Pollspain: the R package for analyzing Spanish elections

# Introduction

Election data is of considerable interest to a broad spectrum of stakeholders, including political analysts, researchers, policymakers, journalists, and the general public. Each of these groups often possesses varying levels of technical expertise, which can present challenges in accessing and interpreting electoral data. Although electoral data is generally available to the public, it is often not presented in a user-friendly format. This can make it difficult for users without advanced programming skills to engage with the data meaningfully. The pollspain R package was developed to address these challenges by providing a centralized, streamlined, and accessible resource for Spanish election data. The package not only facilitates easy access to election results and opinion polling data but also aims to present this information in a format that is easy to understand and use.

The primary objective of the pollspain package is to consolidate data on Spanish election results and opinion polls from a variety of sources, preprocessing it into a format that is immediately accessible to a wide range of users, including those with minimal programming experience. The package is designed to be computationally efficient, minimizing the amount of data that users need to install on their systems to visualize and analyze the information they require. In addition to data retrieval and processing, pollspain offers tools for electoral analysis and visualization, further enhancing its utility for users interested in exploring electoral trends and outcomes in Spain.

The current scope of the pollspain package includes functions that enable users to fetch and process official data on Spanish Congress and Senate elections from 1982 to 2023. This data includes detailed information on the electoral census, candidates, candidacies, and vote shares, all presented at the polling station level. The package is intended for use by a diverse audience, from political scientists and journalists to educators and anyone with an interest in Spanish electoral processes. By simplifying access to this rich dataset, pollspain seeks to empower users to engage more deeply with electoral data, regardless of their technical background.

# Literature review

The Spanish Electoral Archive (SEA), as introduced by Pérez, Aybar, and Pavía in their 2021 paper, represents a significant advancement in the centralization and accessibility of electoral data in Spain. The SEA database offers the most comprehensive repository of Spanish election outcomes available, covering General, Regional, Local, and European Parliamentary elections from the late 1970s to 2021. This database is particularly notable for its inclusion of detailed results down to the ballot box level, making it an invaluable resource for researchers, analysts, and professionals interested in Spanish electoral behavior.

SEA’s approach to data organization and homogenization is a key feature, as it converts the often convoluted and difficult-to-interpret electoral data into a more accessible format. The database includes detailed metadata, ensuring that users can understand and manipulate the data with greater ease. Moreover, SEA is freely available and offers various formats for download, accommodating users with different levels of technical expertise.

While SEA represents a significant leap forward in the centralization and accessibility of Spanish electoral data, there remain several gaps and limitations that the pollspain package aims to address. One of the primary challenges with SEA, and similar existing solutions, is the difficulty for users with minimal technical expertise to access and utilize the data effectively. The raw data, while comprehensive, often requires significant preprocessing before it can be used for analysis, which can be a barrier for users without advanced programming skills.

The pollspain package is designed to overcome these barriers by providing an interface that simplifies data retrieval and processing. Unlike SEA, which offers data primarily in raw formats, pollspain preprocesses the data, allowing users to access and analyze electoral results with minimal effort. This feature is particularly beneficial for stakeholders who need quick and efficient access to data without delving into complex data transformation processes.

Furthermore, pollspain addresses the issue of data granularity and scope. While SEA covers a broad range of elections, pollspain focuses specifically on Spanish congressional and senate elections, providing detailed data at the polling station level. This focus allows for more nuanced analyses of voter behavior and election outcomes within specific local contexts, something that broader databases like SEA might not be able to offer due to their generalized approach.

Finally, pollspain enhances computational efficiency by allowing users to fetch only the data they need, thereby reducing the computational load and storage requirements. This contrasts with SEA, where users might need to download large datasets that include more information than necessary for their specific analysis.

# Package Overview

pollspain is an R package specifically designed to streamline the processing and analysis of Spanish electoral and opinion polling data. The primary purpose of this package is to make it easier for users with minimal programming knowledge and computing resources to access, manage, and analyze comprehensive electoral data. pollspain addresses the common problem of difficult-to-access and poorly organized electoral data by providing a robust solution that preprocesses and organizes the necessary data and auxiliary files required for accurate and efficient analysis.

