Package 'lpanda'

September 1, 2025

Title Local Political Actor Network Diachronic Analysis Tools

Version 0.1.1

Description Provides functions to prepare, visualize, and analyse diachronic network data on local political actors, with a particular focus on the development of local party systems and identification of actor groups. Formalizes and automates a continuity diagram method that has been previously applied in research on Czech local politics, e.g. Bubenicek and Kubalek (2010, ISSN:1803-8220), Kubalek and Bubenicek (2012, ISSN:1803-8220), and Cmejrek, Bubenicek, and Copik (2010, ISBN:978-80-247-3061-5).

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, igraph (>= 2.1.0), magrittr, RColorBrewer, scales

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/localpolitics/lpanda

BugReports https://github.com/localpolitics/lpanda/issues

Depends R (>= 3.5)

LazyData true

NeedsCompilation no

Author Vaclav Bubenicek [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-0906-0750>)

Maintainer Vaclav Bubenicek <bubenicek@pef.czu.cz>

Repository CRAN

Date/Publication 2025-09-01 09:10:07 UTC

Contents

`Doubice_DC_cz` *Municipal Election Data: Doubice (DC, CZ)*

Description

A dataset containing individual-level candidacy records from municipal elections in the municipality of Doubice (district Decin, Czech republic).

Usage

```
Doubice_DC_cz
```

Format

An object of class `data.frame` with 151 rows and 14 columns.

Details

Dataset overview:

Municipality:	Doubice
District:	Decin
Country:	Czech Republic
Number of elections:	11
Elections covered:	1993, 1994, 1998, 2002, 2006, 2007, 2010, 2014, 2015, 2018, 2022
Number of candidacies (rows):	151
Note:	Municipality website

Description of variables

Variable	Description
elections	Election identifiers (numeric)
candidate	Candidate's full name (character)
list_name	Name of the candidate list (character)
list_pos	Candidate's position on the list (numeric)
pref_votes	Number of preferential votes (numeric)
elected	Logical; TRUE if candidate was elected
nom_party	Nominating party (character)
pol_affil	Political affiliation (character)
mayor	TRUE if elected mayor
dep_mayor	TRUE if elected deputy mayor
board	TRUE if member of the executive board
gov_support	TRUE if supported the local government
elig_voters	Number of eligible voters (numeric)
ballots_cast	Number of ballots cast (numeric)

Each record describes one candidate's run for office, including their candidate list affiliation, position on the list, nominating party, political affiliation, number of preferential votes, and whether they were elected or held specific positions (mayor, deputy mayor, member of the executive body).

The dataset also includes contextual election-level information, such as the number of eligible voters and ballots cast, which can be used to calculate voter turnout and related indicators.

Source

Czech Statistical Office, Municipality website, Acta Politologica article

---

`Jilove_DC_cz` *Municipal Election Data: Jilove (DC, CZ)*

Description

A dataset containing individual-level candidacy records from municipal elections in the municipality of Jilove (district Decin, Czech republic).

Usage

```
Jilove_DC_cz
```

Format

An object of class `data.frame` with 745 rows and 14 columns.

Details

Dataset overview:

Municipality:	Jilove
District:	Decin
Country:	Czech Republic
Number of elections:	8
Elections covered:	1994, 1998, 2002, 2006, 2010, 2014, 2018, 2022
Number of candidacies (rows):	745
Note:	Municipality website

Description of variables

Variable	Description
elections	Election identifiers (numeric)
candidate	Candidate's full name (character)
list_name	Name of the candidate list (character)
list_pos	Candidate's position on the list (numeric)
pref_votes	Number of preferential votes (numeric)
elected	Logical; TRUE if candidate was elected
nom_party	Nominating party (character)
pol_affil	Political affiliation (character)
mayor	TRUE if elected mayor
dep_mayor	TRUE if elected deputy mayor
board	TRUE if member of the executive board
gov_support	TRUE if supported the local government
elig_voters	Number of eligible voters (numeric)
ballots_cast	Number of ballots cast (numeric)

Each record describes one candidate's run for office, including their candidate list affiliation, position on the list, nominating party, political affiliation, number of preferential votes, and whether they were elected or held specific positions (mayor, deputy mayor, member of the executive body).

