Package 'eiExpand'

July 22, 2025

Type Package

Title Utilities for Expanding Functionality of 'eiCompare'

Version 1.0.5

Description Augments the 'eiCompare' package's Racially Polarized Voting (RPV) functionality to streamline analyses and visualizations used to support voting rights and redistricting litigation. The package implements methods described in Barreto, M., Collingwood, L., Garcia-Rios, S., & Oskooii, K. A. (2022). ``Estimating Candidate Support in Voting Rights Act Cases: Comparing Iterative EI and EI-R×C Methods" <doi:10.1177/0049124119852394>.

License GPL-3

Depends R (>= 3.5.0)

Imports dplyr, tidyr, tidyselect (>= 1.2.0), stringr, ggplot2, ggmap, viridis, sf, magrittr, rlang

Suggests knitr, markdown, rmarkdown, roxygen2, testthat

RoxygenNote 7.2.1

Encoding UTF-8

Maintainer Rachel Carroll <rachelcarroll4@gmail.com>

LazyData true

LazyDataCompression xz

NeedsCompilation no

Author Rachel Carroll [aut, cre], Loren Collingwood [aut] (ORCID: <https://orcid.org/0000-0002-4447-8204>)

Repository CRAN

Date/Publication 2023-03-14 18:50:08 UTC

Contents

`example_performance_results` *Example performance analysis results*

Description

Example performance analysis results

Usage

```
example_performance_results
```

Format

An object of class `data.frame` with 12 rows and 7 columns.

`example_rpvDF` *Example RPV analysis results in Washington State*

Description

Example RPV analysis results in Washington State

Usage

```
example_rpvDF
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 72 rows and 13 columns.

mt_block_data *Example block-level population data from Montana for split precinct analysis*

Description

Example block-level population data from Montana for split precinct analysis

Usage

```
mt_block_data
```

Format

An object of class `sf` (inherits from `data.frame`) with 3880 rows and 2 columns.

percent_intersect *Calculate percent land area intersections*

Description

Calculates the percent area intersection between the geometries in a sf data.frame and a single boundary shape.

Usage

```
percent_intersect(sfdf, shp)
```

Arguments

`sfdf`	sf dataframe with one or more geometries.
`shp`	sf dataframe with a single shape boundary.

Value

sfdf dataframe with pct_intersect column

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

`performance` *Performance Analysis Calculation*

Description

Performance Analysis calculates election outcomes of past contests given hypothetical voting district(s). This analysis has been used to determine if a Gingles III violation occurs due to how a district map is drawn. It can also be used to demonstrate that a more equitable alternative map exists. This function assumes RPV so it should only be used with contests where RPV has been established.

Usage

```
performance(
  data = NULL,
  cands = "",
  candidate = "",
  preferred_candidate = "",
  total = "",
  contest = "",
  year = "",
  election_type = "",
  map = "",
  jurisdiction = "",
  includeTotal = FALSE
)
```

Arguments

`data`	A data.frame object containing precinct-level election results for contests of interest. It must include candidate vote counts and contest total votes fields and must be subsetted to the relevant precincts. This data.frame will likely be the output of a "Split Precinct Analysis".
`cands`	A character vector of the candidate vote counts field names from `data` that are relevant to the given year and contest being analyzed.
`candidate`	A character vector of candidate names. The names must be listed in the same order `cands`. The values will appear in the output data.frame exactly as they are written in this argument.
`preferred_candidate`	A character vector of preferred racial groups associated with the candidates. The values must be listed in the correct order with respect to the `cands`/`candidate` arguments. The values will appear in the output data.frame exactly as they are written in this argument.
`total`	A character vector of the the contest total vote count field names from `data`.
`contest`	The name of the contest being analyzed

year	The year of the contest being analyzed
election_type	The election type the contest being analyzed (e.g "General" or "Primary")
map	String containing the name of the district map being analyzed (e.g "remedial" or "adopted"). This is an optional field that defaults to blank.
jurisdiction	String containing the name of the jurisdiction being analyzed (i.e a district number or "County"). Be sure that data is subsetted only to this jurisdiction.
includeTotal	Boolean indicating if a total number of votes row should be appended to the output data.frame

Value

data.frame of Performance Analysis results by candidate

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

Loren Collingwood <lcollingwood@unm.edu>

Examples

```
library(eiExpand)
data(south_carolina)

