CLEA
EFFECTIVE NUMBER OF PARTIES
AND
PARTY NATIONALIZATION DATA

Codebook

Version: June 17, 2019
(20190617)

TABLE OF CONTENTS

INTRODUCTION

Effective Number of Parties and Party Nationalization Project Description

The Effective Number of Parties and Party Nationalization datasets uses the Constituency-Level Elections Archive (CLEA) data (described below) to generate the effective number of parties as well as several measures of party nationalization that are described in the political science literature. We present these data in three separate datasets that represent different levels of aggregation: the national level, the party level, and the constituency level. The Effective Number of Parties and Party Nationalization data are provided separately for the lower chamber and upper chamber. This codebook applies to both sets of data as they contain the same variable list and descriptions. Files associated with the lower chamber data begin with '*clea_lc_enp*' and files associated with the upper chamber data begin with '*clea_uc_enp*'.

At the national level, we present three measures of the effective number of parties (Laakso and Taagepera, 1979) and nine measures of party system nationalization, or the degree to which the set of political parties in a given country-year have electoral success at the national level (as opposed to more localized or regionalized success).

At the party level, we provide four measures that indicate the level of nationalization for a particular party in a given country-year.

Finally, the constituency level dataset includes the national measures for each constituency in a given country-year, the effective number of parties at the constituency level, and a local inflation measure.

Details about the interpretation and application of the measures are available in the references provided for each variable. Summaries of the measures are provided in this codebook and the appendix provides full computer code for calculations in *Stata* and *R* software.

CLEA Project Description

The central aim of the Constituency-Level Elections Archive (CLEA) project is to produce a repository of detailed results – i.e. votes received by each candidate/party, total votes cast, number of eligible voters – at a constituency level for the lower chamber and upper chamber legislative elections that have been conducted around the world. Our motivation is to preserve and consolidate these valuable data in one comprehensive reference resource that is publicly available at no cost. This public good is expected to be of use to a range of audiences for purposes of research, education, policy-making, and evaluation.

Funding Support

We gratefully acknowledge the American National Science Foundation (www.nsf.gov) and the Taiwan Foundation for Democracy for their funding of the project, and we thank the Institute of Political Science at the University of St. Gallen and the Department of Political Science at the University of Zurich for research assistance. The Center for Political Studies (CPS) at the University of Michigan has contributed financially and in kind to the CLEA project.

Bibliographic Citation

APA (6th edition)
Kollman, K., Hicken, A., Caramani, D., Backer, D., & Lublin, D. (2019). *Constituency-Level Elections Archive* [data file and codebook]. Ann Arbor, MI: Center for Political Studies, University of Michigan [producer and distributor]. Retrieved from http://www.electiondataarchive.org.

MLA (8th edition)
Kollman, Ken, Allen Hicken, Daniele Caramani, David Backer, and David Lublin. *Constituency-Level Elections Archive*. Ann Arbor, MI: Center for Political Studies, University of Michigan [producer and distributor], 2019. Web. 17 June 2019. <http://www.electiondataarchive.org>

Chicago (17th edition)
Kollman, Ken, Allen Hicken, Daniele Caramani, David Backer, and David Lublin. 2019. *Constituency-Level Elections Archive*. Produced and distributed by Ann Arbor, MI: Center for Political Studies, University of Michigan, http://www.electiondataarchive.org.

These materials are based on work supported by the National Science Foundation (under grants SES-0617101, SES-1021915, and SES-1528262). Any opinions, findings and conclusions or recommendations expressed in these materials are those of the author(s) and do not necessarily reflect the views of the funding organizations.

CLEA Credits

The co-directors of the CLEA project are Ken Kollman of the University of Michigan, Allen Hicken of the University of Michigan, Daniele Caramani of the University of Zurich, David Backer of the University of Maryland, and David Lublin of American University. The project manager is Yioryos Nardis of the Center for Political Studies, University of Michigan.

Data in this and prior releases have been contributed by David Lublin, Jan Teorell, Jose Manuel Magallanes, Nir Atmor, David Backer, Daniele Caramani, Adam Carr, Cengiz Erisen, Federico Ferrara, Brian Gaines, Judy Geist, Allen Hicken, Ken Kollman, Arend Lijphart, Scott

Morgenstern, Jairo Nicolau, Daniel Posner, Jae-Jae Spoon, Marcelo Leiras, Steven Reed, Ching-hsing Yu, Matt Singer, Heather Stoll, Jack Vowles, Sarah Shair-Rosenfield, Daniel Bochsler, Joel Selway, Francesca Jensenius, Gilles Verniers, Yen-Pin Su, Binod Paudel, and Anthony Sayers, as well as ICPSR, The Project on Political Transformation and the Electoral Process in Post-Communist Europe program at the University of Essex, and The Trivedi Centre for Political Data at Ashoka University. Research assistance was provided by Fabricio Vasselai, Kirill Kalinin and Sandra Nwogu.

Data Procedures

In order to construct this dataset, we have used the CLEA data to recreate several measures described in the party nationalization literature. We have attempted to construct the measures in accordance with the procedures described by the originating authors. Should readers have questions about the interpretation of a variable, the originating work (cited in the description of each variable in this codebook) should be considered the authoritative source. Additionally, although great care has been taken to prepare the data and codebook prior to release, we would appreciate receiving your comments, feedback and notifications of any remaining errors by email to: clea-project@umich.edu

Calculation of Variables in ENP Dataset

Variety in electoral systems across countries presents challenges on how to treat party vote totals, constituency and national vote totals, and proportions. The measures in these datasets require calculations of votes cast, votes counted for parties, and votes sometimes aggregated into larger units. These calculations are used in formulae for measures of nationalization. The challenges mean that we need to make decisions that may not seem obvious to all users of the data. As an example, in Germany's mixed system, should the first and second votes be pooled within each constituency to give a party vote for that constituency? In fact, we do not pool; instead, the upper tiers are treated as separate constituencies. We typically do this for countries with upper tiers. This is a choice (perhaps arbitrary), but we are following two principles, one theoretical and one practical.

First, as a general rule in CLEA we try as best we can to organize the data in the main CLEA dataset around how voters encounter their options on ballots. If, for instance, candidate names are given on the ballot and that's how voters choose, then candidates' votes are given in the database. If only parties are listed on ballots as options, then parties are the units with votes in the database. This principle cannot always be followed and we are at the mercy of how data are released by governments. But when faced with decisions about how to organize data, we follow the principle focusing on voters' options on ballots. Second, the manner in which data are organized in the main CLEA dataset determines how the calculations for nationalization are conducted for these datasets here. Thus, in our example on Germany, since the upper tiers are considered separate constituencies in the large elections database, they are treated that way for calculating nationalization measures. The computer code at the end of this codebook shows, for instance, that to calculate party vote totals in each constituency, we use the party codes as they are assigned for the main CLEA dataset. Refer to the computer code in the appendix to this codebook and to the main CLEA dataset for assistance with questions about the treatment of specific countries. And as always, feel free to give comments, feedback and notifications of any remaining errors by email to: clea-project@umich.edu

DOCUMENTATION OF VARIABLES

Variable List

National Dataset

ID	Election Identifier
CTR_N	Country Name
CTR	Country Code
YR	Election Year
MN	Election Month
CST_TOT	Total Number of Constituencies
NVVI	National Valid Vote Indicator
ENP_NAT	Effective Number of Parties (National)
ENP_AVG	Effective Number of Parties (Average)
ENP_WGHT	Effective Number of Parties (Weighted Average)
INFLATION1	Cox Inflation Score
INFLATION2	Moenius and Kasuya Inflation Score
INFLATION3	Moenius and Kasuya Weighted Inflation Score
INFLATION4	Kasuya and Moenius Inflation and Dispersion Score
PSNS	Party System Nationalization Score
PSNS_S	Standardized Party System Nationalization Score
PSNS_W	Weighted Party System Nationalization Score
PSNS_SW	Standardized and Weighted Party System Nationalization Score
LOCAL_E	Local Entrant Measure

Party Dataset

ID	Election Identifier
CTR_N	Country Name
CTR	Country Code
YR	Election Year
MN	Election Month
CST_TOT	Total Number of Constituencies
PTY_N	Party Name
PTY	Party Code
PNS	Party Nationalization Score
PNS_S	Standardized Party Nationalization Score
PNS_W	Weighted Party Nationalization Score
PNS_SW	Standardized and Weighted Party Nationalization Score

Constituency Dataset

ID	Election Identifier
CTR_N	Country Name
CTR	Country Code
YR	Election Year
MN	Election Month
CST_N	Constituency Name
CST	Constituency Code
CST_TOT	Total Number of Constituencies
NVVI	National Valid Vote Indicator
CVVI	Constituency Valid Vote Indicator
ENP_CST	Effective Number of Parties (Constituency)
ENP_NAT	Effective Number of Parties (National)
ENP_AVG	Effective Number of Parties (Average)
ENP_WGHT	Effective Number of Parties (Weighted Average)
INFLATION1	Cox Inflation Score
INFLATION2	Moenius and Kasuya Inflation Score
INFLATION3	Moenius and Kasuya Weighted Inflation Score
INFLATION4	Kasuya and Moenius Inflation and Dispersion Score
INFLATION5	Moenius and Kasuya Local Inflation Score
PSNS	Party System Nationalization Score
PSNS_S	Standardized Party System Nationalization Score
PSNS_W	Weighted Party System Nationalization Score
PSNS_SW	Standardized and Weighted Party System Nationalization Score
LOCAL_E	Local Entrant Measure

DOCUMENTATION OF VARIABLES

Variable Descriptions

ID ***Election Identifier***

This variable uniquely identifies each election in the archive.

