Package: electoral (via r-universe)

September 10, 2024

Type Package

Title Allocating Seats Methods and Party System Scores

Version 0.1.3

Author Jorge Albuja

Maintainer Jorge Albuja <albuja@yahoo.com>

Description Highest averages & largest remainders allocating seats methods and several party system scores. Implemented highest averages allocating seats methods are D'Hondt, Webster, Danish, Imperiali, Hill-Huntington, Dean, Modified Sainte-Lague, equal proportions and Adams. Implemented largest remainders allocating seats methods are Hare, Droop, Hangenbach-Bischoff, Imperial, modified Imperial and quotas & remainders. The main advantage of this package is that ties are always reported and not incorrectly allocated. Party system scores provided are competitiveness, concentration, effective number of parties, party nationalization score, party system nationalization score and volatility. References: Gallagher (1991) <doi:10.1016/0261-3794(91)90004-C>. Norris (2004, ISBN:0-521-82977-1). Consejo Nacional Electoral del Ecuador (2014)<http://cne.gob.ec/documents/Estadisticas/Atlas/ATLAS/CAPITULO%206%20web.pdf>. Laakso & Taagepera (1979) <https://escholarship.org/uc/item/703827nv>. Jones & Mainwaring (2003) <https://kellogg.nd.edu/sites/default/files/old_files/documents/304_0.pdf>. Pedersen (1979) <https://janda.org/c24/Readings/Pedersen/Pedersen.htm>. Golosov (2010) <https://ppq.sagepub.com/content/16/2/171.abstract>. Golosov (2014) <https://ppq.sagepub.com/content/early/2014/09/08/1354068814549342.abstract>.

License GPL-3

Encoding UTF-8

Imports dplyr, ineq, tibble

RoxygenNote 6.0.1

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

RemoteUrl https://github.com/albuja/electoral

RemoteRef HEAD

RemoteSha 603995deef26abfd8c9487a12e1601bb65352bc8

Contents

`competitiveness` *Electoral Competitiveness*

Description

Electoral competitiveness measures the difference in the percentages of valid votes obtained by the two most voted parties in a given territorial unit. It identifies the level of rivalry between the parties who occupied the first and second places in an election. Consejo Nacional Electoral del Ecuador (2014). The formula is:

$$Competitiveness = 1 - (p1 - p2)$$

where p1 and p2 are the 2 votes o seats proportions for the 2 most voted parties. Thus, it requires elections with at least 2 parties.

The index is a number from 0 to 1, 0 means no competitiveness (p1 = 1 and p2 = 0) and 1 means maximum competitiveness (p1 = p2).

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
competitiveness(votes)
```

Arguments

`votes`	2_or_more-length vector of number/share of votes/seats per party

Value

A single numeric with electoral competitiveness value in the interval [0, 1]

Examples

```
competitiveness(votes = c(100, 150, 60))
```

`concentration` *Electoral Concentration*

Description

Electoral concentration is a measure of the accumulation of votes in the two most voted parties of a given territorial unit. It is the cumulative fraction of valid votes obtained by the two most voted parties in a given election (Consejo Nacional Electoral - Ecuador, 2014).

The formula is:

$$concentration = p1 + p2$$

where p1 is the vote/seat proportion for the most voted party, and p2 is the vote/seat proportion for the second most voted party. Thus, it requires elections with at least 2 parties.

The index is a number positive up to 1. A value of 1 means maximum concentration (p1 + p2 = 1, i.e. pi = 0 for i = 3,4...)

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
concentration(votes)
```

Arguments

`votes`	2_or_more-length vector of number o proportion of votes/seats per party

Value

A single numeric with concentration value in the interval (0, 1].

Examples

```
concentration(votes = c(100, 150, 60))
```

enp *Effective Number of Parties (ENP)*

Description

Computes the effective number of parties. The effective number of parties is a concept introduced by Laakso and Taagepera (1979) which provides for an adjusted number of political parties in a country"s party system. The idea behind this measure is to count parties and, at the same time, to weight the count by their relative strength.

Measuring how many parties, weighted according to size, are in a party system in a given election, the effective number of (electoral/legislative) parties is calculated employing the following formula:

$$ENP = 1/sum(pi^2)$$

where pi is the share of votes/seats of the ith party.

A new approach developed by Golosov, Grigorii V. "The Effective Number of Parties: A New Approach" (2010) (http://ppq.sagepub.com/content/16/2/171.abstract) is also available.