# Get sample election data
D5_election <- south_carolina %>%
   dplyr::filter(District == 5)

# Run performance Analysis on 2018 Governor contest
perf_results <- performance(
  data = D5_election,
  cands = c("R_mcmaster", "D_smith"),
  candidate = c("McMaster (R)", "Smith (D)"), # formatted candidate names
  preferred_candidate = c("White", "Black"), # race preference of candidates respectively
  total = "total_gov",
  contest = "Governor",
  year = 2018,
  election_type = "General",
  jurisdiction = "District 5"
)
```

---

performance_plot *Performance Analysis Plotting Function*

Description

Uses output from performance() to create a ggplot performance analysis visualization.

Usage

```
performance_plot(
  perfDF,
  title = "Performance Analysis Results",
  subtitle = NULL,
  legend_name = "Preferred Candidate:",
  preferred_cand_races = NULL,
  colors = NULL,
  breaks = seq(0, 100, 20),
  lims = c(0, 100),
  bar_size = 5,
  label_size = 4,
  position_dodge_width = 0.8,
  cand_name_size = 6,
  cand_name_pad = -1,
  contest_name_size = 20,
  contest_name_pad = NULL,
  panel_spacing = 0.7,
  panelBy = "Jurisdiction",
  includeCandName = TRUE,
  includeMeanDiff = TRUE
)
```

Arguments

perfDF	A data.frame object containing performance analysis results from performance()
title	The plot title
subtitle	The plot subtitle
legend_name	The legend title
preferred_cand_races	A character vector of the unique races contained in the preferred_candidate column of perfDF. This argument is optional and is used with colors to indicate the color of the plot associated with the race preferences.
colors	Plot colors for the voter race groups. Colors must be listed in the desired order with respect preferred_cand_races if arguments are used together.
breaks	Numeric vector containing x axis breaks
lims	Numeric vector containing x axis limits
bar_size	The size of plot bars. Passed to geom_linerange().
label_size	The size of vote share labels
position_dodge_width	The width value passed to position_dodge(). Affects spacing between the plot bars.
cand_name_size	Text size of candidate names if includeCandName = TRUE
cand_name_pad	Padding between candidate name and y axis if includeCandName = TRUE.

`contest_name_size`
: Text size of contest name

`contest_name_pad`
: Padding between contest name and y axis

`panel_spacing`
: space between panels. This argument is relevant only if there are multiple jurisdictions in `perfDF`.

`panelBy`
: Column name from perfDF passed to `facet_grid()` to create panels. Recommended options are Jurisdiction and Map. Defaults to Jurisdiction.

`includeCandName`
: Logical indicating if candidate names should appear on the left side of the plot.

`includeMeanDiff`
: Logical indicating if the mean difference between preferred_candidate across all elections should appear in the plot.

Value

ggplot visualization of performance analysis

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

See Also

performance

Examples

```
library(eiExpand)
data(example_performance_results)
performance_plot(example_performance_results)

#ggplot2::ggsave("perf_plot.png", width = 12, height = 7)
```

---

planShp *Example district plan shape for split precinct analysis*

Description

Example district plan shape for split precinct analysis

Usage

```
planShp
```

Format

An object of class `sf` (inherits from `data.frame`) with 1 rows and 18 columns.

`rpv_coef_plot` *Racially Polarized Voting Analysis (RPV) Coefficient Plot*

Description

Creates a coefficient plot showing of RPV results estimate ranges of all contests by voter race

Usage

```
rpv_coef_plot(
  rpvDF = NULL,
  title = "Racially Polarized Voting Analysis Estimates",
  caption = "Data: eiCompare RPV estimates",
  ylab = NULL,
  colors = NULL,
  race_order = NULL
)
```

Arguments

`rpvDF`	A data.frame containing RPV results
`title`	The plot title
`caption`	The plot caption
`ylab`	Label along y axis
`colors`	Character vector of colors, one for each racial group. The order of colors will be respective to the order of racial groups.
`race_order`	Character vector of racial groups from the `voter_race` column of `rpvDF` in the order they should appear in the plot. If not specified, the race groups will appear in alphabetical order.

Value

Coefficient plot of RPV analysis as a ggplot2 object

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

Stephen El-Khatib <stevekhatib@gmail.com>

Loren Collingwood <lcollingwood@unm.edu>

Examples

```
library(eiExpand)
data(example_rpvDF)

dem_rpv_results <- example_rpvDF %>% dplyr::filter(Party == "Democratic")
rpv_coef_plot(dem_rpv_results)
```

rpv_normalize *Normalize RPV results*

Description

Create a dataframe of normalized RPV results when using the cvap, vap, or bisg denominator method, i.e., take RPV results only among people estimated to have voted.

Usage

