

Note

Response Ratio's (RR) with a comon control group

Following Lajeunesse (2011), when two (or more) reponse ratio (RR) effect sizes share a common control mean ($\bar{X}_C$), such as $RR_{A,C} = ln(\bar{X}_A/\bar{X}_C)$ and $RR_{B,C} = ln(\bar{X}_B/\bar{X}_C)$, then they share a sampling covariance of:

$$cov(RR_{A,C},\ RR_{B,C}) = \frac{(SD_C)^2}{N_C \bar{X}_C^2},$$

where the SD and N are the standard deviation and sample size of $\bar{X}_C$, respectively.

References

Lajeunesse, M.J. 2011. On the meta-analysis of response ratios for studies with correlated and multi-group designs. Ecology 92: 2049-2055.

`effort_distribute` *Assigns title/abstract screening efforts to a team.*

Description

Randomly distributes screening tasks evenly or unevenly across multiple team members. It populates this effort in a data frame column that includes this screening work (e.g., ABSTRACTS and TITLES).

Usage

```
effort_distribute(
  aDataFrame = NULL,
  dual = FALSE,
  reviewers = NULL,
  column_name = "REVIEWERS",
  effort = NULL,
  initialize = FALSE,
  save_split = FALSE,
  directory = getwd()
)
```

Arguments

`aDataFrame`	A data.frame containing the titles and abstracts to be screened by a team. The default assumes that the data.frame has already been formatted using `effort_initialize`. This data.frame will be populated with screening efforts. See example: `example_references_metagear`
`dual`	When `TRUE`, distributes effort using a dual screening design where two members will screen the same random collection of titles/abstracts. Requires the team to have an even number of members.
`reviewers`	A vector with the names of each team member.
`column_name`	Changes the default label of the "REVIEWERS" column that contains the screening efforts of each team member.
`effort`	A vector of percentages used to allocate screening tasks among each team member. When not called explicitly, assumes effort to be distributed evenly among all members. Must be the same length as the number of team members, and also sum to 100.
`initialize`	When `TRUE`, initializes the data.frame so that efforts could be distributed, calls: `effort_initialize`. Default is `FALSE`.
`save_split`	Saves the allocated team effort into separate effort_*.csv files for individual screening tasks. These files can be given to each member to screen their random title/abstract subset. All files can be merged once all screening tasks have been completed using `effort_merge`.
`directory`	Changes the default location/directory for where the effort_*.csv will be saved. If not explicitly called, it will deposit files in the current working directory.

Value

A data.frame with title/abstract screening efforts randomly distributed across a team.

See Also

`effort_initialize`, `effort_merge`, `effort_summary`

Examples

```
## Not run:

data(example_references_metagear)
theTeam <- c("Christina", "Luc")
effort_distribute(example_references_metagear, initialize = TRUE, reviewers = theTeam)

## End(Not run)
```

`effort_initialize` *Formats a reference dataset for title/abstract screening efforts.*

Description

Adds columns with standardized labels to a data framw with bibliographic data on journal articles. These columns will be used to assign reviewers, implementation of dual screening design, and the coding of inclusion/exclusions screening decisions.

Usage

```
effort_initialize(
  aDataFrame,
  study_ID = TRUE,
  unscreenedValue = "not vetted",
  dual = FALSE,
  front = TRUE
)
```

Arguments

`aDataFrame`	A data.frame object that includes the titles and abstracts to be screened. It will be formatted for screening efforts. See example: `example_references_metagear`
`study_ID`	When `FALSE`, does not add a column "STUDY_ID" that includes a unique identification number for each reference (row) in the dataFrame.
`unscreenedValue`	Changes the default coding (a string) of "not vetted" that designates whether an abstract remains to be screened or vetted as part of the "INCLUDE" column.
`dual`	When `TRUE`, formats dataFrame for a dual screening (paired) design. Creates two reviewer teams: REVIEWERS_A and REVIEWERS_B.
`front`	When `FALSE`, adds new columns to the back end of the dataframe. When `TRUE`, adds columns to the front.

Value

A data.frame formatted for title/abstract screening efforts.

See Also

`effort_distribute`, `effort_merge`, `effort_summary`

Examples

```
data(example_references_metagear)
effort_initialize(example_references_metagear)
```

`effort_merge` *Merges multiple files that had title/abstract screening efforts distributed across a team.*

Description

Combines (merges) multiple effort_*.csv files within the same directory that represent the completed screening efforts of multiple team members. These files were originally generated with `effort_distribute`.

Usage

```
effort_merge(directory = getwd(), reviewers = NULL, dual = FALSE)
```

Arguments

`directory`	The directory name for the location of multiple .csv files. Assumes the current working directory if none is explicitly called. File names must include the "effort_" string as originally generated by `effort_distribute`.
`reviewers`	A vector of reviewer names (strings) used to merge effort from a select group of team members. Must be an even collection (e.g., pairs of reviewers) when a dual design was implemented.
`dual`	When `TRUE`, merges files implementing a dual screening design.

Value

A single data.frame merged from multiple files.

See Also

`effort_initialize`, `effort_distribute`, `effort_summary`

Examples

```
## Not run:

data(example_references_metagear)
theTeam <- c("Christina", "Luc")
# warning effort_distribute below, will save two files to working
# directory: effort_Christina.csv and effort_Luc.csv
effort_distribute(example_references_metagear, initialize = TRUE,
                  reviewers = theTeam, save_split = TRUE)
effort_merge()

## End(Not run)
```

`effort_redistribute` *Redistributes title/abstract screening efforts among a review team.*

Description

Randomly re-distributes screening tasks from one reviewers to the rest of the reviewing team. Used when screening effort needs to be re-allocated among reviewing team members.

Usage

```
effort_redistribute(
  aDataFrame,
  column_name = "REVIEWERS",
  reviewer = NULL,
  remove_effort = 100,
  reviewers = NULL,
  effort = NULL,
  save_split = FALSE,
  directory = getwd()
)
```

Arguments

`aDataFrame`	A data.frame containing the titles and abstracts to be screened by a team. The default assumes that the data.frame has already been formatted using `effort_initialize` and populated with `effort_distribute`.
`column_name`	Changes the default label of the "REVIEWERS" column that contains the screening efforts of each team member.
`reviewer`	The name of the reviewer whose effort is to be redistributed.
`remove_effort`	The percentage of effort to be redistributed among the team. The default is that 100% of the effort will be re-distributed.
`reviewers`	A vector of the names of each team member that will take on additional work.
`effort`	A vector of percentages used to allocate screening tasks among each team member. When not called explicitly, assumes effort to be distributed evenly among all members. Must be the same length as the number of team members, and also sum to 100.
`save_split`	Saves the allocated team effort into separate "effort_*.csv" files for individual screening tasks. These files can be given to each member to screen their random title/abstract subset. All files can be merged once all screening tasks have been completed using `effort_merge`.
`directory`	Changes the default location/directory for where the "effort_*.csv" will be saved. If not explicitly called, it will deposit files in the current working directory.