NOTE: Early U.S. elections to the House of Representatives were scheduled differently across states and there were no federal laws or regulations requiring specific timing other than that they occur every two years. U.S. House of Representatives elections prior to 1880 are assigned an election identifier of '-999' in the CLEA Lower Chamber data.

CTR_N ***Country Name***

Names of countries. The following countries are included in the current CLEA release.

Afghanistan
Albania
Andorra
Angola
Anguilla
Antigua and Barbuda
Argentina
Armenia
Aruba
Australia
Austria
Azerbaijan
Bahamas
Bahrain
Bangladesh
Barbados
Belarus
Belgium
Belize
Benin
Bermuda
Bhutan
Bolivia
Bosnia & Herzegovina
Botswana
Brazil
British Virgin Islands
Bulgaria
Burkina Faso
Cambodia
Cameroon
Canada
Cape Verde
Cayman Islands
Chile
Colombia
Comoros
Costa Rica
Croatia
Curaçao
Cyprus
Czech Republic
Denmark
Djibouti
Dominica
Dominican Republic
Ecuador
El Salvador
Equatorial Guinea
Estonia
Ethiopia
Faroe Islands
Fiji
Finland
France
Gabon
Gambia
Georgia
Germany
Ghana
Gibraltar
Greece
Greenland
Grenada
Guatemala
Guinea
Guinea-Bissau
Guyana
Honduras
Hungary
Iceland
India

Indonesia
Iran
Iraq
Ireland
Israel
Italy
Ivory Coast
Jamaica
Japan
Kazakhstan
Kenya
Korea
Kosovo
Kuwait
Latvia
Lebanon
Lesotho
Liberia
Libya
Liechtenstein
Lithuania
Luxembourg
Macedonia
Madagascar
Malawi
Malaysia
Maldives
Malta
Marshall Islands
Mauritius
Mexico
Micronesia
Moldova
Mongolia
Montenegro
Montserrat
Mozambique
Myanmar
Namibia
Nauru
Nepal
Netherlands
New Zealand
Nicaragua
Niger
Nigeria
Norway
Pakistan
Palau
Paraguay
Peru
Philippines
Poland
Portugal
Puerto Rico
Romania
Russian Federation
Rwanda
Saint Kitts and Nevis
Saint Lucia
Samoa
San Marino
Senegal
Serbia
Seychelles
Sierra Leone
Singapore
Slovakia
Slovenia
Solomon Islands
Somaliland
South Africa
Spain
Sri Lanka
St. Vincent and the Grenadines
Sweden
Switzerland
Suriname
Taiwan
Tanzania
Thailand
Timor-Leste
Togo
Tonga
Trinidad and Tobago
Turkey
Turks and Caicos Islands
Uganda
Ukraine
United Arab Emirates
United Kingdom
United States
Uruguay
Vanuatu
Venezuela
Zambia
Zimbabwe

CTR Country Code

Country codes developed by the UN (http://unstats.un.org/unsd/methods/m49/m49.htm)

004	Afghanistan	028	Antigua and Barbuda
008	Albania	032	Argentina
020	Andorra	051	Armenia
024	Angola	533	Aruba
660	Anguilla	036	Australia

040	Austria
031	Azerbaijan
044	Bahamas
048	Bahrain
050	Bangladesh
052	Barbados
112	Belarus
056	Belgium
084	Belize
204	Benin
060	Bermuda
064	Bhutan
068	Bolivia
070	Bosnia and Herzegovina
072	Botswana
076	Brazil
092	British Virgin Islands
100	Bulgaria
854	Burkina Faso
116	Cambodia
120	Cameroon
124	Canada
132	Cape Verde
136	Cayman Islands
152	Chile
170	Colombia
174	Comoros
188	Costa Rica
191	Croatia
531	Curaçao
196	Cyprus
203	Czech Republic
208	Denmark
262	Djibouti
212	Dominica
214	Dominican Republic
218	Ecuador
222	El Salvador
226	Equatorial Guinea
233	Estonia
231	Ethiopia
234	Faroe Islands
242	Fiji
246	Finland
250	France
266	Gabon
270	Gambia
268	Georgia
276	Germany
288	Ghana
292	Gibraltar
300	Greece
304	Greenland
308	Grenada
320	Guatemala
324	Guinea
624	Guinea-Bissau
328	Guyana
340	Honduras
348	Hungary
352	Iceland
356	India
360	Indonesia
364	Iran
368	Iraq
372	Ireland
376	Israel
384	Ivory Coast
380	Italy
388	Jamaica
392	Japan
398	Kazakhstan
404	Kenya
410	Korea
414	Kuwait
428	Latvia
422	Lebanon
426	Lesotho
430	Liberia
434	Libya
438	Liechtenstein
440	Lithuania
442	Luxembourg
807	Macedonia
450	Madagascar
454	Malawi
458	Malaysia
462	Maldives

470	Malta
584	Marshall Islands
480	Mauritius
484	Mexico
583	Micronesia
498	Moldova
496	Mongolia
499	Montenegro
500	Montserrat
508	Mozambique
104	Myanmar
516	Namibia
520	Nauru
524	Nepal
528	Netherlands
554	New Zealand
558	Nicaragua
562	Niger
566	Nigeria
578	Norway
586	Pakistan
585	Palau
600	Paraguay
604	Peru
608	Philippines
616	Poland
620	Portugal
630	Puerto Rico
642	Romania
643	Russian Federation
646	Rwanda
659	Saint Kitts and Nevis
662	Saint Lucia
674	San Marino
882	Samoa
686	Senegal
688	Serbia
690	Seychelles
694	Sierra Leone
702	Singapore
703	Slovakia
705	Slovenia
090	Solomon Islands
710	South Africa
724	Spain
144	Sri Lanka
670	St. Vincent and the Grenadines
740	Suriname
752	Sweden
756	Switzerland
834	Tanzania
764	Thailand
636	Timor-Leste
768	Togo
776	Tonga
780	Trinidad and Tobago
792	Turkey
796	Turks and Caicos Islands
800	Uganda
804	Ukraine
784	United Arab Emirates
826	United Kingdom
840	United States
858	Uruguay
548	Vanuatu
862	Venezuela
894	Zambia
716	Zimbabwe
1001	Taiwan ***
1002	Kosovo
1003	Somaliland

*** Taiwan (the Republic of China), Kosovo, and Somaliland do not have a U.N. Standard Country Code. This code is assigned by CLEA.

YR ***Election Year***

Year of election.

MN ***Election Month***

Month of election, if available.

Note: To conduct calculations for elections spanning more than one month, the *mn* variable is set to “-990. Missing Data (information not available /category not applicable)”.

01. January
02. February
03. March
04. April
05. May
06. June
07. July
08. August
09. September
10. October
11. November
12. December

CST_N ***Constituency Name***

Name of geographical area that a particular elected representative or group of elected representatives represents.

CST ***Constituency Code***

A unique numeric code assigned to each constituency in each election in a country. In general, all constituencies in a country are sorted alphabetically, according to their names, and then assigned a constituency code. This code assignment is repeated in each election in the country. Thus, the same code may or may not belong to the same constituency across elections, depending upon whether redistricting occurs between elections. In the event of special districts for minority populations (e.g., the Maori districts in New Zealand prior to the electoral reform in 1996) or semi-autonomous regions (e.g., Greenland for Danish parliamentary elections) these districts receive the first numeric code following the last alphabetically sorted geographical district.

In a case where a country uses a multi-tier or mixed electoral system, the CLEA dataset uses the following coding scheme:

001-900. Lower-tier electoral districts (in multi-tier PR) or electoral districts where a majoritarian formula is used (in a mixed electoral system)
901-999. Upper-tier electoral districts (in multi-tier PR) or electoral districts where PR is used (in a mixed electoral system)

If a country uses a single-tier system, only constituency codes for lower-tier electoral districts are used.

CST_TOT ***Total Number of Constituencies***

The total number of constituencies in a given election.

PTY_N ***Party Name***

Name of a party or electoral alliance. If possible, the official name in the original language is used. If this name is not available, the transliterated or English-translated party name is used. For more information, refer to *Appendix II: Party Codes*.

In some cases where the original data sources we collected have small parties grouped under an "*Others*" category, "others" is used. In several countries, special kinds of party groupings are used in reported election results. For instance, categories such as "*miscellaneous right-wing*" and "*regionalists and separatists*" are used in France. For those special categories, their names are used for this variable and unique numeric codes are assigned to each such category (see "PTY" below for more information about these categories and also refer to *Appendix II: Party Codes*).

Independent candidates are handled in two different ways when election results are reported. For much of the data we have collected, all independent candidates are grouped under a single category. In such cases, "*Independents*" is used. However, when each and every individual independent candidate is identified and his/her votes received are reported separately in the election returns, "*Independent*" is used instead. Different numeric codes are assigned with these different methods. See "PTY" below for more information about numeric codes and Appendix II for the labels for independent candidates.
In a few countries, there are independent candidates who are affiliated with a party, but cannot officially stand under its label. As a result, they may be labeled in a manner that reflects both their

independent status and their party affiliation (e.g., "*Independent Labour*" or "*Independent Greens"*). In the CLEA dataset, we treat these candidates as independents in the coding, but keep their label under the PTY_N.

For a full list of political parties, see *Appendix II: Party Codes*.