```
rpv_normalize(ei_object, cand_cols, race_cols)
```

Arguments

ei_object	Output from ei_iter() or ei_rxc()
cand_cols	A character vector of the candidate column names to be normalized from ei_object. Only use candidate column name columns, not the No Vote column.
race_cols	A character vector of the racial group column names to be normalized from ei_object

Value

Normalized RPV results in a data.frame

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

Loren Collingwood <lcollingwood@unm.edu>

Examples

```
#library(eiExpand)
#data("south_carolina")
#prec_election_demog <- south_carolina[1:50,]

## run rpv using eiCompare (rxc method)
#rxcVote <-  eiCompare::ei_rxc(
#  data = prec_election_demog,
#  cand_cols = c('pct_mcmaster', 'pct_smith', 'pct_other_gov', 'pct_NoVote_gov'),
#  race_cols = c('pct_white', 'pct_black', 'pct_race_other'),
#  totals_col = "total_vap")

## normalize results accounting for no vote using rpv_normalize()
## only include the candidate and race cols of interest for the rpv analysis
#rpv_results <- rpv_normalize(
 # ei_object = rxcVote,
 # cand_cols = c('pct_mcmaster', 'pct_smith', 'pct_other_gov'),
 # race_cols = c('pct_white', 'pct_black')
```

```
#)
```

---

rpv_plot *Racially Polarized Voting Analysis (RPV) Plot*

Description

Creates a custom visualization of RPV results

Usage

```
rpv_plot(
  rpvDF = NULL,
  title = "Racially Polarized Voting Analysis Results",
  subtitle = "Estimated Vote for Candidates by Race",
  legend_name = "Voters' Race:",
  voter_races = NULL,
  colors = NULL,
  position_dodge_width = 0.8,
  bar_size = NULL,
  label_size = 4,
  contest_name_size = 20,
  cand_name_size = 6,
  contest_name_pad = NULL,
  cand_name_pad = -1.5,
  contest_sep = NULL,
  shade_col = "grey75",
  shade_alpha = 0.1,
  panel_spacing = NULL,
  breaks = seq(0, 100, 20),
  lims = c(0, 110),
  includeErrorBand = FALSE,
  includeCandName = TRUE,
  panelBy = NULL,
  txtInBar = NULL
)
```

Arguments

rpvDF	A data.frame containing RPV results
title	The plot title
subtitle	The plot subtitle
legend_name	The legend title

voter_races	A vector of the unique voter races contained in the Voter_Race column of rpvDF. This argument will set the order in which voter races are displayed in the plot and legend. Can be used with colors, to indicate the which color of the plot to associate with each voter race.
colors	Defines the plot colors for the voter race groups. Colors must be listed in the desired order with respect voter_races if arguments are used together.
position_dodge_width	The width value indicating spacing between the plot bars. Passed to position_dodge().
bar_size	The size of plot bars. Passed to geom_linerange().
label_size	The size of RPV estimate label
contest_name_size	Text size of contest name
cand_name_size	Text size of candidate names if includeCandName = TRUE
contest_name_pad	Padding between contest name and y axis
cand_name_pad	Padding between candidate name and y axis if includeCandName = TRUE.
contest_sep	String indicating how to separate contest. Options "s", "shade", or "shading" shade the background of every other contest. Options "l", "line", "lines" create light grey double lines between contests.
shade_col	color to shade contest separation bands when contest_sep = "s". Defaults to light grey.
shade_alpha	alpha parameter passed to geom_tile() to indicate transparency of contest separation bands when contest_sep = "s"
panel_spacing	Space between facet grid panels
breaks	Numeric vector containing x axis breaks
lims	Numeric vector containing x axis limits
includeErrorBand	Logical indicating if the confidence interval band should appear on the plot. If TRUE, the RPV estimate labels will appear in the middle of each bar instead of at the ends so they don't cover the error bands.
includeCandName	Logical indicating if candidate names should appear on the left side of the plot.
panelBy	Column name from rpvDF passed to facet_grid() to create panels.
txtInBar	Logical indicating location of the RPV estimate labels. If, TRUE, estimates will be in the middle of the plot bars. If FALSE, they will be at the end of the bars.

Value

Bar plot visualization of RPV analysis as a ggplot2 object

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

Loren Collingwood <lcollingwood@unm.edu>

Kassra Oskooii <kassrao@gmail.com>

Examples

```
library(eiExpand)
data(example_rpvDF)