This package is particularly valuable for users who need to interact with Spanish electoral data at a detailed level, such as political analysts, researchers, journalists, and educators. By centralizing various data sources and simplifying the data processing steps, pollspain allows users to focus more on analyzing and interpreting data rather than dealing with the complexities of data acquisition and preparation.

The pollspain package offers a range of powerful features designed to enhance the user experience in working with Spanish electoral data. One of its primary features is the ability to source data directly from the Spanish Ministry of the Interior. This includes comprehensive datasets on municipal census information, candidacies, candidates, polling stations, and CERA (Censo Electoral de Residentes Ausentes) data, all of which are essential for in-depth electoral analysis.

In addition to sourcing and organizing this data, pollspain provides users with ready-made functions for summarizing and visualizing the information. These functions are designed to be intuitive and easy to use, enabling users to quickly generate insights from the data without requiring extensive programming knowledge. Whether users are interested in exploring voter turnout, candidate performance, or electoral trends at the municipal or national level, pollspain offers the tools necessary to conduct these analyses efficiently.

Another key feature of pollspain is its ability to scrape opinion polling data from a curated set of Wikipedia articles. This functionality allows users to access a comprehensive collection of public opinion polling results related to Spanish elections. By automating the web scraping process, pollspain ensures that users have access to up-to-date polling data, which can be crucial for understanding public sentiment leading up to elections.

Overall, pollspain combines data sourcing, processing, and analysis into a single, user-friendly package that empowers users to engage deeply with Spanish electoral data, regardless of their

# Design and Implementation

## Architecture

The architecture of the pollspain package is designed to facilitate the efficient processing and analysis of Spanish electoral and opinion polling data, leveraging a robust backbone that includes raw data storage, auxiliary datasets, and specialized functions. At the core of this architecture is a GitHub repository named Pollspain-data, which serves as the foundation for the package by hosting all the necessary raw data files and scripts. These scripts not only manage data acquisition but also construct auxiliary datasets that are critical for the package's functionality.

**Pollspain-data Repository: The Backbone of pollspain**

The Pollspain-data repository is a comprehensive storage and processing hub where all the raw electoral data files are kept. These files are sourced directly from the download area of the Spanish Ministry of the Interior, ensuring that users have access to the most accurate and up-to-date information. The repository also includes web scraping scripts that gather public opinion polling data from a set of Wikipedia articles. This data, along with the raw electoral files, forms the basis for several auxiliary datasets that are essential for the correct functioning of pollspain.

The repository supports the package by maintaining and processing several key datasets, including cod\_INE\_mun, dates\_elections\_spain, and all POLL data files. These datasets serve different purposes within the package, from ensuring accurate geographical referencing to providing a chronological framework for the analysis of electoral trends.

### Auxiliary Data: Supporting Robust Functionality

1. **Election Dates in Spain (dates\_elections\_spain)**:
   * This auxiliary dataset is constructed through an R script that scrapes election dates from a Wikipedia page listing Spanish elections. The script utilizes the rvest, tidyverse, and lubridate libraries for web scraping and data manipulation. After scraping, the data is processed to convert Spanish date formats into a standardized format and cleaned to include only relevant columns, such as election type, date, year, month, and day. The final dataset, dates\_elections\_spain, contains 119 rows and 7 columns, offering a comprehensive timeline of various elections in Spain, which is crucial for analyzing electoral patterns over time.
2. **Municipality INE Codes (cod\_INE\_mun)**:
   * The cod\_INE\_mun dataset is created by merging two Excel files: one containing INE codes for Spanish provinces and their corresponding Autonomous Communities (CCAA), and another detailing municipal information. The script ensures consistent formatting and creates unique identifiers for each municipality by concatenating relevant codes. The resulting tibble includes detailed information on each of Spain's 8,132 municipalities, making it a vital resource for linking electoral data to specific geographic and administrative units. This dataset is particularly important for detailed analysis of election results at the municipal level.
3. **Political Party Color Codes (party\_colors\_hex)**:
   * The party\_colors\_hex dataset is a small but essential tibble that maps Spanish political parties to their respective color codes, which are used in data visualizations. Consisting of 21 rows and 3 columns, this dataset ensures consistent and accurate representation of political parties across various visual outputs, allowing for clear and recognizable data interpretation.