PTY ***Party Code***

A unique numeric code is assigned to each party that runs a candidate in any given election. In general, political parties in a country are sorted alphabetically according to *PTY_N* and then assigned party codes. Parties have been matched across the lower chamber and upper chamber election results. Therefore, a party in a given country that runs in both chambers receives the same party code. To differentiate political parties and the aforementioned special and residual categories (see *PTY_N* above), the CLEA dataset uses the following coding scheme:

0001-3999. Political parties
- 3996. None of these candidates (in some countries, voters have the option to express disapproval for all the candidates on the ballot)
- 3997. Elected (for several early elections in Iceland and Sweden, the results for political parties are not available)
- 3998. No against for uncontested (in Denmark)
- 3999. Unknown

4000. "Others" (i.e., more than two small parties are grouped)
- 4001-. Special kinds of 'others' (see Appendix II for more information)
- 4998. Write-in
- 4999. Blank/Scattering

5001-5999. Electoral coalitions or alliances between political parties

6000. "Independents" (i.e., more than two independent candidates are grouped)
- 6001-. Independent 1, Independent 2, and so on (i.e., a single unaffiliated candidate), including special kinds of 'independents'.

For a full list of political parties and their codes in each country, see *Appendix II: Party Codes*.

Each party and electoral alliance is assigned a unique numeric code that remains consistent across elections. If a party changes its name, merges with other parties or splits into separate parties, a new numeric code is given to the party that emerges as a result of such changes.

Party codes for 'other' and 'independent' are assigned to parties or unaffiliated candidates in each election. This code assignment is repeated in each election in the country. Thus, the same code does not belong to the same minor party or independent candidate across elections.

NOTE: In India, there were more than 4,000 independent candidates in several general elections. In such cases, many independent candidates are assigned a five-digit party code.

NVVI ***National Valid Vote Indicator***

An indicator variable that takes on a value of "1" when the national effective number of parties measures (ENP_{NAT}, ENP_{AVG}, and ENP_{WGHT}) rely on the total sum of party votes (rather than the raw number of valid votes), and "0" otherwise. The CLEA data on the number of valid votes cast in a constituency do not always sum to the total number of party votes in the constituency. This is problematic in the calculation of the effective number of parties; we substitute summed party votes for valid votes as indicated by this variable.

CVVI ***Constituency Valid Vote Indicator***

An indicator variable that takes on a value of "1" when the constituency effective number of parties measure (ENP_{CST}) relies on the total sum of party votes (rather than the raw number of valid votes), and "0" otherwise. The CLEA data on the number of valid votes cast in a constituency do not always sum to the total number of party votes in the constituency. This is problematic in the calculation of the effective number of parties; we substitute summed party votes for valid votes as indicated by this variable.

ENP_CST ***Effective Number of Parties (Constituency)***

The effective number of (electoral) parties in a country's party system at the constituency level for the specified election year. This is calculated at the constituency level following Laakso and Taagepera's (1979) specification:

$$ENEP_{cst} = \frac{1}{\sum_1^n p_i^2}$$

Here n represents the number of parties in a district and p_i represents the party's share of the constituency vote. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Laakso, M., & Taagepera, R. (1979). "Effective Number of Parties: A Measure with Application to West Europe." *Comparative Political Studies* 12(1): 3-27.

ENP_NAT ***Effective Number of Parties (National)***

The effective number of (electoral) parties in a country's party system at the national level for the specified election year. This is calculated at the national level following Laakso and Taagepera's (1979) specification:

$$ENP_{nat} = \frac{1}{\sum_1^n {p_i}^2}$$

Here *n* represents the number of parties in a country and p_i represents the party's share of the national vote. Values coded as "-990" indicate missing data. Due to the large number of independent candidates, we do not provide these data for Korea in 1950 or the United States before 1834.

For more information on the interpretation and calculation of this variable, see: Laakso, M., & Taagepera, R. (1979). "Effective Number of Parties: A Measure with Application to West Europe." *Comparative Political Studies* 12(1): 3-27.

ENP_AVG ***Effective Number of Parties (Average)***

The average effective number of (electoral) parties in a country's party system at the national level for the specified election year. The number of parties is calculated at the constituency level, following Laakso and Taagepera's (1979) specification, and then averaged at the national level.

$$ENP_{avg} = \frac{\sum_1^d ENEP_{cst}}{d}$$

Here *d* represents the number of districts in a country. Values coded as "-990" indicate missing data. Due to the large number of independent candidates, we do not provide these data for Korea in 1950 or the United States before 1834.

For more information on the interpretation and calculation of this variable, see: Laakso, M., & Taagepera, R. (1979). "Effective Number of Parties: A Measure with Application to West Europe." *Comparative Political Studies* 12(1): 3-27.

ENP_WGHT Effective Number of Parties (Weighted)

A weighted average of the effective number of (electoral) parties in a country's party system at the national level for the specified election year. The number of parties is calculated at the constituency level, following Laakso and Taagepera's (1979) specification, and then averaged at the national level, weighting each constituency according to its share of voters out of the national total (Moenius & Kasuya, 2004). Formally, this is:

$$ENP_{wght} = \sum_{1}^{d} ENP_{cst} \frac{vot_{cst}}{vot_{nat}}$$

Here, *d* represents the number of districts in the country, *vot_{cst}* indicates the number of votes cast in the constituency and *vot_{nat}* indicates the number of votes cast in the country. Values coded as "-990" indicate missing data. Due to the large number of independent candidates, we do not provide these data for Korea in 1950 or the United States before 1834.

For more information on the interpretation and calculation of this variable, see: Laakso, M., & Taagepera, R. (1979). "Effective Number of Parties: A Measure with Application to West Europe." *Comparative Political Studies* 12(1): 3-27.

INFLATION1 Cox Inflation Score

A measure of party linkage across a country's electoral constituencies that builds on the economic principle of inflation. Here inflation refers to the discrepancy that occurs in party linkage as parties are aggregated from the constituency-level to the national-level party system. This measure was developed by Cox (1999).

$$inflation1 = \frac{ENP_{nat} - ENP_{avg}}{ENP_{nat}}$$

Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Cox, G. (1999). "Electoral Rules and Electoral Coordination." *Annual Review of Political Science* 2(1): 145-161.

INFLATION2 Moenius and Kasuya Inflation Score

A measure of party linkage across a country's electoral constituencies that builds on the economic principle of inflation. Here inflation refers to the discrepancy that occurs in party linkage as parties are aggregated from the constituency-level to the national-level party system. This measure corresponds with Moenius and Kasuya's (2004) variable *I*, and is distinct from Inflation 1 because it has a different denominator.

$$inflation2 = \frac{ENP_{nat} - ENP_{avg}}{ENP_{avg}}$$

Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Moenius, J., & Kasuya, Y. (2004). "Measuring Party Linkage Across Districts." *Party Politics* 10(5): 543-564.

INFLATION3 Moenius and Kasuya Weighted Inflation Score

A measure of party linkage across a country's electoral constituencies that builds on the economic principle of inflation. Here inflation refers to the discrepancy that occurs in party linkage as parties are aggregated from the constituency-level to the national-level party system. This measure corresponds with Moenius and Kasuya's (2004) weighted inflation variable I_w, which accounts for variation in district size.

$$inflation3 = \frac{ENP_{nat} - ENP_{wght}}{ENP_{wght}}$$

Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Moenius, J., & Kasuya, Y. (2004). "Measuring Party Linkage Across Districts." *Party Politics* 10(5): 543-564.

INFLATION4 Kasuya and Moenius Inflation and Dispersion Score

A measure of party linkage across a country's electoral constituencies that builds on the economic principle of inflation. Here inflation refers to the discrepancy that occurs in party linkage as parties are aggregated from the constituency-level to the national-level party system. This measure also account for a dispersion, which "refers to the extent to which the contribution of each district's party system to the size of the national-level party system varies across districts" (Kasuya & Moenius, 2008). The formula is:

$$inflation4 = inflation3^{\alpha} D^{1-\alpha}$$

where *inflation3* captures the inflation dimension and *D* captures the dispersion dimension. The formula for dispersion is:

$$D = CV(inflation5)^{\gamma} k(inflation5)^{1-\gamma}$$

Here, *CV* (the coefficient of variation) and *k* (the district's kurtosis) both refer to measures of the local inflation measure (*inflation5*---see next variable). Note that the alpha and gamma parameters and are user-defined. Gamma weights the districts according to their relationship to the aggregate party system inflation rate and alpha is the weight assigned to inflation versus dispersion. For the purposes of this analysis (and following Kasuya and Moenius's basic recommendation) inflation and dispersion are equally weighted (alpha = .5), as are the districts (gamma = 0.5). Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Kasuya, Y., & Moenius, J. (2008). "The Nationalization of Party Systems: Conceptual Issues and Alternative District-focused Measures." *Electoral Studies* 27(1): 126-135.

INFLATION5 Moenius and Kasuya Local Inflation Score

A measure of party linkage that measures the difference between the party system at the local constituency level and the national level. This measure is similar to Moenius and Kasuya's (2004) weighted inflation variable I_i:

$$inflation5 = \frac{ENP_{nat} - ENP_{cst}}{ENP_{cst}}$$

Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Moenius, J., & Kasuya, Y. (2004). "Measuring Party Linkage Across Districts." *Party Politics* 10(5): 543-564

PNS ***Party Nationalization Score***

A measure of the nationalization of a party, based on the Gini coefficient of inequality (G_i) in vote shares across constituencies (Jones & Mainwaring, 2003). To calculate, we take the inverse of the Gini coefficient:

$$PNS = 1 - G_i$$

The Stata program *INEQDEC0* (Jenkins, 1999) was used to create the Gini coefficient. We do not calculate the Party Nationalization Score for parties that received less than a five percent share of the national vote in an election. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Jones, M. P., & Mainwaring, S. (2003). "The Nationalization of Parties and Party Systems." *Party Politics* 9(2): 139-166.