# Note that these plots are designed to be
# saved as a png using ggplot2::ggsave(). See first example for recommending
# sizing, noting that height and weight arguments may need adjusting
# depending on plot attributes such as number of contests and paneling

# plot county-level results with all defaults
rpvDF_county <- example_rpvDF %>% dplyr::filter(Jurisdiction == "County")
rpv_plot(rpvDF_county)

# save to png with recommended sizing
# ggplot2::ggsave("rpv_plot_default.png", height = 10, width = 15)

# include CI bands
rpv_plot(rpvDF_county, includeErrorBand = TRUE)

# include CI bands with estimate labels outside bar
rpv_plot(
  rpvDF_county,
  includeErrorBand = TRUE,
  txtInBar = FALSE
)

# panel by preferred candidate
rpvDF_county$Year <- paste(rpvDF_county$Year,
                           "\n") # so contest and year are on different lines
rpvDF_county$Preferred_Candidate <- paste(rpvDF_county$Preferred_Candidate,
                                          "\nPreferred Candidate")
rpv_plot(
  rpvDF_county,
  panel_spacing = 6,
  panelBy = "Preferred_Candidate"
)

# plot all jurisdictions with panels
rpv_plot(example_rpvDF, panelBy = "Jurisdiction")
# add contest separation shading
rpv_plot(
  example_rpvDF,
  panelBy = "Jurisdiction",
  contest_sep = "s"
)

# plot panels by voter_race and remove legend
rpv_plot(rpvDF_county,
panel_spacing = 6,
panelBy = "Voter_Race") +
  ggplot2::theme(legend.position="none")
```

rpv_toDF *Transform RPV results from* eiCompare *into a simple dataframe object*

Description

Create a dataframe from RPV analysis output to facilitate RPV visualizations. The output dataframe of this function can be used directly in rpv_plot().

Usage

```
rpv_toDF(
  rpv_results = NULL,
  model = NULL,
  jurisdiction = "",
  preferred_candidate = "",
  party = "",
  election_type = "",
  year = "",
  contest = "",
  candidate = ""
)
```

Arguments

rpv_results	RPV analysis results either from the output of ei_iter() or ei_rxc() from the eiCompare package or from the internal function ci_cvap_full().
model	A string indicating the model used to create rpv_results. Examples include "ei", "rxc", "ei cvap", etc.
jurisdiction	A string of the jurisdiction.
preferred_candidate	A character vector of races indicating racial preference of each candidate. The racial preferences must be listed in the correct order with respect to candidate.
party	A character vector containing the political parties of the candidates. Must be listed in the correct order with respect to candidate.
election_type	A string on the election type (usually "General" or "Primary")
year	The year of the contest
contest	A string of contest name as it would appear in an rpv visualization (e.g. "President" or "Sec. of State")
candidate	A character vector of candidate names written as they would appear on a visualization. The candidate names must be listed in the same order as the candidate estimates appear in rpv_results, i.e the same order as the cands argument in eiCompare::ei_iter() or eiCompare::ei_rxc().

Value

rpv results in a data.frame

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

Kassra Oskooii <kassrao@gmail.com>

Examples

```
#library(eiExpand)
#data("south_carolina")
#prec_election_demog <- south_carolina[1:50,]

## run rpv analysis
#eiVote <- eiCompare::ei_iter(
 # data = prec_election_demog,
 # cand_cols = c('pct_mcmaster', 'pct_smith'),
 # race_cols = c('pct_white', 'pct_black'),
 # totals_col = "total_vap"
#) %>%
 # rpv_normalize(
 #   cand_cols = c('pct_mcmaster', 'pct_smith'),
 #   race_cols = c('pct_white', 'pct_black')
 # )

## use function to create dataframe from rpv results
#plotDF <- rpv_toDF(
#   rpv_results = eiVote,
#   model = "ei vap", #since we used ei_iter model normalized with vap denominator
#   jurisdiction = "Statewide",
#   candidate = c("McMaster", "Smith"), #must be in correct order relative to rpv_results
#   preferred_candidate = c("White", "Black"), #must be in correct order rpv_results
#   party = c("Republican", "Democratic"),
#   election_type = "General",
#   year = "2020",
#   contest = "Governor"
# )
```

`south_carolina` *Example election and demographic data from South Carolina 2020 General Elections*

Description

Example election and demographic data from South Carolina 2020 General Elections

Usage