### Data Import Functions: Processing Raw Electoral Data

The pollspain package features a suite of functions designed to import and process raw electoral data files from the Ministry of the Interior (MIR). These functions are tailored to handle different types of electoral data, ranging from candidacies to polling stations, ensuring that users can efficiently extract, process, and analyze the data.

1. **import\_raw\_candidacies\_file**:
   * This function imports raw candidacies files by specifying the election type, year, and month. It constructs the appropriate URL to access the data, reads it as a fixed-width file, and processes it into a structured tibble that includes detailed information about the candidates. The function also merges this data with additional contextual information, ensuring a comprehensive dataset for analysis.
2. **import\_raw\_candidacies\_poll\_file**:
   * Focused on importing raw candidacies poll data, this function follows a similar process, allowing users to access detailed poll station data, including the number of ballots cast and results for each candidacy. It handles special cases, such as the processing of CERA data, to ensure accuracy and completeness.
3. **import\_raw\_candidates\_file**:
   * This function is tailored to import and process raw candidate data files, adjusting its logic based on the election year. It captures detailed candidate information, including names, genders, and candidacy codes, and merges this data with election date information for further analysis.
4. **import\_raw\_mun\_MIR\_files**:
   * The import\_raw\_mun\_MIR\_files function processes raw municipal data files, extracting information about municipalities, including population and census data. This function is crucial for detailed electoral analysis at the municipal level.
5. **import\_poll\_stations\_MIR\_files**:
   * This function imports and processes data related to individual poll stations, handling special cases like CERA data and ensuring that the output is ready for analysis.

### Scrapping raw data: Survey Data Scrapper

This web scraping script is designed to collect and process opinion polling data for Spanish general elections from Wikipedia. The process is divided into several stages: generating survey links, scraping HTML content, extracting and cleaning tables, and finally tidying the data.

Generating Survey Links

* The initial stage involves creating a list of URLs to historical opinion polling pages on Wikipedia. The script starts by filtering a dataset that contains information about Spanish elections, focusing specifically on congressional elections from the year 1982 onwards. It constructs the URLs for each survey page based on the year and, in some cases, the month of the survey. This approach ensures that both annual and monthly opinion polling data are included. For years with multiple surveys, URLs are generated for both specific monthly and general yearly pages. This process results in a comprehensive list of links pointing to the relevant Wikipedia pages containing polling data.

Scraping HTML Content

* Once the URLs are generated, the script proceeds to fetch the HTML content from these Wikipedia pages. It uses a loop to go through each URL and download the HTML of the corresponding page. The HTML content is stored in a list, where each entry is associated with a unique identifier based on the year and month of the survey. This step is crucial as it gathers the raw data needed for further processing. Any errors encountered during this process are handled gracefully, ensuring that the script continues to function even if some pages fail to load.

Extracting and Cleaning Tables

* After collecting the HTML content, the script extracts tables from the pages. Wikipedia pages often present polling data in tabular formats, and the script identifies these tables within the HTML content. The extraction process involves parsing the HTML to locate and convert tables into data frames. Once the tables are extracted, they undergo a cleaning process to make them suitable for analysis.

The cleaning involves several steps:

1. \*\*Removing Unnecessary Data\*\*: The script discards irrelevant rows and columns that do not contain meaningful data. This includes removing header rows or columns that do not fit the expected structure.

2. \*\*Standardizing Column Names\*\*: Column names are cleaned and standardized for consistency. This is done by parsing the HTML to retrieve header information, which is then used to rename the columns appropriately.

3. \*\*Handling Special Characters and Formatting\*\*: Data values are cleaned to remove special characters and formatting issues. For instance, turnout values with ranges are averaged, and sample sizes are extracted as numeric values.

4. \*\*Extracting Additional Information\*\*: Additional information, such as the media source from which the polling data was obtained, is extracted and included in the cleaned tables. Dates are processed to ensure they are in the correct format, and fields related to polling duration and exit polling are computed.