Value

A single data.frame with effort re-allocated among team members.

`effort_summary` *Provides a text summary of screening efforts among the reviewing team.*

Description

Summarizes the number of studies screened, which were identified to be included/excluded from the project, as well as those with conflicting agreement on their inclusion/exclusion. If a dual (paired) design was implemented to screen references, then it also provides inter-reviewer agreement estimate following Cohen (1960) that describes the agreement (or repeatability) of screening/coding decisions. The magnitudes of inter-reviewer agreement estimates are then interpreted following Landis & Koch (1977).

Usage

```
effort_summary(
  aDataFrame,
  column_reviewers = "REVIEWERS",
  column_effort = "INCLUDE",
  dual = FALSE,
  quiet = FALSE
)
```

Arguments

`aDataFrame`	A data.frame containing the titles and abstracts that were screened by a team. The default assumes that the data.frame is the merged effort across the team using `effort_merge`.
`column_reviewers`	Changes the default label of the "REVIEWERS" column that contains the screening efforts of each team member.
`column_effort`	Changes the default label of the "INCLUDE" column that contains the screening decisions (coded references) of each team member.
`dual`	When `TRUE`, provides a summary of the dual screening effort as well as estimation of inter-reviewer agreements following Cohen's (1960) kappa (K) and Landis and Koch's (1977) interpretation benchmarks.
`quiet`	When `TRUE`, does not print to console the summary table.

Value

A data frame with summary information on the screening tasks of a reviewing team.

References

Cohen, J. 1960. A coefficient of agreement for nominal scales. Educational and Psychological Measurement 20: 37-46.

Landis, J.R., and Koch, G.G. 1977. The measurement of observer agreement for categorical data. Biometrics 33: 159-174.

See Also

`effort_initialize`, `effort_distribute`, `effort_merge`

`example_references_metagear`

A collection of bibliographic references

Description

An example dataset containing 11 journal references. The variables are described below.

Usage

```
data(example_references_metagear)
```

Format

A data frame with 12 rows and 9 variables.

Details

- AUTHORS. Authors of the journal article
- YEAR. Publication year
- TITLE. Article title
- JOURNAL. Journal name
- VOLUME. Journal volume number
- LPAGES. Lower page number
- UPAGES. Upper page number
- DOI. Digital object identifier (DOI) of journal article
- ABSTRACT. Full text of the journal article abstract

`figure_add` *Manually add/detect points to a scatter plot figure.*

Description

Allows for the user to manually add an unlimited number of points to a figure image, by left-clicking over a figure's point. Click on the red upper-right box called "EXIT" to end recording the position of manually detected points.

Allows for the user to manually add an unlimited number of points to a figure image, by left-clicking over a figure/image point. Click on the red upper-right box called "EXIT" to end recording the position of manually detected points.

Usage

```
figure_add(file = file.choose(), color = "#009900", size = 0.03)
```

```
figure_add(file = file.choose(), color = "#009900", size = 0.03)
```

Arguments

`file`	The file name and location of a figure. Prompts for file name if none is explicitly called. Can also be a binary figure image with detected points (an `EBImage` object). See: `figure_detectAllPoints`
`color`	The color to paint the manually detected points; default is green.
`size`	The radius of the painted points.

Value

A data frame with detected points.

A data frame with detected points.

See Also

figure_detectAllPoints

figure_detectAllPoints

`figure_barPlot` *Detect and display all bar plot objects.*

Description

Automated detection of grouped data displayed in a bar-plot/chart figure image. The default returns these detected objects as an `EBImage` raster image, and as a vector of all the estimated lengths that are proportional to the values presented on each bar (and their error bars, if they are present). Note that the extracted points will be sorted by their positioning on the X-axis (or Y if the plot is a horizontal bar plot). For example, if there were error bars in the figure these will be grouped with the detected bar column. However, within these X-axis positioning they will not be sorted. See vignette for worked several illustrations.

Usage

```
figure_barPlot(
  file = file.choose(),
  horizontal = FALSE,
  binary_threshold = 0.6,
  axis_thickness = 3,
  axis_sensitivity = 0.2,
  axis_length = 0.75,
  axis_X_color = "#00ABAB",
  axis_Y_color = "#B0D36A",
  Y_min = 0,
  Y_max = 100,
  bar_width = 9,
  bar_sensitivity = 0.1,
  point_color = "#0098B2",
  point_size = 9,
  ignore = FALSE
)
```

Arguments

`file`	The file name and location of a bar-plot figure. Prompts for file name if none is explicitly called.
`horizontal`	If `TRUE` then aims to detect objects from a bar-plot that depicts data horizontally (rather than vertically).
`binary_threshold`	A proportion from zero to one designating the gray-scale threshold to convert pixels into black or white. Pixel intensities below the proportion will be converted to black, and those above white.
`axis_thickness`	An integer used to designate the thickness of the axis lines on a figure. Close alignment to the thickness of the axis on a figure will improve axis detection.

`axis_sensitivity`	A value designating the sensitivity of identifying straight lines on figure. A smaller number results in a higher sensitivity to identify axes.
`axis_length`	The relative size of the axis to the figure. The default is that axis lengths are 0.75 (75 percent) the size of the figure. This option is necessary since bar lengths may be similar to the axis length. Values should range between zero and one.
`axis_X_color`	The color to paint the detected X-axis.
`axis_Y_color`	The color to paint the detected Y-axis.
`Y_min`	The minimum Y value displayed on the Y-axis (used to scale detected data points).
`Y_max`	The maximum Y value displayed on the Y-axis (used to scale detected data points).
`bar_width`	An integer value designating the width of vertical lines on bars. A smaller number should be used when the width of bars are small (as well as the width of error bars).
`bar_sensitivity`	A value designating the sensitivity of identifying the vertical lines on bars. A smaller number should be used when the thickness of bars are small (as well as the width of error bars).
`point_color`	The color to paint the circles identifying the detected levels on bar columns and error bars.
`point_size`	An integer used to designate the size of the points painting the detected bars on a figure.
`ignore`	When `TRUE` does not display painted image with detections, only returns the data frame with detected points.