PNS_S ***Standardized Party Nationalization Score***

A measure of the nationalization of a party, based on the Gini coefficient of inequality in vote shares across constituencies (Jones & Mainwaring, 2003). For this measure, the Party Nationalization Score (*PNS*) is standardized to account for variation in the number of territorial units across countries (Bochsler, 2006). The formula is:

$$PNS_s = (PNS)^{\frac{1}{\log(d)}}$$

where *d* is the number of districts in a country. The Stata program *INEQDEC0* (Jenkins, 1999) was used to create the Gini coefficient. We do not calculate the Standardized Party Nationalization Score for parties that received less than a five percent share of the national vote in an election. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Bochsler, D. (2006). "The Nationalization of Political Parties: A Triangle Model, Applied on the Central and Eastern European Countries." *CEU Political Science Journal* 1(4): 6-32.

PNS_W ***Weighted Party Nationalization Score***

A measure of the nationalization of a party, based on the Gini coefficient of inequality in vote shares across constituencies (Bochsler, 2010). The formula is:

$$PNS_w = 2 * \frac{\sum_1^d (vot_{cst} * (\sum_1^i pty_j - \frac{pty_i}{2}))}{\sum_1^d vot_{cst} * \sum_1^d pty_i}$$

The number of votes (absolute) in the constituency is vot_{cst} and pty_i is the number of votes for party *i* (absolute) in the constituency. Here pty_j is the cumulative number of votes for party *i* in the election, with districts sorted (increasing) according to the party's vote share in the constituency. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Bochsler, D. (2010). "Measuring Party Nationalisation: A New Gini-based Indicator that Corrects for the Number of Units." *Electoral Studies* 29(1): 155-168.

PNS_SW ***Standardized and Weighted Party Nationalization Score***

A measure of the nationalization of a party that standardizes for the number of territorial units in a country and also weights for the size of the territorial units (Bochsler, 2010). This measure builds upon the weighted party nationalization score (*PNS_W*) by adding an additional correction for the unequal sizes of units across countries. The formula is:

$$PNS_{sw} = (PNS_w)^{\frac{1}{log(E)}}$$

where, the variable *E* is a constant calculated at the national level as follows:

$$E = \frac{(\sum_1^d vot_{cst})^2}{\sum_1^d vot_{cst}{}^2}$$

where vot_{cst} is the raw number of votes cast in constituency *i*. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Bochsler, D. (2010). "Measuring Party Nationalisation: A New Gini-based Indicator that Corrects for the Number of Units." *Electoral Studies* 29(1): 155-168.

PSNS ***Party System Nationalization Score***

A summary expression of the level of nationalization of a party system (Jones & Mainwaring, 2003). It is calculated according to the following formula:

$$PSNS = \sum_{1}^{n} PNS_i * p_i$$

where p_i is the party's share of the national vote. Parties that received less than a five percent share of the vote at the national level are excluded from this analysis. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Jones, M. P., & Mainwaring, S. (2003). "The Nationalization of Parties and Party Systems." *Party Politics* 9(2): 139-166.

PSNS_S ***Standardized Party System Nationalization Score***

A summary expression of the level of nationalization of a party system that is standardized to account for variation in the number of territorial units across countries (Bochsler, 2006).

$$PSNS_s = \sum_{1}^{n} PNS_{S,i} * p_i$$

where p_i is the party's share of the national vote. Parties that received less than a five percent share of the vote at the national level are excluded from this analysis. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Bochsler, D. (2006). "The Nationalization of Political Parties: A Triangle Model, Applied on the Central and Eastern European Countries." *CEU Political Science Journal* 1(4): 6-32.

PSNS_W ***Weighted Party System Nationalization Score***

A summary expression of the level of nationalization of a party system that is weighted to account for variation in the size of the territorial units across countries (Bochsler, 2010).

$$PSNS_W = \sum_1^n PNS_{W,i} * p_i$$

where p_i is the party's share of the national vote. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Bochsler, D. (2010). "Measuring Party Nationalisation: A New Gini-based Indicator that Corrects for the Number of Units." *Electoral Studies* 29(1): 155-168.

PSNS_SW ***Standardized and Weighted Party System Nationalization Score***

A summary expression of the level of the nationalization of a party system that standardizes for the number of territorial units and also weights for the size of the territorial units (Bochsler, 2010).

$$PSNS_{SW} = \sum_1^n PNS_{SW,i} * p_i$$

where p_i is the party's share of the national vote. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Bochsler, D. (2010). "Measuring Party Nationalisation: A New Gini-based Indicator that Corrects for the Number of Units." *Electoral Studies* 29(1): 155-168.

LOCAL_E ***Local Entrant Measure***

A measure of party system nationalization based on parties' entry decisions (Lago & Montero, 2010). This measure accounts for the differential size of territorial units, the variance in party supply across districts, and party votes shares. It varies between 0 and 1. The formula for this measure is:

$$local_E = \sum_1^n v_i * s_i$$

Here, v_i is the proportion of votes obtained by party *i* at the national level and s_i is the proportion of seats of the national total to be allocated in those districts where party *i* entered the race. Values coded as "-990" indicate missing data.

For more information on the interpretation and calculation of this variable, see: Lago, I., & Montero, J. R. (2010). *The Nationalisation of Party Systems Revisited: A New Measure Based on Parties' Entry Decisions*, *Electoral Results*, *and District Magnitude.* Paper presented at the Annual Meeting of the Canadian Political Science Association, Concordia University, Montreal, PQ.

REFERENCES

Bochsler, D. (2006). The Nationalization of Political Parties: A Triangle Model, Applied on the Central and Eastern European Countries. *CEU Political Science Journal, 1*(4), 6 - 32.

Bochsler, D. (2010). Measuring Party Nationalisation: A New Gini-based Indicator that Corrects for the Number of Units. *Electoral Studies, 29*(1), 155-168.

Cox, G. (1999). Electoral Rules and Electoral Coordination. *Annual Review of Political Science, 2*(1), 145-161.

Jenkins, S. P. (1999). INEQDEC0: Stata Module to Calculate Inequality Indices with Decomposition by Subgroup: Boston College Department of Economics.

Jones, M. P., & Mainwaring, S. (2003). The Nationalization of Parties and Party Systems. *Party Politics, 9*(2), 139-166.

Kasuya, Y., & Moenius, J. (2008). The Nationalization of Party Systems: Conceptual Issues and Alternative District-focused Measures. *Electoral Studies*, *27*(1), 126-135.

Laakso, M., & Taagepera, R. (1979). Effective Number of Parties: A Measure with Application to West Europe. *Comparative political studies, 12*(1), 3-27.

Lago, I., & Montero, J. R. (2010). *The Nationalisation of Party Systems Revisited: A New Measure Based on Parties' Entry Decisions, Electoral Results, and District Magnitude.* Paper presented at the Annual Meeting of the Canadian Political Science Association, Concordia University, Montreal, PQ.

Moenius, J., & Kasuya, Y. (2004). Measuring Party Linkage Across Districts. *Party Politics, 10*(5), 543-564.

APPENDIX: COMPUTER CODE FOR ENP CALCULATIONS

The CLEA ENP datasets are created using the following code in *Stata*. The equivalent code is also provided for *R* on page 41.

The three ENP datasets are created by first calculating the Gini index for parties, followed by the party level of aggregation, national level of aggregation, and lastly the constituency level of aggregation.

Stata Code for Calculations

1. Gini coefficient calculation

```
*******************
Gini Index for Parties
*******************
/*This file creates the Gini index for CLEA data, indicating the inequality in vote shares received by parties in districts. This information is used for calculating Jones and Mainwaring's (2003) Party Nationalization Score (PNS).*/


*To create the Gini index, we use the command INEQDEC0.  The code below installs the command.
ssc inst ineqdec0, replace

*Create Gini using CLEA data
use "clea_20190617.dta", clear

*Eliminate single constituency elections
use "clea_20190617.dta", clear
bysort id ctr yr mn (cst): gen single_constituency =  cst[1] == 1 & cst[_N] == 1
drop if single_constituency==1
save "CLEA.no.SC.dta", replace

*Move the data from candidate-cst level to party-cst level
collapse (first) ctr_n vv1 pv1 if vv1 >0 & pv1 > 0, by (id ctr yr mn pty cst)

*Drop observations where the party wins less than five percent of the national party vote
egen nat_vv1 = sum(pv1), by(id ctr yr mn)
egen pty_nat_vv1 = sum(pv1), by(id ctr yr mn pty)
gen nat_pvs = pty_nat_vv1/ nat_vv1
drop if nat_pvs < .05
drop nat_vv1 pty_nat_vv1 nat_pvs
save "CLEA.list.dta", replace
```

```
*Create a shell dataset that includes all the parties/constituencies
drop pty ctr_n vv1 pv1
duplicates drop
save "CLEA.unique.constituencies.dta", replace
use "CLEA.list.dta", clear
drop cst ctr_n vv1 pv1
duplicates drop
joinby id ctr yr mn using "CLEA.unique.constituencies.dta", unmatched(both)
drop _merge
joinby id ctr yr mn cst pty using "CLEA.no.SC.dta", unmatched(master)
drop _merge release rg sub cst_n mag pty_n can pev1 vot1 ivv1 to1 cv1 cvs1 cvs1 pvs1
save "CLEA.gini.dta", replace

*Create party vote share (constituency) variable
gen pvs = pv1/vv1
replace pvs = 0 if pvs ==.

*gen gini placeholder
gen gini = .

*Now we use the INEQDECO command to create the Gini index using the CLEA data
egen group = group(id ctr yr mn pty) if gini == .
save, replace

su group, meanonly
forval i = 1/`r(max)' {
	ineqdec0 pvs if group == `i'
	replace gini = r(gini) if group == `i'
}

save "CLEA.gini.dta", replace

**cleaning**
drop ctr_n vv1 pv1 pvs
sort id ctr yr mn cst
save "CLEA.gini.dta", replace
```