Final Tidy-Up and Saving

* The cleaned tables are further processed to ensure they are in a tidy format suitable for analysis. This involves converting the data into a long format, where each row represents a single observation of vote share by party, rather than wide-format tables with multiple party columns. This transformation facilitates easier data manipulation and analysis.

Finally, the tidy data frames are saved to files for future use. The script handles the saving process by ensuring the output directory exists and by saving each data frame in a suitable format, such as RDS or CSV files, depending on the specified requirements.

## Functionality

The `pollspain` package provides a wide array of functions designed to facilitate the efficient handling, analysis, and visualization of Spanish electoral and opinion polling data. The functions are organized into utility functions that support the core operations of the package, functions that retrieve and process election data, and functions that aggregate and analyze this data. Below is a detailed overview of the core functions, their purpose, and examples of their usage.

### Utility Functions

These foundational functions ensure that the core functionalities of the package work as intended by handling data cleaning, standardization, and format conversion.

- \*\*`recod\_parties`\*\*: This function standardizes and recodes political party names and abbreviations within a dataset. It cleans the data by removing special characters, trimming extra spaces, and converting characters to ASCII format. The function then applies pattern matching to recode specific party names and abbreviations into a consistent format. This is particularly useful for consolidating different representations of the same party, ensuring uniformity across the dataset, which is crucial for accurate analysis.

\*\*Usage Example\*\*:

```r

standardized\_data <- recod\_parties(raw\_party\_data)

```

- \*\*Inputs\*\*: A data frame with party names and abbreviations.

- \*\*Outputs\*\*: A cleaned data frame with standardized party names and abbreviations.

- \*\*`extract\_code`\*\*: This function is used to extract specific regional codes from standardized poll station identifiers used in Spanish electoral data. It allows users to retrieve codes corresponding to different levels of geographic aggregation, such as autonomous communities (CCAA), provinces, municipalities, or census sections.

\*\*Usage Example\*\*:

```r

province\_codes <- extract\_code(poll\_station\_id, level = "province")

```

- \*\*Inputs\*\*: A poll station identifier and the desired level of geographic aggregation.

- \*\*Outputs\*\*: A vector containing the extracted codes.

- \*\*`type\_to\_code\_election`\*\*: This utility maps election types (e.g., "congress," "senate," "local") to their corresponding codes used by the Spanish Ministry of the Interior. This function ensures that the correct datasets are accessed and manipulated based on the type of election.

\*\*Usage Example\*\*:

```r

election\_code <- type\_to\_code\_election("congress")

```

- \*\*Inputs\*\*: A string indicating the type of election.

- \*\*Outputs\*\*: A string with the corresponding election code.

### Data Retrieval Functions

These functions are designed to fetch and process specific types of election data from the `pollspain` package’s data repository.

- \*\*`get\_mun\_census\_data`\*\*: This function retrieves and processes municipal-level census data for specific elections in Spain. It downloads the relevant `.rda` files based on the election type, year, and month, and returns a tidy data frame containing detailed information about the municipalities involved in the election.

\*\*Usage Example\*\*:

```r

mun\_census\_data <- get\_mun\_census\_data("congress", 2023, 07)

```

- \*\*Inputs\*\*: The type of election, year, and month.

- \*\*Outputs\*\*: A tidy data frame with municipal census data.

- \*\*`get\_poll\_station\_data`\*\*: This function fetches and processes detailed data for individual polling stations for specified elections. It supports retrieving data for both single and multiple elections and provides extensive information for each polling station, including voter statistics and ballot data.

\*\*Usage Example\*\*:

```r

poll\_station\_data <- get\_poll\_station\_data("congress", 2023, 07)

```

- \*\*Inputs\*\*: The type of election, year, and month.

- \*\*Outputs\*\*: A tidy data frame with polling station data.

- \*\*`get\_candidates\_data`\*\*: This function retrieves detailed information about candidates who participated in specified elections, including their names, gender, and ballot order. The data is organized by province and electoral district, making it easy to analyze the distribution and characteristics of candidates.

\*\*Usage Example\*\*:

```r

candidates\_data <- get\_candidates\_data("congress", 2023, 07)

```

- \*\*Inputs\*\*: The type of election, year, and month.

- \*\*Outputs\*\*: A tidy data frame with candidate data.