Value

A vector of scaled lengths for detected column and error bars.

`figure_detectAllPoints`

Automated detection of plotted points from a scatter-plot figure image.

Description

Attempts to detect all points of a certain shape and size from a scatter-plot figure image (even those lying outside of the axis range).

Usage

```
figure_detectAllPoints(
  aBinaryPlot,
  sensitivity = 0.2,
  point_shape = "circle",
  point_size = 5
)
```

Arguments

`aBinaryPlot`	A binary figure image (an EBImage object). See: `figure_transformToBinary`
`sensitivity`	A value designating the sensitivity of identifying unique points that overlap. A smaller number results in a higher sensitivity to split overlapping points; a larger number will extract only a single point from a cluster of overlapping points.
`point_shape`	The shape of points on figure: can be `"circle"`, `"square"`, or `"diamond"`. If these options do not fit the shape found in a figure, use the option that best approximates that shape.
`point_size`	An integer used to designate the size of the points on the figure. Close alignment to the size of the points on a figure will improve point detection. See `EBImage` to help determine which size to use.

Value

An `EBImage` object with detected scatter-plot points.

See Also

`figure_detectAxis`

`figure_detectAxis` *Detect an axis from a figure image.*

Description

Attempts to detect either the X (horizontal) or Y (vertical) axis from a plotted figure.

Attempts to detect either the X (horizontal) or Y (vertical) axis from a plotted figure.

Usage

```
figure_detectAxis(
  aBinaryPlot,
  axis_type = "X",
  axis_thickness = 5,
  sensitivity = 0.2
)

figure_detectAxis(
  aBinaryPlot,
  axis_type = "X",
  axis_thickness = 5,
  sensitivity = 0.2
)
```

Arguments

`aBinaryPlot`	A binary figure image (an EBImage object). See: `figure_transformToBinary`
`axis_type`	The axis to be detected from a figure: can be `X` or `Y`.
`axis_thickness`	An integer used to designate the thickness of the axis lines on a figure. Close alignment to the thickness of the axis on a figure will improve axis detection.
`sensitivity`	A value designating the sensitivity of identifying straight lines on a figure. A smaller number results in a higher sensitivity to identify axes.

Value

An `EBImage` object with detected points.

An `EBImage` object with detected points.

See Also

figure_detectAllPoints

figure_detectAllPoints

`figure_display` *Displays an image plot.*

Description

Displays a .jpg, .jpeg, .png, or .tiff image file containing a plotted figure, or plots an EBImage object.

Usage

```
figure_display(file = file.choose(), browser = FALSE)
```

Arguments

`file`	The file name and location of a plot figure or EBImage object. Prompts for file name if nothing is explicitly called. Preferably in .jpg format.
`browser`	When `"TRUE"`, displays the figure image in the default web browser.

Value

An `EBImage` object figure.

See Also

figure_read

`figure_displayDetectedPoints` *Displays detected points on figure.*

Description

Generates a raster image of a figure with the detected points painted on a background/reference figure.

Usage

```
figure_displayDetectedPoints(
  aDetectedPlot,
  background = NULL,
  color = "red",
  size = 2,
  ignore = FALSE
)
```

Arguments

`aDetectedPlot`	A binary figure image with detected points (an `EBImage` object). See: `figure_detectAllPoints`
`background`	An EBImage figure of same size to be used as background (e.g., the original (RGB/color) figure image).
`color`	The color to paint the detected points.
`size`	The radius of the painted points.
`ignore`	When `TRUE` does not display painted image, only returns painted image EBImage object.

Value

A RGB `EBImage` painted with detected points.

See Also

figure_displayDetections

`figure_displayDetections`

Displays the detected figure objects.

Description

Generates a raster image of a figure with the detected objects painted on a background/reference figure.

Usage

```
figure_displayDetections(
  aDetectedPlot,
  background = NULL,
  color = "red",
  ignore = FALSE
)
```

Arguments

`aDetectedPlot`	A binary figure image with detected objects (an `EBImage` object).
`background`	An `EBImage` figure of same size to be used as background (e.g., the original [RGB/color] figure image).
`color`	The color to paint the detected objects.
`ignore`	When `TRUE` does not display painted image, only returns painted image EBImage object.

Value

A RGB `EBImage` painted with detected figure objects.

`figure_extractDetectedPoints`

Extracts data points from a detected image.

Description

Extracts raw X and Y data from the points detected in a scatter-plot figure.

Usage

```
figure_extractDetectedPoints(
  aDetectedPlot,
  xAxis = NULL,
  yAxis = NULL,
  X_min = NULL,
  X_max = NULL,
  Y_min = NULL,
  Y_max = NULL,
  summarize = TRUE
)
```

Arguments

`aDetectedPlot`	A binary figure image with detected points (an `EBImage` object). See: `figure_detectAllPoints`
`xAxis`	A binary figure image with detected X-axis (an `EBImage` object). See: `figure_detectAxis`.
`yAxis`	A binary figure image with detected Y-axis (an `EBImage` object). See: `figure_detectAxis`.
`X_min`	The minimum value of X reported on the figure X-axis.
`X_max`	The maximum value of X reported on the figure X-axis.
`Y_min`	The minimum value of Y reported on the figure Y-axis.
`Y_max`	The maximum value of Y reported on the figure Y-axis.
`summarize`	When `TRUE` returns a summary of the regression parameters (intercept + slope * X), R-squared, Pearson's product moment correlation coefficient (r), and its variance (var_r) and sample size (N).

Value

A data frame with the extracted X and Y values.

`figure_read` *Reads/loads a figure image from file.*

Description

Reads a .jpg, .jpeg, .png, or .tiff image file containing a plotted figure.

Usage

```
figure_read(file = file.choose(), display = FALSE)
```

Arguments

`file`	The file name and location of a plot figure. Prompts for file name if none is explicitly called. Preferably in .jpg format.
`display`	When `"TRUE"`, displays the figure as a raster image.

Value

An `EBImage` object figure.