```
south_carolina
```

Format

An object of class `data.frame` with 750 rows and 42 columns.

`split_precinct_analysis`
Split precinct analysis - VAP Adjusted Election Data

Description

Run Split Precinct Analysis using precinct-level geometries and election data, a district shape, and block-level vap data. This function calculates the percent vap of a precinct contained in the district boundary of interest. Then, if specified, multiplies election vote counts by percent vap.

Usage

```
split_precinct_analysis(
  vtd,
  planShp,
  block_pop_data,
  vote_col_names = NULL,
  lower_thresh = 0.02,
  upper_thresh = 0.98,
  keepOrigElection = TRUE,
  generatePlots = FALSE,
  ggmap_object = NULL,
  verbose = FALSE
)
```

Arguments

`vtd`	A sf dataframe with precinct-level geometries potentially in the district from `planShp`. For election adjustments, it should also contain election results in columns defined in `vote_col_names`.
`planShp`	A sf dataframe with one row containing district plan shape boundary (one district).
`block_pop_data`	A sf object of blocks covering the region, with vap column
`vote_col_names`	Character vector containing the name of the columns to be adjusted based on percent vap. This should include election results columns names in `vtd`.
`lower_thresh`	A decimal. If the percent area of a precinct inside the `planShp` is equal to or below this threshold, the precinct will be removed. Defaults to .02.

upper_thresh A decimal. If the percent area of a precinct inside the planShp is equal to or above this threshold, the precinct will be considered to be fully contained in the district. Defaults to .98.

keepOrigElection A boolean indicating if original election vote counts should be preserved in the output dataset for comparison purposes.

generatePlots Boolean indicating if function should generate a list of map checking plots. If TRUE, the function output will include a list of plots that show split precincts and intersecting blocks within and outside of the district.

ggmap_object A ggmap object of the area on interest to be the background of plots if generatePlots = TRUE. If this argument is not specified, plots will be generated without a map background.

verbose A boolean indicating whether to print out status messages.

Value

If generatePlots = FALSE, returns a split precinct results data.frame with vap percentages and adjusted election data. If generatePlots = TRUE, returns a list with the result data.frame in the first element and the list of plots in the second.

Author(s)

Rachel Carroll <rachelcarroll4@gmail.com>

Loren Collingwood <lcollingwood@unm.edu>

Examples

```
library(eiExpand)
library(sf)

# load data and shps
data(planShp); data(vtd); data(mt_block_data)

# filter to a few vtds for this example
vtd <- vtd %>%
  dplyr::filter(
    GEOID20 %in% c("30091000002", "30085000012", "30085000018", "30085000010")
  )

# run split precinct analysis without plots
spa_results <- split_precinct_analysis(
  vtd = vtd,
  planShp = planShp,
  block_pop_data = mt_block_data,
  vote_col_names = c('G16HALRZIN', 'G16HALDJUN', 'G16HALLBRE',
                     'G16GOVRGIA', 'G16GOVDBUL', "G16GOVLDUN"),
  keepOrigElection = TRUE,
  generatePlots = FALSE)

# run with plots
```

```
spa_list <- split_precinct_analysis(
  vtd = vtd,
  planShp = planShp,
  block_pop_data = mt_block_data,
  vote_col_names = c('G16HALRZIN', 'G16HALDJUN', 'G16HALLBRE',
                     'G16GOVRGIA', 'G16GOVDBUL', "G16GOVLDUN"),
  lower_thresh = 0,
  keepOrigElection = TRUE,
  generatePlots = TRUE)

# View results
spa_list[["results"]]

# View plots
#library(gridExtra)
#do.call("grid.arrange", c(spa_list[["plots"]], ncol=1))
```

`vtd` *Example vtd-level sf dataframe with election results for split precinct analysis*

Description

Example vtd-level sf dataframe with election results for split precinct analysis

Usage

```
vtd
```

Format

An object of class `sf` (inherits from `data.frame`) with 19 rows and 12 columns.

`washington` *Example election data with BISG demographics from Washington 2020 General Presidential Election*

Description

Example election data with BISG demographics from Washington 2020 General Presidential Election

Usage

```
washington
```

Format

An object of class `data.frame` with 637 rows and 27 columns.

wa_block_data *Example block-level population data from Washington for BISG*

Description

Example block-level population data from Washington for BISG

Usage

```
wa_block_data
```

Format

An object of class `sf` (inherits from `data.frame`) with 821 rows and 8 columns.

wa_geocoded *Example geocoded voter file from Washington for BISG*

Description

Example geocoded voter file from Washington for BISG

Usage

```
wa_geocoded
```

Format

An object of class `data.frame` with 1000 rows and 6 columns.

Index