- \*\*`get\_candidacies\_data`\*\*: This function processes candidacies data for Spanish elections, allowing users to explore the details of each candidacy, such as election type, date, and associated candidates. The function is particularly useful for historical analyses, as it can handle data across different years and election types.

\*\*Usage Example\*\*:

```r

candidacies\_data <- get\_candidacies\_data("congress", 1986, 06)

```

- \*\*Inputs\*\*: The type of election, year, and month.

- \*\*Outputs\*\*: A tidy data frame with candidacy data.

- \*\*`get\_CERA\_data`\*\*: This function aggregates Census of Absent Residents (CERA) data, which is crucial for analyzing voter turnout at various administrative levels. It summarizes key metrics such as census counts and voter turnout percentages.

\*\*Usage Example\*\*:

```r

CERA\_data <- get\_CERA\_data(poll\_station\_data, level = "municipal")

```

- \*\*Inputs\*\*: A data frame with election data, the desired level of aggregation, and optional parameters for customization.

- \*\*Outputs\*\*: A data frame summarizing CERA data by the specified level.

- \*\*`get\_candidacy\_ballot\_data`\*\*: This function fetches data on the ballots cast for each candidacy in Spanish elections. It allows users to analyze voting patterns and assess the performance of candidacies across different regions and election types.

\*\*Usage Example\*\*:

```r

candidacy\_ballot\_data <- get\_candidacy\_ballot\_data("congress", 2023, 07, include\_names = TRUE)

```

- \*\*Inputs\*\*: The type of election, year, month, and an optional parameter to include candidacy names.

- \*\*Outputs\*\*: A tidy data frame with ballot data for each candidacy.

### Data Usage Functions

These functions are designed to aggregate and analyze the election data retrieved by the previous functions, enabling users to generate insights and visualizations.

- \*\*`aggregate\_election\_data`\*\*: This function summarizes election data by various geographic levels, such as Autonomous Communities (CCAA), Provinces, and Municipalities. It is useful for understanding voting patterns and aggregating results by candidacies across regions.

\*\*Usage Example\*\*:

```r

aggregated\_data <- aggregate\_election\_data(election\_data, scope = "ccaa", group\_by\_candidacy = TRUE)

```

- \*\*Inputs\*\*: A data frame with election data and parameters for aggregation.

- \*\*Outputs\*\*: A data frame with aggregated election data by the specified level.

- \*\*`allocate\_seats\_dhondt`\*\*: This function applies the D'Hondt method to allocate seats in the Spanish Congress based on election results. It aggregates votes by province, applies the D'Hondt method, and returns the final seat distribution.

\*\*Usage Example\*\*:

```r

seat\_distribution <- allocate\_seats\_dhondt(last\_election\_ballots, level = "ccaa")

```

- \*\*Inputs\*\*: A data frame with vote counts and the desired level of aggregation.

- \*\*Outputs\*\*: A data frame with the final seat distribution.

- \*\*`plot\_election\_results`\*\*: This function generates a map of Spain visualizing election results by either province or Autonomous Community. The map colors each region according to the party with the most votes in that area.

\*\*Usage Example\*\*:

```r

election\_map <- plot\_election\_results(election\_data, level = "prov")

```

- \*\*Inputs\*\*: A data frame with election results and the desired level of geographic visualization.

- \*\*Outputs\*\*: A `ggplot2` object displaying the election results on a map of Spain.

- \*\*`plot\_parliament\_distribution`\*\*: This function generates a semicircle plot of parliamentary seat distribution using the `ggparliament` package. It visualizes the number of seats allocated to each party.

\*\*Usage Example\*\*:

```r

parliament\_plot <- plot\_parliament\_distribution(seat\_data)

```

- \*\*Inputs\*\*: A data frame with seat distribution data.

- \*\*Outputs\*\*: A `ggplot2` object representing the seat distribution in a semicircle plot.

Each of these functions contributes to the overall goal of `pollspain`: making Spanish electoral data accessible, analyzable, and visually interpretable for users with varying levels of technical expertise. By combining these functions, users can perform in-depth analyses of Spanish elections, from the municipal level to national outcomes.

# Documentation and Usage

# Testing and Validation

# Discussion