The dataset also includes contextual election-level information, such as the number of eligible voters and ballots cast, which can be used to calculate voter turnout and related indicators.

Source

Czech Statistical Office, Municipality website, Student thesis

plot_continuity *Visualization of Candidacy Continuity Diagram*

Description

Visualizes the continuity of candidacies over time, illustrating the evolution of the local party system through a network of candidate lists linked by candidate transitions across elections.

Usage

```
plot_continuity(
  netdata,
  mark = NULL,
  separate_groups = FALSE,
  lists = c("all", "elected"),
  elections = NULL,
  show_elections_between = TRUE,
  parties = NULL,
  links = c("continuity", "all"),
  order_lists_by = c("votes", "seats"),
  order_groups_by = c("elections", "votes", "seats"),
  personalization = FALSE,
  coloured = TRUE,
  group_colours = c(),
  show_legend = TRUE,
  plot_title = NULL,
  ...
)
```

Arguments

netdata	A named list created by prepare_network_data containing the continuity network data. Alternatively, a data.frame can also be used, but is recommended only for quick or exploratory plotting of a basic continuity diagram.
mark	Character or character vector. Specifies which type of group should be visually distinguished in the diagram. Options include "parties", "cores", or c("candidate", "candidate name"). Defaults to NULL (no group highlighting). See *Details* and *Examples* for usage.
separate_groups	Logical. If TRUE, groups of candidate lists are plotted in separate rows on the y-axis, improving clarity for group-level analysis. See *Details*.
lists	Character. Candidate lists to be included in the plot. Either "all" (default) or "elected" to include only lists with at least one elected candidate (councillor).
elections	Character or character vector. Filters the range of elections to be shown in the diagram. By default (NULL), all available elections in the netdata object are included. You can specify: individual elections (e.g., "1994", "2022"), ranges

(e.g., "2002-", "-2010", "1994-2010") or combinations of both (e.g., "-1998, 2002, 2003.11, 2018-"). See *Details* and *Examples* for more information and usage.

show_elections_between
: Logical. If TRUE (default), the plot includes all election periods between those selected via the elections argument, even if no candidate lists are present for those years because of the selection. This is especially useful when visualizing groups that did not run in every election — empty columns help preserve the visual continuity of timelines. Setting this to FALSE will omit those gaps. Recommended to keep TRUE when analyzing individual groups or when filtering only a subset of elections.

parties
: Integer or character vector. Filters the so-called political parties, i.e., groups of candidate lists identified via community detection (see prepare_network_data). Use this to display only selected parties, for example: parties = c(1, 3, 5). Party IDs can be found in the network data object under netdata$parties$node_attr$vertices.

links
: Character. Determines which links between candidate lists are plotted. "continuity" (default) includes only connections between *adjacent* elections. "all" includes links across any elections. This option is mainly useful when analyzing a selection of non-consecutive elections.

order_lists_by
: Character. Sorts candidate lists within each election vertically. Options are: "votes" (default) or "seats". If separate_groups = TRUE, sorting is applied within each group.

order_groups_by
: Character vector. Used when separate_groups = TRUE. Specifies the order of groups on the y-axis. Options: "elections", "votes", "seats", or "none". Multiple criteria can be provided in order of priority. To display groups in the order they are listed in netdata, use "none" or NULL. See *Details* for more information.

personalization
: Logical. If TRUE, appends the coefficient of variation of preferential votes to the candidate list name. See *Details* for interpretation. Default is FALSE.

coloured
: Logical. Specifies whether candidate lists of different groups should be distinguished in colour (TRUE, default) or in grayscale when using the mark argument. Ignored if group_colours is provided.

group_colours
: A character vector of colour values (e.g., hex codes or R colour names). Custom colours for marked groups. To maintain the same colours when displaying the diagram repeatedly, the number of colours (elements in the vector) must match the number of all identified groups, even if only a subset is shown. If NULL (default), the function will select the most appropriate colour palette.

show_legend
: Logical. Whether to display the legend (only applies when groups are marked). Default is TRUE.

plot_title
: Character. Title displayed above the diagram. Default is NULL (no title).