See Also

figure_write

`figure_removeOutlyingPoints` *Remove outlier points from a figure.*

Description

Removes all detected points outside of axis range. Requires three detected images: one based on `figure_detectAllPoints`, and two others based on detected X- and Y-axes (i.e. `figure_detectAxis`)

Usage

```
figure_removeOutlyingPoints(aDetectedPlot, xAxis = NULL, yAxis = NULL)
```

Arguments

`aDetectedPlot`	A binary figure image with detected points (an `EBImage` object). See: `figure_detectAllPoints`
`xAxis`	A binary figure image with detected X-axis (an `EBImage` object). See: `figure_detectAxis`
`yAxis`	A binary figure image with detected Y-axis (an `EBImage` object). See: `figure_detectAxis`

Value

An `EBImage` object with detected points within the specified X- and Y-axis ranges.

`figure_scatterPlot` *Detect and display all scatter plot objects.*

Description

Automated detection of the X-axis, Y-axis, and points on a scatter-plot figure image. The default returns these detected objects as an `EBImage` raster image, as well as the estimated effect size (correlation coefficient or r) of the data within the scatter-plot.

Usage

```
figure_scatterPlot(
  file = file.choose(),
  binary_threshold = 0.6,
  binary_point_fill = FALSE,
  binary_point_tolerance = 2,
  axis_thickness = 5,
  axis_sensitivity = 0.2,
  axis_X_color = "#00ABAB",
  X_min = 40,
  X_max = 140,
  axis_Y_color = "#B0D36A",
  Y_min = 40,
  Y_max = 140,
  point_sensitivity = 0.2,
  point_shape = "circle",
  point_size = 3,
  point_color = "#0098B2",
  ignore = FALSE
)
```

Arguments

`file`	The file name and location of a scatter plot figure. Prompts for file name if none is explicitly called.
`binary_threshold`	A proportion from zero to one designating the gray-scale threshold to convert pixels into black or white. Pixel intensities below the proportion will be converted to black, and those above white.
`binary_point_fill`	If `TRUE` then fills empty points/symbols in figure.
`binary_point_tolerance`	An integer used to designate the size of the points to the fill. Increase value to better fill empty points.
`axis_thickness`	An integer used to designate the thickness of the axis lines on a figure. Close alignment to the thickness of the axis on a figure will improve axis detection.
`axis_sensitivity`	A value designating the sensitivity of identifying straight lines on figure. A smaller number results in a higher sensitivity to identify axes.
`axis_X_color`	The color to paint the detected X-axis.
`X_min`	The minimum X value displayed on the X-axis (used to scale detected data points).
`X_max`	The maximum X value displayed on the X-axis (used to scale detected data points).
`axis_Y_color`	The color to paint the detected Y-axis.

`Y_min`	The minimum Y value displayed on the Y-axis (used to scale detected data points).
`Y_max`	The maximum Y value displayed on the Y-axis (used to scale detected data points).
`point_sensitivity`	A value designating the sensitivity of identifying unique points that overlap. A smaller number results in a higher sensitivity to split overlapping points; a larger number will extract only a single point from a cluster of overlapping points.
`point_shape`	The shape of points on figure: can be `"circle"`, `"square"`, or `"diamond"`. If these options do not fit the shape found in a figure, use the option that best approximates that shape.
`point_size`	An integer used to designate the size of the points on the figure. Close alignment to the size of the points on a figure will improve point detection. See `EBImage` package to help determine which size to use.
`point_color`	The color to paint the detected scatter plot points.
`ignore`	When `TRUE` does not display painted image, only returns painted image EBImage object.

Value

A data frame with detected points.

`figure_splitPlot` *Splits a composite figure that contains multiple plots.*

Description

Automatically detects divisions among multiple plots found within a single figure image file. It then uses these divisions to split the image into multiple image files; each containing only a single X-Y plot. Currently only works on composite figures that have a matrix-style presentation where each sub-plot has the same size.

Usage

```
figure_splitPlot(
  file = file.choose(),
  binary_threshold = 0.6,
  space_sensitivity_X = 0.4,
  space_sensitivity_Y = 0.6,
  border_buffer = 5,
  guess_limit = 10,
  ignoreX = FALSE,
  ignoreY = FALSE,
  quiet = FALSE
)
```

Arguments

`file`	The file name and location of a composite figure. Prompts for file name if none is explicitly called.
`binary_threshold`	A proportion from zero to one designating the gray-scale threshold to convert pixels into black or white. Pixel intensities below the proportion will be converted to black, and those above white.
`space_sensitivity_X`	A proportion ranging from zero to one that designates the size of the separation among sub-plots along the X-axis relative to the largest empty space detected in the figure image. As space_sensitivity_X approaches 1, finer empty spaces (e.g., empty spaces found in between plot captions and the axis line) will be treated as plot divisions.
`space_sensitivity_Y`	A proportion ranging from zero to one that designates the size of the seperation among sub-plots along the Y-axis relative to the largest empty space detected in the figure image. As space_sensitivity_Y approaches 1, finer empty spaces (e.g., empty spaces found in between plot captions and the axis line) will be treated as plot divisions.
`border_buffer`	An integer value designating the amount of empty space around the figure image that should be ignored. As the number increases, more blank space near the image's edge will be ignored.
`guess_limit`	An integer value designating the number of guesses for within a figure image. The default value designates the top 10 guesses of divisions. Increase this number if there are more than 6 subplots per axis.
`ignoreX`	When `TRUE`, ignores detection of sub-plots along the X-axis.
`ignoreY`	When `TRUE`, ignores detection of sub-plots along the Y-axis.
`quiet`	When `TRUE`, does not print to console the saved file names.

Value

The number of sub-plots saved to file.

`figure_transformByColors`

Transforms RBG figure into list of binary images.

Description

Generates a list of binary images relative to the number of colors in an RBG figure. Useful to do when there are multiple objects in a figure presented with different colors.

Usage

```
figure_transformByColors(aFigure, colorsToSplit = 2)
```

Arguments

`aFigure`	The original (RBG/color) figure image (an `EBImage` object).
`colorsToSplit`	An integer designating the number of colors in the figure. The number indicates the number of color intensities to divide into separate binary figures.

Value

A colorsToSplit + 1 list of `EBImage` black and white objects. The final item in this list will be an inverse binary of the original figure.

See Also

`figure_transformToBinary`

`figure_transformToBinary` *Transforms figure to binary image.*

Description

Transforms a figure into a black and white image. This pre-processing of the image is necessary to help identify objects within the figure (e.g., axes, plotted points).