2. Party level of aggregation

```
*******************************************
*Party Nationalization Measures -Party Level Dataset
*******************************************
```

```
*Create Party System Nationalization Score (PSNS) and Standardized Party Nationalization
Score (PSNS_s)
use "clea_20190617.dta", clear
keep ctr ctr_n yr mn pty_n pty
sort id ctr yr mn pty
duplicates drop
save "clea_enp_20190617_party.level.dta", replace

use "clea_20190617.dta", clear
*Move data from candidate-cst level to pty-cst level
collapse (first) ctr_n vv1 pv1 if vv1 >0 & pv1 > 0, by (id ctr yr mn pty cst)

*Call in the gini data
joinby id ctr yr mn pty cst using "CLEA.gini.dta", unmatched(using)
drop _merge

*Gen PNS
gen PNS = 1 - gini

*Standardized PNS (PNS_s)
by id ctr yr mn cst , sort: gen districts = _n == 1
egen n_districts = sum(districts), by(id ctr yr mn)
gen power = 1/(log(n_districts))
gen PNS_s = PNS^power
drop districts n_districts power

keep id ctr yr mn pty PNS PNS_s
duplicates drop

joinby id ctr yr mn pty using "clea_enp_20190617_ party.level.dta", unmatched(using)
drop _merge

save "clea_enp_20190617_party.level.dta", replace

*Create Bochsler's PNS_w and PNS_sw (2010)
use "clea_20190617.dta", clear

collapse (first) pv1 vv1 ,by (id ctr yr mn pty cst)

*Create the denominator
drop if pv1 < 0
drop if vv1 < 0
egen nat_vv1 = sum(pv1), by(id ctr yr mn)
egen pty_vv1 = sum(pv1), by(id ctr yr mn pty)
gen denominator = nat_vv1 * pty_vv1
```

```
*Create the numerator
rename vv1 cst_vv1
gen vote_share = pv1/cst_vv1
sort id ctr yr mn pty vote_share
by id ctr yr mn pty: gen p_j =sum(pv1)
gen inside = cst_vv1 * (p_j - (pv1/2))
egen numerator = sum(inside), by(id ctr yr mn pty)
gen PNS_w = (2 * numerator)/ denominator

*FIX PNS_w for countries where sum(pv1) != vv1 & -990 values lead to inaccuracies
egen cst_vv1_new = sum(pv1), by(id ctr yr mn cst)
gen vote_share_new = pv1/cst_vv1_new
sort id ctr yr mn pty vote_share_new
by id ctr yr mn pty: gen p_j_new =sum(pv1)
gen inside_new = cst_vv1_new * (p_j_new - (pv1/2))
egen numerator_new = sum(inside_new), by(id ctr yr mn pty)
gen PNS_w_new = (2 * numerator_new)/ denominator

gen alt_vv1 = 1 if PNS_w >= 1
replace PNS_w = PNS_w_new if PNS_w >= 1

replace PNS_w = -990 if pty == 3999 | pty >= 6000

*Setting single constituencies to missing (single constituencies cause inaccurate PNS_w scores)
bysort id ctr yr mn (cst): gen single_constituency =  cst[1] == 1 & cst[_N] == 1
replace PNS_w = -990 if single_constituency==1
drop single_constituency

*Also PNS_w is messed up for one cst in US in 1959, so correct
replace PNS_w = -990 if ctr == 840 & yr == 1959 & cst ==435
replace PNS_w = -990 if PNS_w ==.

replace alt_vv1 = 0 if PNS_w == -990
keep ctr  yr mn cst pty cst_vv1 nat_vv1 PNS_w alt_vv1

*Create the PSNS_sw measure from the PNS_w measure
gen top = nat_vv1 * nat_vv1
gen square = cst_vv1 * cst_vv1
gen helper = .
by id ctr yr mn cst, sort: gen pid = _n
replace helper = square if pid == 1
egen bottom = sum(helper), by(id ctr yr mn)
gen power_E = top/ bottom
gen PNS_sw = (PNS_w)^(1/(log10(power_E)))
replace PNS_sw = -990 if PNS_w == -990
replace PNS_sw = -990 if PNS_sw >1 | PNS_sw ==.
keep id ctr yr mn pty PNS_w PNS_sw
duplicates drop
sort id ctr yr mn pty

joinby id ctr yr mn pty using " clea_enp_20190617_party.level.dta", unmatched (using)
drop _merge

label var ctr_n "Country name"
label var yr "Election year"
label var mn "Election month"
label var pty_n "Party name"
label var pty "Party code"
label var ctr "CLEA country code"
label var PNS "Jones & Mainwaring Party Nationalization Score"
label var PNS_s "Bochsler standardized Party Nationalization Score"
label var PNS_w "Bochsler weighted Party Nationalization Score"
label var PNS_sw "Bochsler standarized & weighted Party Nationalization Score"

*Cleaning
replace PNS = -990 if PNS ==.
replace PNS_s = -990 if PNS_s ==.
replace PNS_w = -990 if PNS_w ==.
replace PNS_sw = -990 if PNS_sw ==.

compress

replace PNS = -990 if PNS == 1
replace PNS = -990 if pty >= 3999 & pty < 5000
replace PNS_s = -990 if  pty >= 3999 & pty < 5000
replace PNS_w = -990 if pty >= 3999 & pty < 5000
replace PNS_sw = -990 if pty >= 3999 & pty < 5000

replace pty = 3999 if pty == -990

replace PNS= -990  if pty >= 3996  & pty < 5000
replace PNS_s= -990  if pty >= 3996  & pty < 5000
replace PNS_w= -990  if pty >= 3996  & pty < 5000
replace PNS_sw= -990  if pty >= 3996  & pty < 5000

order ctr_n id ctr yr mn pty_n pty PNS PNS_s PNS_w PNS_sw, first
sort ctr_n id ctr yr mn pty_n pty PNS PNS_s PNS_w PNS_sw

*drop all duplicates
duplicates drop id ctr yr mn pty, force
```

```
save " clea_enp_20190617_party.level.dta", replace
```