...
: Additional technical arguments passed internally, primarily for testing and improving the diagram display.

Details

Recommendation about using the raw data:

For more advanced use, especially when identifying political parties or analyzing system stability, it is recommended to first process the election data using `prepare_network_data`. This function builds the necessary network structures and attributes also for groups of candidate lists that sometimes takes few minutes but you would need to do it only once. Using raw data frames as input in case of `plot_continuity` is intended mainly for quick and basic visualizations, without the group identification.

Usage of `mark` argument:

A central feature of this function is the `mark` argument, which allows highlighting of specific groups in the diagram. The most common options are `"parties"` or `"cores"`, referring to communities of candidate lists detected through community detection.

When using `mark = "parties"` or `"cores"`, you can further specify which groups to highlight visually by adding their IDs (e.g., `mark = c("parties", 2, 5)`). Party and core IDs are available in `netdata$parties$node_attr$vertices` or `netdata$cores$node_attr$vertices`.

You can also highlight individual candidates by using `mark = c("candidate","Candidate Name")`, which will highlight the candidate lists on which the person has appeared in colours of the candidate lists' groups.

You may combine the `mark` argument with group separation, and filtering.

Groups separation:

The `separate_groups` argument improves diagram readability by placing each group on its own line. This is particularly helpful when analyzing continuity, volatility, and structural reproduction of the party system.

Elections filtering:

Filtering elections using the `elections` argument is useful when dealing with many elections that may not fit into a single figure in a report or publication. In such cases, you can split the diagram into two parts (e.g., one with `elections = "-2002"` and one with `elections = "2002-"`, so that the links between the elections adjacent to the 2002 elections are not lost) and stack them vertically.

When selecting non-consecutive elections, it is **strongly recommended** to set `links = "all"` to retain meaningful connections between candidate lists across time. Otherwise, continuity may appear broken due to missing intermediate elections.

For a meaningful continuity analysis, include at least two elections.

About `order_groups_by` argument:

The `order_groups_by` argument is relevant only when `separate_groups = TRUE`. You can sort groups by `"elections"`, `"votes"`, `"seats"`, or `"none"` (the original order in the data). If multiple criteria are provided (e.g., `c("votes", "elections")`), they are applied in priority order. The criteria of `"votes"` and `"seats"` will sort the groups according to the value of the given criterion. The `"elections"` criterion ranks groups based on their participation in the most recent election and falls back recursively to earlier ones in case of ties.

About `personalization` argument:

The `personalization` option appends the coefficient of variation of preferential votes to the name of each candidate list. A lower value may indicate a party's electoral program voting,

while higher variability may suggest a personalized choice (for example, where voters support a prominent individual rather than the whole candidate list). In the case of a limited number of preferential votes, such an interpretation may be debatable and should therefore be used with caution.

Value

NULL, invisibly. Called for its side effect: plotting the continuity diagram.

Note

The mark = "cores" option is currently experimental, as the conversion of their definition into code is still being sought, and may be subject to change in future versions. Use with caution.

Examples

```
data(sample_data, package = "lpanda")

# basic continuity diagram
plot_continuity(sample_data)

# preparing network data
netdata <- prepare_network_data(sample_data, verbose = FALSE, quick = TRUE)

# highlighting groups
plot_continuity(netdata, mark = "parties")
plot_continuity(netdata, mark = c("parties", 3), order_lists = "seats")
plot_continuity(netdata, mark = "parties", separate_groups = TRUE, show_legend = FALSE)

# candidate tracking
plot_continuity(netdata, mark = c("candidate", "c03"))

# filtering elections and parties
plot_continuity(netdata, mark = "parties", elections = "18-")
plot_continuity(netdata, elections = c(14, 22), links = "all", show_elections_between = FALSE)
plot_continuity(netdata, parties = 1)
```

---

prepare_network_data *Prepare Network Data for LPANDA*

Description

Transforms time series data of local election results into a set of network data for use in Local Political Actor Network Diachronic Analysis (LPANDA). The function constructs a bipartite network (candidate – candidate list), its projected one-mode networks (candidate – candidate and list – list), a continuity graph (linking candidate lists between adjacent elections), and an elections network (its node attributes can serve as electoral statistics). It also detects parties (as clusters of candidate lists based on community detection applied to the bipartite network) and constructs their network.