Usage

```
figure_transformToBinary(
  aFigure,
  threshold = 0.6,
  point_fill = FALSE,
  point_tolerance = 2
)
```

Arguments

`aFigure`	The original figure image (an `EBImage` object).
`threshold`	A proportion from zero to one designating the gray-scale threshold to convert pixels into black or white. Pixel intensities below the proportion will be converted to black, and those above white. Helps remove noise and increase contrast among candidate objects to detect.
`point_fill`	If `TRUE` then fills empty points/symbols in figure.
`point_tolerance`	An integer used to designate the size of the points to fill. Increase value to better fill empty points.

Value

An `EBImage` black and white object ready for object detection.

`figure_write` *Saves/writes a figure image.*

Description

Writes a figure image to file and returns the file name.

Usage

```
figure_write(aFigure, file = NULL)
```

Arguments

`aFigure`	The `EBImage` figure.
`file`	Name and location of file to save. Supports .jpg, .png, and .tiff image formats.

Value

Vector of file names.

See Also

`figure_read`

`impute_missingness` *Provides a summary of missingness in a dataset.*

Description

Generates a summary of the percentage of missing data in a dataset. Provides insight on the appropriateness of imputation methods. For example, if 30% of data is missing, then perhaps this is too much to impute.

Usage

```
impute_missingness(aDataFrame)
```

Arguments

`aDataFrame`	A data.frame containing columns that will be assessed for missingness.

Value

A data frame that summarizes percent missingness for each column of a dataset.

Examples

```
data(example_references_metagear)
impute_missingness(example_references_metagear)
```

`impute_SD` *Imputes missing standard deviations in a dataset.*

Description

Imputes (fills gaps) of missing standard deviations (SD) using simple imputation methods following Bracken (1992) and Rubin and Schenker's (1991) "hot deck" approach.

Usage

```
impute_SD(
  aDataFrame,
  columnSDnames,
  columnXnames,
  method = "Bracken1992",
  range = 3,
  M = 1
)
```

Arguments

`aDataFrame`	A data frame containing columns with missing SD's (coded as `NA`) and their complete means (used only for nearest-neighbor method).
`columnSDnames`	Label of the column(s) with missing SD. Can be a string or list of strings.
`columnXnames`	Label of the column(s) with means (X) for each SD. Can be a string or list of strings. Must be complete with no missing data.
`method`	The method used to impute the missing SD's. The default is `"Bracken1992"` which applies Bracken's (1992) approach to impute SD using the coefficient of variation from all complete cases. Other options include: `"HotDeck"` which applies Rubin and Schenker's (1991) resampling approach to fill gaps of missing SD from the SD's with complete information, and `"HotDeck_NN"` which resamples from complete cases with means that are similar to missing SD's.
`range`	A positive number on the range of neighbours to sample from for imputing SD's. Used in combination with `"HotDeck_NN"`. The default is 3; which indicates that the 3 means that are most similar in rank order to the mean with the missing SD will be resampled.
`M`	The number of imputed datasets to return. Currently only works for `"HotDeck"` method.

Value

An imputed (complete) dataset.

References

Bracken, M.B. 1992. Statistical methods for analysis of effects of treatment in overviews of randomized trials. Effective care of the newborn infant (eds J.C. Sinclair and M.B. Bracken), pp. 13-20. Oxford University Press, Oxford.

Rubin, D.B. and Schenker, N. 1991. Multiple imputation in health-care databases: an overview and some applications. Statistics in Medicine 10: 585-598.

`isPDF` *Evaluates whether a file is a PDF document.*

Description

Checks if provided file is in Portable Document Format (PDF).

Usage

```
isPDF(aFileName, verbose = TRUE)
```

Arguments

`aFileName`	A string that identifies a file name (and directory path) of the PDF candidate.
`verbose`	Provides more elaborate description of why the file could not be evaluated as a PDF (e.g., when validating a PDF online). When `"quiet"`, an error message is not generated.

Value

A logical value indicating whether the file is a PDF document.

`Kam_et_al_2003_Fig2.jpg` *An example image of a scatterplot figure*

Description

A jpg image of a scatterplot from Figure 2 of Kam, M., Cohen-Gross, S., Khokhlova, I.S., Degen, A.A. and Geffen, E. 2003. Average daily metabolic rate, reproduction and energy allocation during lactation in the Sundevall Jird Meriones crassus. Functional Ecology 17:496-503.

Format

A raw jpg-formated image

Note

How to use

`readImage(system.file("images", "Kam_et_al_2003_Fig2.jpg", package = "metagear"))`

`Kortum_and_Acymyan_2013_Fig4.jpg`
An example image of a bar plot figure

Description

A jpg image of a bar plot from Figure 4 of Kortum, P., and Acymyan, C.Z. 2013. How low can you go? Is the System Usability Scale range restricted? Journal of Usability Studies 9:14-24.

Format

A raw jpg-formated image

Note

How to use

`readImage(system.file("images", "Kortum_and_Acymyan_2013_Fig4.jpg", package = "metagear"))`

`MA_effectsTable` *Generate an ANOVA-like effects table for a meta-analysis.*

Description

Generates an ANOVA-like effects table that summarizes the within and between-study homogeneity tests (Q-tests), as well as moderator level Q-tests as originally described by Hedges and Olkin (1985; p. 156).

Usage

```
MA_effectsTable(model, weights, data, effects_model = "random")
```

Arguments

`model`	A two-sided linear formula object describing the model, with the response (effect sizes) on the left of a ~ operator and the moderator variables, separated by +, :, * operators, on the right.
`weights`	A column label from data.frame of variances to be used as weights.
`data`	An optional data frame containing the variables named in the model.
`effects_model`	The default is `"random"`, which specifies a random-effects meta-analysis (DerSimonian and Laird method). Other options include `"fixed"` which presents fixed-effect analyses.

Value

An lm object of main effects.

References

DerSimonian, R., and N. Laird. 1986. Meta-analysis in clinical trials. Controlled Clinical Trials, 7, 177-188.

Hedges, L.V., and I. Olkin. 1985. Statistical methods for meta-analysis. Academic Press, New York, USA.