3. National level of aggregation

```
************************************************
*Party Nationalization Measures - National Level Dataset
************************************************

use "clea_20190617.dta", clear
keep  ctr_n id ctr yr mn

duplicates drop
save "clea_enp_20190617_national.level.dta", replace

*Create ENEP national
use "clea_20190617.dta", clear

*Get rid of candidate level by moving data to pty-cst level
collapse (first) vv1 pv1 if vv1 >0 & pv1 > 0, by (id ctr yr mn pty cst)
sort id ctr yr mn cst pty

*Move to pty-ctr level
collapse(sum) pv1, by (id ctr yr mn pty)
egen nat_vv1 = sum(pv1), by(id ctr yr mn)
gen party_prop_nat2 = (pv1/nat_vv1) * (pv1/nat_vv1)
egen denom = sum(party_prop_nat2), by(id ctr yr mn)
gen ENEP_nat = 1/denom
drop  pty pv1 nat_vv1 party_prop_nat2 denom
duplicates drop
replace ENEP_nat = . if yr < 1834 & ctr == 840
replace ENEP_nat = . if yr == 1950 & ctr == 410
joinby id ctr yr mn using "national.level.dta", unmatched (using)
drop _merge
save "clea_enp_20190617_national.level.dta", replace
clear

*Create ENEP national average & ENEP national weighted average
use "clea_20190617.dta", clear
keep id ctr yr mn cst pty vv1 pv1
duplicates drop
collapse (sum) pv1 if pv1 >=0 & vv1 > 0, by (id ctr yr mn pty cst)
egen new_vv1 = sum(pv1), by (id ctr yr mn cst)
duplicates drop
```

```
save "clea_enp_20190617_national.level.pv1.dta", replace

use "clea_20190617.dta", clear
keep id ctr yr mn cst pty vv1 pv1
duplicates drop
collapse (first) vv1 if pv1 >=0 & vv1 > 0, by (id ctr yr mn pty cst)
joinby id ctr yr mn pty cst using "national.level.pv1.dta", unmatched(using)
drop _merge

gen indicator = 1 if new_vv1 != vv1
gen share2 = (pv1/new_vv1) * (pv1/new_vv1)
egen denom = sum(share2), by (id ctr yr mn cst)
gen ENEP_cst = 1/denom
drop  pv1 vv1 share2 denom pty
duplicates drop
egen nat_vv1 = sum(new_vv1), by(id ctr yr mn)
gen cst_wght =  new_vv1/ nat_vv1
gen weighted =  cst_wght *  ENEP_cst
egen ENEP_wght = sum(weighted), by(id ctr yr mn)
egen ENEP_avg = mean(ENEP_cst), by(id ctr yr mn)
drop  new_vv1 ENEP_cst cst nat_vv1 cst_wght weighted
duplicates drop
collapse (first) ENEP_avg ENEP_wght (max) indicator, by (id ctr yr mn)
joinby id ctr yr mn using "national.level.dta", unmatched(using)
drop _merge
order ctr ctr_n yr mn ENEP_nat ENEP_avg ENEP_wght, first
replace ENEP_nat = . if yr < 1834 & ctr == 840
replace ENEP_nat = . if yr == 1950 & ctr == 410
replace ENEP_avg = . if yr < 1834 & ctr == 840
replace ENEP_avg = . if yr == 1950 & ctr == 410
replace ENEP_wght = . if yr < 1834 & ctr == 840
replace ENEP_wght = . if yr == 1950 & ctr == 410

duplicates drop
save "clea_enp_20190617_national.level.dta", replace

*Create Cox score (Cox)
gen Cox =  (ENEP_nat - ENEP_avg)/  ENEP_nat

*Create Moenius & Kasuya Inflation score (MK_I)
gen MK_I = ( ENEP_nat -  ENEP_avg)/  ENEP_avg

*Create Moenius & Kasuya Weighted Inflation score (MK_I_w)
gen MK_I_w = ( ENEP_nat -   ENEP_wght)/   ENEP_wght

duplicates drop
```

```
save "clea_enp_20190617_national.level.dta", replace

*Create composite party system nationalization measure (MK_N)

*Note: first define the values of the scalars: gamma, alpha, and beta
/*According to Kasuya and Moenius, these scalars are arbitrary and depend upon the research question at hand.
To give equal weight to the inflation and dispersion measures, define alpha = 0.5, gamma = 0.5, and beta = 0.25.*/

scalar alpha = .5
scalar beta = .25
scalar gamma = .5

use "clea_20190617.dta", clear
collapse (first) pv1 vv1 if pv1 >=0 & vv1 > 0, by (id ctr yr mn pty cst)
egen new_vv1 = sum(pv1), by (id ctr yr mn cst)
gen share2 = (pv1/new_vv1) * (pv1/new_vv1)
egen denom = sum(share2), by (id ctr yr mn cst)
gen ENEP_cst = 1/denom

*Create I_i
joinby id ctr yr mn using "national.level.dta", unmatched (using)
drop  _merge
duplicates drop
gen I_i = (( ENEP_nat-  ENEP_cst)/ ENEP_cst) * 100

*Create W_tilde
egen n_districts = max(cst), by(id ctr yr mn)
egen nat_vote = sum(new_vv1), by(id ctr yr mn)

gen cst_vote_proportion = new_vv1/nat_vote
gen product =  ENEP_cst *  cst_vote_proportion
egen sum_cst = sum (product), by (id ctr yr mn)
gen denominator = n_districts * sum
gen W_tilde = ENEP_cst/ denominator
drop  denominator

*Create I_w
gen I_w = ((ENEP_nat - sum_cst)/sum_cst)*100

*Create coefficient of variation
gen numerator = (I_i -  I_w)^2 * W_tilde
egen sum_numerator = sum(numerator), by(id ctr yr mn)
gen coeff_var_I_i = sqrt(sum_numerator)/  I_w
```

```
*Create kurtosis measure
gen numerator2 = ((I_i - I_w)^4) * W_tilde
egen sum_numerator2 = sum(numerator2), by (id ctr yr mn)

gen denominator2 = ((I_i - I_w)^2) * W_tilde
egen sum_denominator2 = sum(denominator2), by (id ctr yr mn)
gen sq_sum_denominator2 = (sum_denominator2)^2

gen kurtosis_I_i = sum_numerator2/sq_sum_denominator2

*Create dispersion measure
gen D =  (coeff_var_I_i)^gamma * (kurtosis_I_i)^(1 - gamma)

*Create composite measure (Inflation and Dispersion)
gen MK_N = ((I_w)^alpha) * (D^(1 - alpha))
gen MK_N_two = ((I_w)^alpha) *  ((coeff_var_I_i)^beta) * ((kurtosis_I_i)^(1 - alpha - beta))

collapse (first) MK_N MK_N_two, by (id ctr yr mn)

joinby id ctr yr mn using "national.level.dta", unmatched(both)
drop _merge
save "clea_enp_20190617_national.level.dta", replace

*Create Party System Nationalization Score (PSNS) measures
use "clea_20190617.dta", clear

*Move data from candidate-cst level to pty-cst level
collapse (first) ctr_n vv1 pv1 if vv1 >0 & pv1 > 0, by (id ctr yr mn pty cst)

egen nat_vv1 = sum(pv1), by(id ctr yr mn)
egen pty_vv1 = sum(pv1), by(id ctr yr mn pty)
keep id ctr yr mn pty pty_vv1 nat_vv1
duplicates drop
joinby id ctr yr mn pty using " clea_enp_20190617_party.level.dta", unmatched(using)
order ctr ctr_n yr mn pty pty_n
drop _merge

*Gen Party System Nationalization Score (PSNS)
replace PNS = . if PNS == -990
replace PNS_s = . if PNS_s == -990
replace PNS_w = . if PNS_w == -990
replace PNS_sw = . if PNS_sw == -990

gen weight = pty_vv1/nat_vv1

egen PSNS = sum(PNS * weight), by(id ctr yr mn)
egen PSNS_s = sum(PNS_s * weight), by(id ctr yr mn)
egen PSNS_w = sum(PNS_w * weight), by(id ctr yr mn)
egen PSNS_sw = sum(PNS_sw *weight), by(id ctr yr mn)

keep ctr ctr_n yr mn PSNS PSNS_s PSNS_w PSNS_sw
duplicates drop

replace PSNS = -990 if PSNS == 0
replace PSNS_s = -990 if PSNS_s == 0
replace PSNS_w = -990 if PSNS_w ==0
replace PSNS_sw = -990 if PSNS_sw == 0

replace PSNS = -990 if PSNS > 1
replace PSNS_s = -990 if PSNS_s > 1
replace PSNS_sw = -990 if PSNS_sw > 1
replace PSNS_w = -990 if PSNS_w > 1

joinby id ctr yr mn using "national.level.dta", unmatched (both)
drop _merge
save "clea_enp_20190617_national.level.dta", replace


*Create Local Entrant Measure E
use "clea_20190617.dta", clear
collapse (first) ctr_n vv1 pv1 (sum)seat cv1 if seat >= 0 & vv1 >0 & pv1 > 0, by (id ctr yr mn pty cst)
egen nat_vote = sum(pv1) if pv1>= 0, by (id ctr yr mn)
egen seat_cst = sum(seat), by (id ctr yr mn cst)
collapse (first) ctr_n vv1 pv1 nat_vote seat_cst (sum) cv1 seat if vv1  >0 & pv1 > 0 & seat >= 0, by (id ctr yr mn pty cst)
sort id ctr yr mn pty
gen party_vote_proportion =  pv1/ nat_vote
egen seat_total = sum(seat), by (id ctr yr mn)
egen seat_contest = sum(seat_cst), by (id ctr yr mn pty)
gen seat_proportion = seat_contest/seat_total
gen local_E =  party_vote_proportion * seat_proportion
collapse (sum) local_E, by(id ctr yr mn)
joinby id ctr yr mn using "national.level.dta", unmatched(both)
drop _merge
drop  MK_N_two

label var ctr_n "Country name"
label var ctr "CLEA country code"
label var yr "Election year"
label var mn "Election month"
label var ENEP_nat "Effective Number of Electoral Parties nationally"
label var ENEP_avg "Average Effective Number of Electoral Parties nationally"
label var  ENEP_wght "Weighted average of Effective Number of Electoral Parties nationally"
label var Cox "Cox inflation score"
label var MK_I "Moenius & Kasuya inflation score"
label var MK_I_w "Moenius & Kasuya weighted inflation score"
label var MK_N "Moenius & Kasuya composite measure"
label var PSNS "Jones & Mainwaring Party System Nationalization Score"
label var PSNS_s "Bochsler standardized Party System Nationalization Score"
label var PSNS_w "Bochsler weighted Party System Nationalization Score"
label var PSNS_sw "Bochsler standarized & weighted Party System Nationalization Score"
label var local_E "Local entrant measure"
label var indicator "National Valid Vote Indicator"
compress

save "clea_enp_20190617_national.level.dta", replace

*Cleaning
rename Cox inflation1
rename MK_I inflation2
rename MK_I_w inflation3
rename MK_N inflation4
rename indicator nvvi

replace nvvi = 0 if nvvi ==.
replace ENEP_nat = -990 if ENEP_nat ==.
replace ENEP_avg = -990 if ENEP_avg ==.
replace ENEP_wght = -990 if ENEP_wght ==.
replace inflation1 = -990 if inflation1 ==.
replace inflation2 = -990 if inflation2 ==.
replace inflation3 = -990 if inflation3 ==.
replace inflation4 = -990 if inflation4 ==.
replace PSNS = -990 if PSNS ==.
replace PSNS_s = -990 if PSNS_s == .
replace PSNS_w = -990 if PSNS_w ==.
replace PSNS_sw = -990 if PSNS_sw ==.
replace local_E = -990 if local_E ==.

rename ENEP_nat ENP_nat
rename ENEP_avg ENP_avg
rename ENEP_wght ENP_wght

replace ENP_nat = -990 if ctr == 144 & yr == 1947 | ctr == 144 & yr == 1952 | ctr == 144 & yr
== 1956 | ctr == 144 & yr == 1960 | ctr == 144 & yr == 1965 | ctr == 144 & yr == 1970 | ctr ==
144 & yr == 1977
```

```
replace ENP_avg = -990 if ctr == 144 & yr == 1947 | ctr == 144 & yr == 1952 | ctr == 144 & yr == 1956 | ctr == 144 & yr == 1960 | ctr == 144 & yr == 1965 | ctr == 144 & yr == 1970 | ctr == 144 & yr == 1977

replace ENP_wght = -990 if ctr == 144 & yr == 1947 | ctr == 144 & yr == 1952 | ctr == 144 & yr == 1956 | ctr == 144 & yr == 1960 | ctr == 144 & yr == 1965 | ctr == 144 & yr == 1970 | ctr == 144 & yr == 1977

replace inflation1 = -990 if inflation1 == 0
replace inflation2 = -990 if inflation2 == 0
replace inflation3 = -990 if inflation3 == 0
replace inflation4 = -990 if inflation4 == 0

order ctr_n id ctr yr mn nvvi ENP_nat ENP_avg ENP_wght inflation1 inflation2 inflation3 inflation4 PSNS PSNS_s PSNS_w PSNS_sw local_E, first
sort ctr_n id ctr yr mn nvvi ENP_nat ENP_avg ENP_wght inflation1 inflation2 inflation3 inflation4 PSNS PSNS_s PSNS_w PSNS_sw local_E

*drop all duplicates
duplicates drop id ctr yr mn, force
save "clea_enp_20190617_national.level.dta", replace
```