Usage

```
prepare_network_data(df, input_variable_map = list(), verbose = TRUE, ...)
```

Arguments

`df`	A data.frame containing data from elections, with one row per candidate. The function also accepts a single election, though diachronic outputs will then be empty or trivial. See the *Expected structure of input data* section for the expected data format and required variables.
`input_variable_map`	A list mapping variable names in `df` that differ from the expected ones: `elections` = **unique** election identifiers (numeric), `candidate` = candidate's name used as a **unique** identifier (character), `list_name` = name of the candidate list (character), `list_pos` = candidate's position on the list (numeric), `pref_votes` = preferential votes received by the candidate (numeric), `list_votes` = * total votes received by the candidate list (numeric), `elected` = whether the candidate was elected (logical), `nom_party` = party that nominated the candidate (character), `pol_affil` = declared political affiliation of the candidate (character), `mayor` = whether the councillor became mayor (logical), `dep_mayor` = whether the councillor became deputy mayor (logical), `board` = whether the councillor became a member of the executive board (logical), `gov_support` = whether the councillor supported the executive body (logical), `elig_voters` = * number of eligible voters (numeric), `ballots_cast` = * number of ballots cast (numeric), `const_size` = * size of the constituency (number of seats) (numeric) * Variables marked with an asterisk should appear only once per election and constituency — in the row of any **one** candidate running in that specific elections and constituency. See the *Expected input data structure* section to find out how to use it.
`verbose`	Logical, default TRUE. If FALSE, suppresses informative messages.
`...`	Optional arguments reserved for internal development, experimental features and future extensions, such as `include_cores` (logical, default FALSE). Not intended for standard use yet (behavior may change without notice). Unknown keys in ... are ignored.

Value

A list of network data objects for diachronic analysis using LPANDA or other social network analysis tools. Each component contains `edgelist` (data.frame of edges) and `node_attr` (data.frame of node attributes). The exact set of columns depends on the input and may evolve. See *Output data structure* for a description of the returned object.

Expected structure of input data

The input data frame (`df`) **must** include at least the election identifiers (year[.month]), candidates' names (uniquely identifying individuals), and list names. Other variables are optional. If variable names in the dataset differ from the expected ones, they should be specified in the `input_variable_map` as a named list (only differing names need to be listed).

Just in case - a named list is a list where each element has a name (the expected variable name) and a value (the actual name used in your data frame), for example: `list(list_name = "party", elected = "seat", list_votes = "votes_total")`.

Examples of expected and acceptable values in `df`:

- `elections` (required): Election identifier in the format YY[YY][.MM]: e.g., 94 | 02 | 1998 | "2024" | 2022.11
- `candidate` (required): e.g., "John Doe" | "John Smith (5)" | "Jane Doe, jr."
- `list_name` (required): *for independent candidates, you can use:* e.g., "John Smith, Independent Candidate" | "J.S., IND."
- `list_pos`, `pref_votes`, `list_votes`: must be numeric
- `elected`, `mayor`, `dep_mayor`, `board`, `gov_support`: 1 | "0" | T | "F" | "TRUE" | FALSE (non-logical inputs will be coerced to logical).
- `nom_party`: *for independent candidates, you can use:* "IND" | "Independent Candidate"
- `pol_affil`: *for independent candidates, you can use:* "non-partisan"
- `elig_voters`, `ballots_cast`, `const_size`: A numeric that should appear only once in any candidate row within a given election and constituency

If `pref_votes` are present but `list_votes` are not, the function assumes a voting system where list votes are calculated by summing the preferential votes of candidates on the list.