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
enp(votes, method)
```

Arguments

votes	vector of number/share of votes/seats per party
method	string to choose from c("Laakso-Taagepera", "Golosov"). Defaults to "Laakso-Taagepera"

Value

«««< HEAD A single numeric with the effective number of parties (ENP) value. ======= A single numeric with the effective number of parties value. »»»> 1e569c45b115684281e88833fa245b2c223ee30a

Examples

```
enp(votes = c(94545, 74162, 25273))
enp(votes = c(94545, 74162, 25273), "Golosov")
```

pns *Party Nationalization Score (PNS)*

Description

Party nationalization score is a measure for the uniformity of vote share of a party over subnational units (provinces for Ecuador). It is computed as 1 minus Gini inequality index (Jones and Mainwaring, 2003)

$$PNS = 1 - G$$

$$G = (2 * sum(i * xi))/(n * sum(xi)) - (n + 1)/n$$

in which G is the Gini inequality index, xi is the vote share in the province ith, indexed in non-decreasing order (xi <= xi+1) i is an index from 1 to n n is the total of provinces

Party nationalization score is a number from 0 to 1, a low value (near 0) means a low level of nationalization, i.e. heterogeneous distribution of vote shares in subnational territorial units.

High score (near 1) indicates a high level of nationalization, i.e. homogeneous distribution of vote shares in subnational territorial units.

A new approach developed by Golosov (2014) 'Party System Nationalization: The Problems of Measurement with an Application to Federal States' (http://ppq.sagepub.com/content/early/2014/09/08/1354068814549342.a is also available.

See psns function for further information.

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
pns(subnational_shares, method)
```

Arguments

`subnational_shares`
vector of vote shares per subnational unit (province) for a single party

`method` string to choose from c('Jones-Mainwaring', 'Golosov'). Defaults to 'Jones-Mainwaring'

Value

A single numeric with party nationalization score of a party, in the interval [0, 1]

Examples

```
pns(subnational_shares = c(0.467265, 0.542505))
pns(subnational_shares = c(0.467265, 0.542505), method = 'Golosov')
```

`psns` *Party System Nationalization Score (PSNS)*

Description

Party System Nationalization Score is a measure for the uniformity of vote share of a party system over subnational units (provinces for Ecuador). It is computed as the sum of part nationalization scores, weighted by the national share of every party (Jones and Mainwaring, 2003).

$$PSNS = sum(PNSi * pi)$$

where PNSi is the party nationalization score for party ith, and pi is the national vote share for party ith.

Party system nationalization score is a number from 0 to 1, low value (near 0) means a low level of nationalization, i.e. heterogeneous distribution of vote shares in subnational territorial units.

High score (near 1) indicates a high level of nationalization, i.e. homogeneous distribution of vote shares in subnational territorial units.

A new approach developed by Golosov (2014) 'Party System Nationalization: The Problems of Measurement with an Application to Federal States' (http://ppq.sagepub.com/content/early/2014/09/08/1354068814549342.a is also available.

Input tibble must have 3 columns with fixed names: 'PROVINCE', 'PARTY and 'VOTES'.

See `pns` function for further information.

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
psns(tidy_votes, method)
```

Arguments

`tidy_votes` a tibble/data.frame in tidy format with 3 variables (columns): 'PROVINCE', 'PARTY' and 'VOTES', where:

1. PROVINCE: names or codes of subnational units.
2. PARTY: names or codes of national parties.
3. VOTES: number of votes for every PARTY in every PROVINCE.

`method` string to choose from c('Jones-Mainwaring', 'Golosov'). Defaults to 'Jones-Mainwaring'

Value

A single numeric with Party System Nationalization Score (PSNS) in the interval [0, 1]

Examples

```
data <- data.frame(PROVINCE = c(1, 1, 1, 2, 2, 2),
               PARTY = c('1', '2', '3',  '1', '2', '3'),
               VOTES = c(66389, 55372, 20319, 28156, 18790, 4954))
psns(data)
psns(data, method = 'Golosov')
```

---

seats *Allocating Seats by any method*

Description

This function is a wrapper for seats_ha and seats_lr functions. A shortcut to allocate seats by any method selected by its string name.

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
seats(parties, votes, n_seats, method)
```

Arguments

`parties`	vector of names of parties, must be uniques
`votes`	vector of votes, same length as parties.
`n_seats`	number of seats to be allocated (integer equal or greater 1).
`method`	string according seat allocating method: "dhondt", "webster", "danish", "imperiali", "hill-huntington", "dean", "mod-saint-lague", "equal-proportions", "adams", "hare", "droop", "hangenbach-bischoff", "imperial", "mod-imperial" or "quotas-remainders".