`PDFs_collect` *Attempts to download PDFs from multiple DOI links.*

Description

Tries to download a collection of PDF files using multiple digital object identifier (DOI) links. Updates a data frame with the success of these downloads. The function is a wrapper for `PDF_download`. NOTE: A single DOI may generate multiple PDF files. **If running downloader in Windows, having** `"WindowsProxy = TRUE"` **will significantly improve download success.**

Usage

```
PDFs_collect(
  aDataFrame,
  DOIcolumn,
  FileNamecolumn,
  directory = getwd(),
  randomize = FALSE,
  seed = NULL,
  buffer = FALSE,
  validatePDF = TRUE,
  quiet = FALSE,
  showSummary = TRUE,
  WindowsProxy = FALSE
)
```

Arguments

`aDataFrame`	A data frame containing a column of DOIs and a column of individual file names for each downloaded PDF.
`DOIcolumn`	The label of the column containing all the DOI links.
`FileNamecolumn`	The label of the column containing all the strings that will be used to rename the downloaded files.
`directory`	A string of the location (directory) were downloaded PDF files are to be saved. NOTE: helps to have this directory created before initializing the `PDFs_collect` function.
`randomize`	When `TRUE` will attempt to download PDFs in a random order. This may be necessary to ensure that host websites do not have their HTML and files repeatedly accessed.
`seed`	An integer used to enforce repeatability when randomly downloading PDFs.
`buffer`	When `TRUE` will randomly delay the downloads by a few seconds (with a mean 4 seconds and a range of 1 to 20 seconds). Another strategy to avoid quickly and repeatedly accessing host websites.
`validatePDF`	When `TRUE` will only save to files that are valid PDF documents. When `FALSE` will save all candidate files, even if they are not valid PDF formats.
`quiet`	When `FALSE` does not print to console individual download progress and summary.
`showSummary`	When `FALSE` does not print overall summary of download successes and failures.
`WindowsProxy`	When `TRUE` significantly improves download success for computers running Windows; when `FALSE` on a Windows based computer, you may only be able to download 30 to 50 PDFs at a time before a connection error occurs and halts all downloads (e.g., `InternetOpenUrl failed` error).

Value

The data frame with new column containing download-outcome successes.

See Also

`PDF_download`

Examples

```
## Not run:

data(example_references_metagear)
someRefs <- effort_initialize(example_references_metagear)
dir.create("metagear_downloads")
PDFs_collect(aDataFrame = someRefs, DOIcolumn = "DOI",
             FileNamecolumn = "STUDY_ID", directory = "metagear_downloads",
   WindowsProxy = TRUE)

## End(Not run)
```

`PDF_download` *Attempts to download a PDF using a DOI link.*

Description

Tries to download a PDF file using the digital objected identifier (DOI) link. Uses ad hoc searches of journal HTML pages to detect candidate PDFs for download. Downloads all candidate pdfs. **If running downloader in Windows, having `"WindowsProxy = TRUE"` will significantly improve download success.**

Usage

```
PDF_download(
  DOI,
  directory = getwd(),
  theFileName = "temp",
  validatePDF = TRUE,
  quiet = FALSE,
  WindowsProxy = FALSE
)
```

Arguments

`DOI`	A string of the DOI (digital object identifier) used to identify the source of a journal article PDF file(s).
`directory`	A string of the location (directory) were downloaded PDF files are to be saved. Directory name must end with "\\".
`theFileName`	Used to rename the downloaded file. No need to include extension ".pdf".
`validatePDF`	When `"TRUE"` will only save to files that are valid PDF documents. When `"FALSE"` will save all candidate files, even if they are not valid PDF formats.
`quiet`	When `"FALSE"` does not print to console download progress and summary.
`WindowsProxy`	When `TRUE` significantly improves download success for computers running Windows; when `FALSE` on a Windows based computer, you may only be able to download 30 to 50 PDFs at a time before a connection error occurs and halts all downloads (e.g., `InternetOpenUrl failed` error).

Value

A string describing the download success. If unsuccessful, returns the type of error during the download attempt.

See Also

`PDFs_collect`

`PDF_extractImages` *Attempts to extract all images from a PDF*

Description

Tries to extract images within a PDF file. Currently does not support decoding of images in CCITT compression formats. However, will still save these images to file; as a record of the number of images detected in the PDF.

Usage

```
PDF_extractImages(file = file.choose())
```

Arguments

`file`	The file name and location of a PDF file. Prompts for file name if none is explicitly called.

Value

A vector of file names saved as images.

`plot_PRISMA` *Plots and creates a PRISMA flow diagram.*

Description

Creates a PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) flow diagram depicting the 'flow' of study inclusions and exclusions during various systematic review phases. It is meant to describe the number of studies identified, included, excluded, reasons for inclusion/exclusions, and final number of studies used in the meta-analysis. NOTE: currently only supports two start phases, and final phase must not have an exclude branch.

Usage

```
plot_PRISMA(
  aPhaseVector,
  colWidth = 30,
  excludeDistance = 0.8,
  design = "classic",
  hide = FALSE
)
```

Arguments

`aPhaseVector`	A vector of ordered labels (strings) for each phase of the PRISMA diagram. Labels designating the beginning of the diagram are commented with "START_PHASE: " and those designating exclusion phases "EXCLUDE_PHASE: ". These comments will be removed from the diagram.
`colWidth`	An optional value (integer) designating the width of the text box of each phase.
`excludeDistance`	An optional value designating the the distance of exclude phase box from the main flow diagram. Larger values (> 0.8) increase this distance.
`design`	Designates the colorscheme and design of the the flow diagram. The default is `classic` (as in versions of metagear prior to v. 0.4). Others schemes are also available with color and more flat designs, and these can be further customized; see NOTE below for these details.
`hide`	When FALSE, the PRISMA flow diagram is not plotted.

Value

a grid object (grob) list

Note

Using canned or custom PRISMA design layouts

There are several color schemes and design layouts (e.g. curved or flat) available. These designs include: `cinnamonMint`, `sunSplash`, `pomegranate`, `vintage`, `grey`, and `greyMono`. Custom schemes can also be developed by modifying each aspect of the design. These are:

S color of start phases (default: white)

P color of the main phases (default: white)

E color of the exclusion phases (default: white)

F color of the final phase (default: white)

fontSize the size of the font (default: 12)

fontColor the font color (default: black)

fontFace either plain, bold, italic, or bold.italic (default: plain)

flatArrow arrows curved when FALSE (default); arrows square when TRUE

flatBox boxes curved when FALSE (default); Boxes square when TRUE

For example, changing the defaults to have red rather than white exclusion phases, and square boxes, would be: `design = c(E = "red", flatBox = TRUE)`.

References

Moher, D., Liberati, A., Tetzlaff, J. and Altman, D.G., PRISMA Group. (2009) Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. BMJ 339, b2535.