If `const_size` is missing, it will be estimated based on the number of elected candidates (if available).

For the purposes of analysis, a new variable `list_id` (class character) is added to the internally processed copy of `df` and carried to the output. It uniquely identifies each candidate list in a given election (combining `list_name` and `elections`), e.g., *Besti Flokkurinn (2010)*, *SNP (2019)*, or *"John Smith (5), IND. (2022.11)"*. This variable serves as a **key identifier** in LPANDA for tracking candidate lists across elections and constructing network relations.

Output data structure

The returned object is a named list with up to seven network objects:

- `bipartite`: bipartite network (candidates-lists).
- `candidates`: projected candidate–candidate network.
- `lists`: projected list–list network (directed by election order).
- `continuity`: filtered version of `lists` network (edges of adjacent elections only).
- `parties`: network of detected party clusters (via community detection applied on `bipartite` network).
- (`cores`): higher-level clusters of `parties`. Cores are currently experimental and will not appear in the standard output network data. See **Note** below.

- `elections`: inter-election candidate flow and election-level stats

Each object is a list with two components:

- `edgelist`: a data.frame representing network edges
- `node_attr`: a data.frame with attributes for each node

For example, `...$candidates$edgelist` contains edges between individuals who appeared on the same candidate list, and `...$elections$node_attr` includes several election statistics (e.g., number of candidates, distributed seats, plurality index, voter turnout for each election, etc.).

Note

Cores are currently experimental, as the conversion of their definition into code is still being sought, and may be subject to change in future versions. It is not yet intended for standard use in analyses and academic studies, since their calculation is not yet comprehensive, so the cores' network structure will not appear in the standard output network data unless explicitly called with the `include_cores = TRUE` parameter. Use with caution, their interpretation is highly questionable.

Examples

```
data(sample_diff_varnames, package = "lpanda")
df <- sample_diff_varnames
str(df) # different variable names: "party" and "seat"
input_variable_map <- list(list_name = "party", elected = "seat")

netdata <- prepare_network_data(df, input_variable_map, verbose = FALSE)
str(netdata, vec.len = 1)
```

---

sample_binary_values *Sample Dataset with Binary Values*

Description

A variant of `sample_data` containing binary values instead of TRUE/FALSE values.

Usage

```
sample_binary_values
```

Format

A data frame with 18 rows and 5 variables (same structure as `sample_data`).

Source

Fictitious data

sample_data *Simple Sample Dataset*

Description

Basic fictitious dataset simulating election results.

Usage

```
sample_data
```

Format

A data frame with 18 rows and 5 variables:

elections Election identifier (numeric)

candidate Candidate identifier (character)

list_name Candidate list name (character)

elected Logical; `TRUE` if the candidate was elected

mayor Logical; `TRUE` if the candidate became mayor

Source

Fictitious data

sample_diff_varnames *Sample Dataset with Some Different Variable Names*

Description

A variant of `sample_data` with different names for variables list_name (party) and elected (seat), useful for testing robustness of input handling.

Usage

```
sample_diff_varnames
```

Format

A data frame with 18 rows and 5 variables (same structure as `sample_data`).

Source

Fictitious data

sample_no_continuity *Sample Dataset Without Continuity Between Elections*

Description

A variant of sample_data in which no candidate appears in more than one election. This breaks the continuity between elections, making it useful for testing whether network-building functions correctly handle datasets with no longitudinal links between candidate lists.

Usage

```
sample_no_continuity
```

Format

A data frame with 15 rows and 5 variables (same structure as sample_data).

Source

Fictitious data

sample_no_crossing *Sample Dataset Without Candidate Switching*

Description

A variant of sample_data in which candidates may run in multiple elections, but always remain within the same political group (they do not switch between candidate list clusters). This is useful for testing continuity logic and verifying that no cross-group transitions occur.

Usage

```
sample_no_crossing
```

Format

A data frame with 15 rows and 5 variables (same structure as sample_data).

Source

Fictitious data

Index