Value

A tibble

Examples

```
seats(parties = c("A", "B", "C"),
       votes = c(100, 150, 60),
       n_seats = 5,
       method = "hare")

seats(parties = c("V", "W", "X", "Y", "Z"),
      votes = c(100, 150, 60, 80, 160),
      n_seats = 15,
      method = "droop")

seats(parties = c("A", "B", "C"),
       votes = c(100, 150, 60),
       n_seats = 5,
       method = "dhondt")

seats(parties = c("V", "W", "X", "Y", "Z"),
      votes = c(100, 150, 60, 80, 160),
      n_seats = 15,
      method = "webster")
```

seats_ha *Allocating Seats - Highest Averages*

Description

Highest averages is the name for a variety of ways to allocate seats proportionally for representative assemblies with party list voting systems. It requires the number of votes for each party to be divided successively by a series of divisors. This produces a table of quotients, or averages, with a row for each divisor and a column for each party. The nth seat is allocated to the party whose column contains the nth largest entry in this table, up to the total number of seats available. Different methods uses different series of divisors:

If there are n_seats seats to allocate these methods requires n_seats divisors (from 1 to n_seats), being the nth divisor:

D'Hondt: n

Webster: 2 * n - 1

Danish: 3 * n - 2

Imperiali: n + 1

Hill-Huntington: sqrt(n * (n + 1))

Dean: (2 * n) * (n + 1) / (2 * n + 1)

Modified Sainte-Lague: (10 * n - 5) / 7

Equal proportions: sqrt(n * (n - 1))

Adams: n - 1

In case of ties, the implemented algorithm not allocate the involved seats, and prints how many seats had been allocated and how many are in tie.

More information: https://en.wikipedia.org/wiki/Highest_averages_method

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
seats_ha(parties, votes, n_seats, method)
```

Arguments

`parties`	vector of names of parties, must be uniques
`votes`	vector of votes, same length as parties.
`n_seats`	number of seats to be allocated (integer equal or greater 1).
`method`	string according seat allocating method: "dhondt", "webster", "danish", "imperiali", "hill-huntington", "dean", "mod-saint-lague", "equal-proportions" or "adams".

Value

A vector of the number seats allocated for each party in "parties", with the same length as parties and votes.

Examples

```
seats_ha(parties = c("A", "B", "C"),
       votes = c(100, 150, 60),
       n_seats = 5,
       method = "dhondt")

seats_ha(parties = c("V", "W", "X", "Y", "Z"),
      votes = c(100, 150, 60, 80, 160),
      n_seats = 15,
      method = "webster")
```

`seats_lr` *Allocating Seats - Largest Remainders*

Description

Largest remainders methods require the numbers of votes for each party to be divided by a quota representing the number of votes required for a seat (i.e. usually the total number of votes cast divided by the number of seats, or some similar formula). The result for each party will usually consist of an integer part plus a fractional remainder. Each party is first allocated a number of seats equal to their integer. This will generally leave some seats unallocated: the parties are then ranked on the basis of the fractional remainders, and the parties with the largest remainders are each allocated one additional seat until all the seats have been allocated. This gives the method its name. The quota formula for each largest remainder method is:

Hare: (sum(votes))/n_seats

Droop: (sum(votes))/(n_seats + 1) + 1

Hangenbach Bischoff: (sum(votes))/(n_seats + 1)

Imperial: (sum(votes))/(n_seats + 2)

Modified Imperial: (sum(votes))/(n_seats + 3)

Quotas & remainders:

1. threshold: select all parties that meet votes_i >= (sum(votes))/(2*n_seats)
2. quota: (sum(votes))/n_seats (only over selected parties)

In case of ties, the implemented algorithm not allocate the involved seats, and prints how many seats had been allocated and how many are in tie.

More information: https://en.wikipedia.org/wiki/Largest_remainder_method

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
seats_lr(parties, votes, n_seats, method)
```

Arguments

`parties` vector of names of parties, must be uniques

`votes` vector of votes, same length as parties.

`n_seats` number of seats to be allocated (integer equal or greater 1).

`method` string according seat allocating method: "hare", "droop", "hangenbach-bischoff", "imperial", "mod-imperial" or "quotas-remainders".

Value

A vector of the number seats allocated for each party.

Examples

```
seats_lr(parties = c("A", "B", "C"),
       votes = c(100, 150, 60),
       n_seats = 5,
       method = "hare")

seats_lr(parties = c("V", "W", "X", "Y", "Z"),
      votes = c(100, 150, 60, 80, 160),
      n_seats = 15,
      method = "droop")
```

`volatility` *Electoral Volatility*

Description

Defined as the net change within the electoral party system resulting from individual vote transfers, electoral volatility is measured according to the following formula:

$$Volatility = sum|pi, t-1 - pi, t|/2$$

in which pi,t is the vote/seat share for the party ith at a given election (t) and pi,t-1 is the vote/seat share of the same party ith at the previous elections (t-1) (Pedersen, 1979).

The index is a number from 0 to 1, 0 means no volatility (proportion of votes/seats remains constant for every party) and 1 means total volatility (every party pass from 0 to any votes or viceversa).

Developed by Jorge Albuja Delgado (albuja@yahoo.com).

Usage

```
volatility(votes_1, votes_2)
```

Arguments

votes_1	vector of number/share of votes/seats per party at time t-1 (previous election)
votes_2	vector of number/share of votes/seats per party at time t (given election

Value

A single numeric with volatility value in the interval [0, 1]

Examples

```
volatility(votes_1 = c(100, 150, 60),
           votes_2 = c(80, 120, 100))
```

Index