Examples

```
phases <- c("START_PHASE: # of studies identified through database searching",
            "START_PHASE: # of additional studies identified through other sources",
            "# of studies after duplicates removed",
            "# of studies with title and abstract screened",
            "EXCLUDE_PHASE: # of studies excluded",
            "# of full-text articles assessed for eligibility",
            "EXCLUDE_PHASE: # of full-text excluded, not fitting eligibility criteria",
            "# of studies included in qualitative synthesis",
            "EXCLUDE_PHASE: # studies excluded, incomplete data reported",
            "final # of studies included in quantitative synthesis (meta-analysis)")
plot_PRISMA(phases, design = "cinnamonMint")
```

`random_d` *Random generation of Hedges' d effect sizes.*

Description

Generates random Hedges' d (1981, 1982) effect sizes and their variances.

Usage

```
random_d(K, X_t, var_t, N_t, X_c, var_c, N_c, bias_correction = TRUE)
```

Arguments

`K`	Number of effect sizes to generate.
`X_t`	The population mean (mu) of the (t)reatment group.
`var_t`	The population variance of the treatment group mean.
`N_t`	The number of samples of the treatment mean. When a non-negative integer, all treatment means will be estimated using the same N. A vector of unequal N's can also be taken; if so, K will be ignored and the number of randomly generated means will equal the length of that vector, and each mean will be based on each N within the vector.
`X_c`	The population mean (mu) of the (c)ontrol group.
`var_c`	The population variance of the control group mean.
`N_c`	The number of samples of the control mean. When a non-negative integer, all control means will be estimated using the same N. A vector of unequal N's can also be taken; if so, K will be ignored and the number of randomly generated means will equal the length of that vector, and each mean will be based on each N within the vector.
`bias_correction`	When `"FALSE"`, returns Cohen's g effect sizes that are not adjusted using a small-sample correction (J).

Value

A data table with columns of random effect sizes (d) and their variances (var_d).

References

Hedges, L.V. 1981. Distribution theory for Glass's estimator of effect size and related estimators. Journal of Educational Statistics 6: 107-128.

Hedges, L.V. 1982. Estimation of effect size from a series of independent experiments. Psychological Bulletin 92: 490-499.

Examples

```
random_d(K = 5, X_t = 25, var_t = 1, N_t = 15, X_c = 10, var_c = 1, N_c = 15)
```

`random_missingness` *Random generation of missingness in a data frame.*

Description

Generates random `NA`'s in in a column or groups of columns of a data frame. Used in imputation simulations based on complete datasets.

Usage

```
random_missingness(aDataFrame, columnNames, percentMissing = 10)
```

Arguments

`aDataFrame`	A data.frame where missingness will be simulated.
`columnNames`	A string or a vector of strings that describe the column names (labels) where missingness will be simulated.
`percentMissing`	The percentage of missingness within specified columns. "Percent missing" uses a binomial distribution to simulate missing data. Default is 10 (i.e. 10% missing). Use `impute_missingness` for a summary of these randomly generated missing data.

Value

A data table with columns of missing data (specified as `NA`'s).

`random_N` *Random generation of sample sizes (N) for study outcomes.*

Description

Generates random sample sizes (N) by either sampling from a Negative Binomial or Poisson distribution.

Usage

```
random_N(K, method = "NegativeBinomial", mean = 15, min = 3, NB_size = 15)
```

Arguments

`K`	Number of sample sizes to generate.
`method`	A string that defines what sampling distribution to generate random N. The default is `"NegativeBinomial"` but a `"Poisson"` distribution can also be used.
`mean`	The population mean (mu) if `"NegativeBinomial"`, or the lambda (dispersion parameter) if `"Poisson"`. The default is 15, which will generate sample sizes that on average will center around N = 15.
`min`	A non-negative integer that specifies the minimum sample size that can be generated. Default is N = 3.
`NB_size`	Dispersion parameter for the `"Negative Binomial"` distribution that must be strictly positive, but need not be integer. Default is 15, which creates a long tail for random N's ranging to about N = 60. Increase value to create a longer tail of random sample sizes.

Value

A vector of random sample sizes (N).

`random_OR` *Random generation of odds ratio (OR) effect sizes.*

Description

Generates random odds ratios, logged odds ratios, and their variances (Cornfield 1951).

Usage

```
random_OR(K, p_A, N_A, p_B, N_B, continuity = 0.5, logged = TRUE)
```

Arguments

`K`	Number of effect sizes to generate.
`p_A`	The odds of the event of interest for Group A. A probability ranging from zero to one.
`N_A`	The total number of samples of Group A.
`p_B`	The odds of the event of interest for Group B. A probability ranging from zero to one.
`N_B`	The total number of samples of Group B.
`continuity`	Odds ratios with zero events cannot be computed. Following, Cox (1970), a continuity correction can be added to each cell of the 2 by 2 table to help improve this problem of zero events within the table. The default value added is 0.5.
`logged`	When `"FALSE"`, returns non-logged transformed odds ratios and appropriate variances. Default is TRUE.

Value

A data table with columns of random effect sizes (OR) and their variances.

References

Cornfield, J. 1951. A method for estimating comparative rates from Clinical Data. Applications to cancer of the lung, breast, and cervix. Journal of the National Cancer Institute 11: 1269-1275.

Cox, D.R. 1970. The continuity correction. Biometrika 57: 217-219.

Examples

```
random_OR(K = 5, p_A = 0.3, N_A = 100, p_B = 0.1, N_B = 60)
```

`random_pairedN` *Random generation of paired sample sizes (N) for study outcomes.*

Description

Generates random paired sample sizes (N). For example, sample sizes for a treatment group and samples sizes for a control group. These paired N are often correlated within studies.

Usage

```
random_pairedN(K, mean = 15, min = 3, correlation = 0.95)
```

Arguments

`K`	Number of paired sample sizes to generate.
`mean`	The lambda (dispersion parameter) of a Poisson distribution. The default is 15, which will generate sample sizes that on average will center around N = 15.
`min`	A non-negative integer that specifies the minimum sample size that can be generated. Default is N = 3.
`correlation`	A correlation ranging from zero to one that specifies how 'similar' the paired sample sizes will be to one another. Default is 0.95 (i.e. the paired sample sizes will be highly correlated).

Value

A data table of paired random sample sizes (N).

`random_r` *Random generation of correlation coefficients.*

Description

Generates random correlation coefficients (r or Pearson product-moment correlation coefficients) and their variances (Pearson 1895). Also provides Fisher z-transformed correlation coefficients (Fisher 1915).