4. Constituency level of aggregation

```
***************************************************
*Party Nationalization Measures - Constituency Level Dataset
***************************************************

/*Note: this part of the file uses CLEA data to create several alternate measures of party nationalization identified in the literature */

*Create frame dataset
use "clea_20190617.dta", clear
keep ctr_n id ctr yr mn cst_n cst

duplicates drop
save "clea_enp_20190617_cst.level.dta", replace

*Create ENP constituency
use "clea_20190617.dta", clear
collapse (first) pv1 vv1 if pv1 >=0 & vv1 > 0, by (id ctr yr mn pty cst)
egen new_vv1 = sum(pv1), by (id ctr yr mn cst)
```

```
gen share2 = (pv1/new_vv1) * (pv1/new_vv1)
egen denom = sum(share2), by (id ctr yr mn cst)
gen ENP_cst = 1/denom
gen cvvi = 1 if vv1 != new_vv1
drop share2 denom  pv1  vv1 new_vv1 pty
duplicates drop
joinby id ctr yr mn cst using "cst.level.dta", unmatched (using)
drop _merge

save "clea_enp_20190617_cst.level.dta", replace

*Add national level data
joinby id ctr yr mn using "national.level.dta", unmatched (master)
drop _merge
save "clea_enp_20190617_cst.level.dta", replace
compress

gen inflation5 = (ENP_nat - ENP_cst)/ENP_cst if ENP_nat != -990 & ENP_cst != -990
replace inflation5 = -990 if inflation5 ==.
duplicates drop

order ctr_n id ctr yr mn cst_n cst nvvi cvvi ENP_cst ENP_nat ENP_avg ENP_wght inflation1 inflation2 inflation3 inflation4 inflation5 PSNS PSNS_s PSNS_w PSNS_sw local_E
sort ctr_n id ctr yr mn cst_n cst nvvi cvvi ENP_cst ENP_nat ENP_avg ENP_wght inflation1 inflation2 inflation3 inflation4 inflation5 PSNS PSNS_s PSNS_w PSNS_sw local_E

label var ctr_n "Country Name"
label var ENP_cst "Effective Number of Electoral Parties at the constituency level"
label var inflation5 "Moenius & Kasuya local inflation measure"
label var cvvi "Constituency Valid Vote Indicator"

replace ENP_cst = -990 if ENP_cst ==.
replace cvvi = 0 if cvvi == .
replace ENP_nat = -990 if ENP_nat ==.
replace ENP_avg = -990 if ENP_avg ==.
replace ENP_wght = -990 if ENP_wght ==.
replace inflation1 =-990 if inflation1 == .
replace inflation2 = -990 if inflation2 ==.
replace inflation3 = -990 if inflation3 ==.
replace inflation4 = -990 if inflation4 ==.
replace inflation5 = -990 if inflation5 ==.
replace PSNS = -990 if PSNS ==.
replace PSNS_s = -990 if PSNS_s ==.
replace PSNS_w = -990 if PSNS_w ==.
replace PSNS_sw = -990 if PSNS_sw ==.
replace local_E =-990 if local_E ==.
replace ENP_cst = -990 if yr <1834 & ctr == 840

*remove all duplicates
duplicates drop ctr yr cst mn, force
save "clea_enp_20190617_cst.level.dta", replace
```

R Code for Calculations

```
#Build_ENP_Datasets(file.name="", mn=FALSE, ineq.ind="Gini",  filterIndependents=FALSE,
filterSingleConstituencies=TRUE)
#
#Parameters:
#    file_name                  - file's name. If omitted the script attempts to download CLEA data
(Stata file) from the CLEA website.
#    ineq.ind                   - by default computes indices based on "Gini" (other inequality indices,
which can be tried include "RS", "Atkinson", "Theil", "Kolm", "var", "square.var", "entropy" --
see package "ineq" for more detail).
#                               produced Gini indices can be different from the ones produced by the
Stata's INEQDECO module
#    filterIndependents         - by default includes independents.
#    filterSingleConstituencies - by default excludes single constituencies.

#Build_ENP_Datasets()  - downloads CLEA data from the CLEA website and computes the
datasets using default values.

#Output consists of three files "party.level.csv", "national.level.csv", "cst.level.csv"

#Steps to run the script:
#Step 1  Highlight the function, and press the "Run" button to get it into R's memory.
#Step 2  Run the function:  Build_ENP_Datasets(), defining any parameters of interest.
#Step 3  To run the function producing the data identical to CLEA's 9.0 version,
#        enter full path of the file:  Build_ENP_Datasets("C:/R10_mnel.dta", mn=TRUE)


###FUNCTION STARTS HERE
Build_ENP_Datasets<-function(file.name="", mn=FALSE, ineq.ind="Gini",
filterIndependents=FALSE,
                             filterSingleConstituencies=TRUE){

  usePackage <- function(required.package) {
    if (!is.element(required.package, installed.packages()[,1]))
      install.packages(required.package, dep = TRUE)
    require(required.package, character.only = TRUE)
  }

  usePackage("foreign")
  usePackage("readstata13")
  usePackage("plyr")
  usePackage("magrittr")
  usePackage("dplyr")
  usePackage("ineq")
  usePackage("XML")
```

```
#Supplementary script
if (file.name==""|length(file.name)==0){
  doc.html = htmlTreeParse('http://www.electiondataarchive.org/datacenter.html', useInternal = TRUE)
  doc.text = xpathSApply(doc.html, "//a/@href")
  download.link <-  doc.text[which(grepl("clea_\\d+_stata.zip",  doc.text))]

  temp <- tempfile()
  print("Loading data from http://www.electiondataarchive.org/...", quote=FALSE)
  download.file(paste("http://www.electiondataarchive.org", download.link, sep=""),temp)
  print("Opening data file...", quote=FALSE)
  file.name<-gsub(".zip", "", gsub("/data/releases/", "", download.link))
  data <- read.dta13(unzip(temp, paste(file.name,"/", gsub("_stata", ".dta", file.name), sep="")), convert.factors = FALSE)
  unlink(temp)} else {
    file.ext<-regmatches(file.name, regexpr(".\\w+$", file.name))
    folder.name<-gsub("\\/\\w+.\\w+$", "", file.name)
    if(nchar(folder.name)!=nchar(file.name)){setwd(folder.name)}
    if(file.ext==".csv"){data<-read.csv(file.name, stringsAsFactors = FALSE)}
    if(file.ext==".dta"){data<-read.dta13(file.name, convert.factors = FALSE)}
    if(file.ext==".tsv"){data<-read.table(file.name, stringsAsFactors = FALSE, sep="\t", quote = "", header=TRUE, fill = TRUE )}
  }

if ((mn==TRUE & !"mn"%in%names(data))|mn==FALSE){data$mn=data$mn}
```

1. Gini coefficient calculation

```
*******************
Gini Index for Parties
*******************

if (ineq.ind!="Gini" & ineq.ind%in%c("Gini", "RS", "Atkinson", "Theil", "Kolm", "var", "square.var", "entropy")==FALSE){ineq.ind="Gini"}

data.grid<-data%>%
  select(ctr_n, ctr, yr, mn, cst,  cst_n, pty, pty_n)

data<-data %>%select(ctr_n, ctr, yr, mn, cst,  cst_n, pty, pty_n, vv1, pv1, seat)

data.a<-data.b<-data %>%
  do((function(x) {
    x$vv1<-ifelse(x$vv1<0,NA,x$vv1)
    x$pv1<-ifelse(x$pv1<0,NA,x$pv1)
    na_cand <- ifelse(is.na(x$vv1) | is.na(x$pv1 ),1,0)
    data.frame(x, na_cand)})(.)) %>%
  distinct(.keep_all = TRUE) %>%
  filter(na_cand==0)%>%
  select(-na_cand)

data.c<-data %>%
  do((function(x) {
    x$vv1<-ifelse(x$vv1<0,NA,x$vv1)
    x$pv1<-ifelse(x$pv1<0,NA,x$pv1)
    x$seat<-ifelse(x$seat<0,NA,x$seat)
    na_cand <- ifelse(is.na(x$vv1) | is.na(x$pv1) | is.na(x$seat),1,0)
    data.frame(x, na_cand)})(.)) %>%
  distinct(.keep_all = TRUE) %>%
  filter(na_cand==0)%>%
  select(-na_cand)



if(filterSingleConstituencies==TRUE){
  if(filterIndependents==TRUE){
  data.a<-data.a %>%
  group_by(ctr, yr, mn)%>%
  arrange(cst)%>%
  do((function(x) {
    sin_const <- ifelse(x$cst[1] == 1 & x$cst[length(x$cst)]==1, 1, 0)
    data.frame(x, sin_const)})(.)) %>%
  filter(sin_const==0,  pty!= 6000) %>%
  select(-sin_const)%>%
  ungroup()

  data.b<-data.b %>%
  filter(pty!= 6000)

  data.c<-data.c %>%
  filter(pty!= 6000)}else{
  data.a<-data.a %>%
  group_by(ctr, yr, mn)%>%
  arrange(cst)%>%
  do((function(x) {
    sin_const <- ifelse(x$cst[1] == 1 & x$cst[length(x$cst)]==1, 1, 0)
    data.frame(x, sin_const)})(.)) %>%
  filter(sin_const==0) %>%
  select(-sin_const) %>%
  ungroup()}}

if(filterSingleConstituencies==FALSE & filterIndependents==TRUE){
  data.a<-data.a %>%
  filter(pty!= 6000)
  data.b<-data.b %>%
  filter(pty!= 6000)
  data.c<-data.c %>%
  filter(pty!= 6000)}


#Main Script
print("Computing inequality measures...", quote=FALSE)

start.timer <- proc.time()
gini<-data.b%>%
  distinct(ctr, yr, mn, cst, pty, .keep_all = TRUE)%>%   #distinct(ctr, yr, mn, cst, pty, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn, cst, pty) %>%
  mutate(
    vv1=sum(vv1,na.rm=TRUE),
    pv1=sum(pv1,na.rm=TRUE)) %>%
  group_by(ctr, yr, mn) %>%
  mutate(
    nat_vv1 = sum(pv1, na.rm=TRUE)) %>%
  group_by(ctr, yr, mn, pty) %>%
  mutate(
    pty_nat_vv1 = sum(pv1, na.rm=TRUE)) %>%
  mutate(
    nat_pvs = pty_nat_vv1/nat_vv1) %>%
  filter(nat_pvs>0.05) %>%
  mutate(
    pvs=pv1/vv1,
    pvs=ifelse(is.infinite(pvs), NA, pvs),
    pvs=ifelse(is.na(pvs), 0, pvs))%>%
  group_by(ctr, yr, mn, pty) %>%
  do((function(x) {
    giniI<-ineq(x$pvs, NULL,  type = ineq.ind, na.rm=TRUE)
    data.frame(x, giniI)})(.)) %>%
  mutate(
    giniI=ifelse(giniI<0,0,giniI),
    giniI=ifelse(giniI==0,NA,giniI)) %>%
  select(ctr_n, ctr, yr, mn, cst, pty, giniI, pv1, vv1) %>%
  arrange(ctr_n, ctr, yr, mn, cst, pty) %>%
  as.data.frame()
```