Usage

```
random_r(K = 100, correlation = 0.5, N = 10, Fisher_Z = FALSE)
```

Arguments

`K`	Number of effect sizes to generate.
`correlation`	The mean population correlation coefficient (rho) to simulate. Must range between -1 to 1.
`N`	The number of samples used to estimate each correlation coefficient. When a non-negative integer, all r will be estimated using the same N. A vector of unequal N's can also be taken; if so, K will be ignored and the number of randomly generated r will equal the length of that vector.
`Fisher_Z`	When `TRUE`, also returns the Fisher z-transformed correlation coefficients and their variances (Fisher 1915).

Value

A data table with columns of random effect sizes (r), their variances and sample sizes.

References

Pearson, K. 1895. Notes on regression and inheritance in the case of two parents. Proceedings of the Royal Society of London 58: 240-242.

Fisher, R.A. 1915. Frequency distribution of the values of the correlation coefficient in samples of an indefinitely large population. Biometrika 10: 507-521.

Examples

```
random_r(K = 5, correlation = 0.5, N = 50)
```

`random_RR` *Random generation of log response ratio (RR) effect sizes.*

Description

Generates random log response ratios and their variances (Hedges et al. 1999). NOTE: samples from a log-normal distribution to generate non-negative control and treatment means (following Lajeunesse 2015).

Usage

```
random_RR(K, X_t, var_t, N_t, X_c, var_c, N_c)
```

Arguments

`K`	Number of effect sizes to generate.
`X_t`	The population mean (mu) of the (t)reatment group (numerator of ratio).
`var_t`	The population variance of the treatment group mean.
`N_t`	The number of samples of the treatment mean. When a non-negative integer, all treatment means will be estimated using the same N. A vector of unequal N's can also be taken; if so, K will be ignored and the number of randomly generated means will equal the length of that vector, and each mean will be based on each N within the vector.
`X_c`	The population mean (mu) of the (c)ontrol group (denominator of ratio).
`var_c`	The population variance of the control group mean.
`N_c`	The number of samples of the control mean. When a non-negative integer, all control means will be estimated using the same N. A vector of unequal N's can also be taken; if so, K will be ignored and the number of randomly generated means will equal the length of that vector, and each mean will be based on each N within the vector.

Value

A data table with columns of random effect sizes (RR) and their variances.

References

Hedges, L.V., J. Gurevitch, and P.S. Curtis. 1999. The meta-analysis of response ratios in experimental ecology. Ecology 80: 1150-1156.

Lajeunesse, M.J. 2015. Bias and correction for the log response ratio used in ecological meta-analysis. Ecology.

Examples

```
random_RR(K = 5, X_t = 25, var_t = 1, N_t = 15, X_c = 10, var_c = 1, N_c = 15)
```

`replicate_MetaWin2.0` *Replicate meta-analysis results and summaries from MetaWin 2.0.*

Description

Replicate meta-analysis results and summaries from Rosenberg's et al. (2000) software 'MetaWin' 2.0. Currently only replicates moderator analyses and not meta-regressions.

Usage

```
replicate_MetaWin2.0(
  model,
  weights,
  effects_model = "random",
  data,
  bootstraps = 999
)
```

Arguments

`model`	A two-sided linear formula object describing the model, with the response (effect sizes) on the left of a ~ operator and the moderator variables, separated by `+`, :, * operators, on the right. NOTE: MetaWin was limited to analyses with a single moderator variable. This function currently supports only categorical moderators.
`weights`	A vector of effect size variances that will be used as weights for the meta-analysis.
`effects_model`	The default is `"random"`, which specifies a random-effects meta-analysis. Other options include `"fixed"` which presents fixed-effect analyses.
`data`	An optional data frame containing the variables named in model and weights.
`bootstraps`	The number of bootstraps used to estimate confidence intervals. As with 'MetaWin' 2.0, the default is 999.

References

Rosenberg, M.S., Adams, D.C., and Gurevitch, J. 2000. MetaWin: Statistical Software for Meta-Analysis. Sinauer Associates Sunderland, Massachusetts.

`replicate_phyloMeta1.3` *Replicate phylogeneic meta-analysis results and summaries from phyloMeta 1.3.*

Description

Replicate phylogenetic meta-analysis results and summaries from Lajeunesse (2011) software 'phyloMeta' 1.3. Currently does not fully replicate all functionality.

Usage

```
replicate_phyloMeta1.3(model, weights, data, phylogenyFile)
```

Arguments

`model`	A two-sided linear formula object describing the model, with the response (effect sizes) on the left of a ~ operator and the moderator variables, separated by +, :, * operators, on the right. NOTE: phyloMeta was limited to analyses with a single moderator variable. This function currently supports only numerical categorical moderators.
`weights`	A vector of effect size variances that will be used as weights for the meta-analysis.
`data`	A data frame containing the variables named in model and weights and species names (names must be exact as specified in phylogeny).
`phylogenyFile`	A text file containing a NEWICK phylogeny. The number of species must be same as (k) number of effect sizes in data.

References

Lajeunesse, M.J. (2011) phyloMeta: a program for phylogenetic comparative analyses with meta-analysis. Bioinformatics 27, 2603-2604.

`scrape_bibliography` *Attempts to scrape/extract bibliographic data from Web of Science.*

Description

A not so elegant way to extract bibliographic data of a research article by scraping the contents of Web of Science (WOS). Requires the DOI (digital object identifier) of an article, as well as web access with an institutional subscription to WOS. Note: This function is not suited to extract data for book chapters available on WOS. Current extractions include: a vector of authors (author), publication year (year), article title (title), journal title (journal), journal volume (volume), page numbers (pages), abstract (abstract), number of references (N_references), number of citations (N_citations), journal impact factor (journal_IF), and the year the journal impact factor was released (journal_IF_year). Finally the date of the scrape is also provided (date_scraped). Bulleted abstracts or those with subheadings or subparagraphs will not be extracted properly.

Usage

```
scrape_bibliography(DOI, quiet = FALSE)
```

Arguments

`DOI`	A string as the DOI (digital object identifier) of a research article.
`quiet`	When `TRUE`, does not print an MLA-style reference of the extracted article.

Value

A list of bibliographic extractions and a timestamp of the scrape.

Examples

```
## Not run:

# use DOI to scrape number of WOS citations of a research article
data(example_references_metagear)
someRefs <- effort_initialize(example_references_metagear)
theWOSRef <- scrape_bibliography(someRefs$DOI[1])
print(paste("citations = ", theWOSRef$N_citations))


## End(Not run)
```

Index