2. Party level of aggregation

```
*******************************************
*Party Nationalization Measures -Party Level Dataset
*******************************************

print("Computing Party Level Dataset...", quote=FALSE)

party.level<-data.b %>%
  distinct(.keep_all = TRUE) %>%
  left_join(gini) %>%
  mutate(
    PNS= 1 - giniI)%>%
  group_by(ctr, yr)%>%
  mutate(
  district_size=length(unique(cst)),
    PNS_s = PNS^(1/(log(district_size))))%>%
  distinct(ctr, yr, mn, cst, pty, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn)%>%
  mutate(
    nat_vv1=as.double(sum(pv1, na.rm=TRUE)))%>%
  group_by(ctr, yr, mn, pty)%>%
  mutate(
    pty_vv1=as.double(sum(pv1, na.rm=TRUE)),
    denominator = nat_vv1 * pty_vv1,
    cst_vv1=vv1,
    vote_share=pv1/cst_vv1)%>%
  arrange(ctr, yr, mn, pty, vote_share) %>%
  group_by(ctr, yr, mn, pty)%>%
  mutate(
    p_j=cumsum(pv1),
    inside = cst_vv1 * (p_j - (pv1/2)),
    numerator = sum(inside, na.rm=TRUE),
    PNS_w = (2 * numerator)/ denominator)%>%
  group_by(ctr, yr, mn, cst)%>%
  mutate(
    cst_vv1_new = sum(pv1, na.rm=TRUE),
    vote_share_new = pv1/cst_vv1_new) %>%
  group_by(ctr, yr, mn, pty)%>%
  mutate(
    p_j_new=cumsum(pv1),
    inside_new = cst_vv1_new * (p_j_new - (pv1/2)),
    numerator_new = sum(inside_new, na.rm=TRUE),
    PNS_w_new = (2 * numerator_new)/ denominator,
    alt_vv1 = 0,
    alt_vv1 = ifelse(PNS_w>=1, 1, alt_vv1),
    PNS_w = ifelse(PNS_w>=1, PNS_w_new, PNS_w),
    PNS_w = ifelse(PNS_w==1, NA, PNS_w),
    PNS_w = ifelse(PNS_w>=1, PNS_w_new, PNS_w),
    PNS_s = ifelse(PNS_s==0, NA, PNS_s),
    PNS_w = ifelse(pty == 3999|pty >= 6000|ctr == 292 |
                         ctr ==376|ctr == 674|
                        (ctr == 840 & yr == 1959 & cst ==435), NA, PNS_w),
    alt_vv1 = ifelse(is.na(PNS_w), 0, alt_vv1))%>%
    select(ctr_n, ctr, yr, mn, cst, pty_n, pty, cst_vv1, nat_vv1, PNS, PNS_s, PNS_w,
alt_vv1)%>%
  mutate(
    top = nat_vv1^2,
    square = cst_vv1^2)%>%
  group_by(ctr, yr, mn, cst)%>%
  mutate(
    pid=1:length(nat_vv1),
    helper = rep(NA,length(nat_vv1)),
    helper = ifelse(pid==1, square, helper)) %>% group_by(ctr, yr)%>%
  mutate(
    bottom=sum(helper, na.rm=TRUE),
    power_E = top/bottom,
    PNS_sw = (PNS_w)^(1/(log10(power_E))),
    PNS_sw = ifelse(is.na(PNS_sw), NA, PNS_sw),
    PNS_sw = ifelse(PNS_sw>=1, NA, PNS_sw),
    PNS = ifelse(PNS==1, NA, PNS),
    PNS_s = ifelse(PNS_s==1, NA, PNS_s),
    PNS_s = ifelse(pty >= 3996 & pty < 5000, NA, PNS_s),
    PNS_w=ifelse(pty >= 3996 & pty < 5000, NA, PNS_w),
    PNS_sw=ifelse(pty >= 3996 & pty < 5000, NA, PNS_sw))%>%
  right_join(data.grid)%>%
  select(ctr_n, ctr, yr, mn, pty_n, pty, PNS, PNS_s, PNS_w, PNS_sw)%>%
  arrange(ctr_n, ctr, yr, mn, pty_n, pty, PNS, PNS_s, PNS_w, PNS_sw)%>%
  distinct(ctr, yr, mn, pty, .keep_all = TRUE)
write.csv(party.level, "party.level.csv")
print(paste("Party Level Dataset has been successfully saved to the folder ", getwd(), sep=""),
quote=FALSE)
```

3. National level of aggregation

```
**********************************************
*Party Nationalization Measures - National Level Dataset
**********************************************
print("Computing National Level Dataset...", quote=FALSE)

national.level.nat<-data.b %>%
  distinct(ctr, yr, mn, cst, pty, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn, pty) %>%
  mutate(
    pv1=sum(pv1, na.rm=TRUE)) %>%
  distinct(ctr, yr, mn, pty, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn) %>%
  mutate(
    nat_vv1 = sum(pv1, na.rm=TRUE),
    party_prop_nat2 = (pv1/nat_vv1)^2,
    denom = sum(party_prop_nat2),
    ENEP_nat = 1/denom) %>%
  select(ctr_n, ctr, yr, mn, ENEP_nat)%>%
  mutate(
    ENEP_nat= ifelse(yr < 1834 & ctr == 840, NA, ENEP_nat),
    ENEP_nat= ifelse(yr == 1950 & ctr == 410, NA, ENEP_nat))%>%
  distinct(ctr_n, ctr, yr, mn, .keep_all = TRUE)%>%
  arrange(ctr_n, ctr, yr, mn)


national.level.nat2<-data.b %>%
  select(ctr, yr, mn, cst, pty, vv1, pv1)%>%
  distinct(.keep_all = TRUE)%>%
  group_by(ctr, yr, mn, cst, pty)%>%
  mutate(
   pv1=sum(pv1))%>%
  arrange(ctr, yr, mn, cst, pty)%>%
  distinct(.keep_all = TRUE)%>%
  group_by(ctr, yr, mn, cst)%>%
  mutate(
    new_vv1=sum(pv1),
    share2 = (pv1/new_vv1)^2,
    denom = sum(share2),
    ENEP_cst = 1/denom,
    indicator = 0,
    indicator= ifelse(new_vv1 != vv1, 1,  indicator))%>%
  distinct(ctr, yr, mn, cst, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn)%>%
  mutate(
    indicator = ifelse(sum(indicator, na.rm=TRUE)>0,1,indicator),
    nat_vv1 = sum(new_vv1, na.rm=TRUE),
    cst_wght =  new_vv1/nat_vv1,
    weighted =  cst_wght * ENEP_cst,
    ENEP_wght = sum(weighted, na.rm=TRUE),
    ENEP_avg = mean(ENEP_cst, na.rm=TRUE))%>%
  right_join(national.level.nat)%>%
  select(ctr_n, ctr, yr, mn, ENEP_nat, ENEP_avg, ENEP_wght, indicator)%>%
  mutate(
    ENEP_avg= ifelse(yr < 1834 & ctr == 840, NA,  ENEP_avg),
    ENEP_avg= ifelse(yr == 1950 & ctr == 410, NA, ENEP_avg),
    ENEP_wght= ifelse(yr < 1834 & ctr == 840, NA, ENEP_wght),
    ENEP_wght= ifelse(yr == 1950 & ctr == 410, NA, ENEP_wght))%>%
  distinct(ctr_n, ctr, yr, mn, .keep_all = TRUE)%>%
  mutate(
    Cox =  (ENEP_nat - ENEP_avg)/ ENEP_nat,
    MK_I = (ENEP_nat -  ENEP_avg)/ ENEP_avg,
    MK_I_w = (ENEP_nat - ENEP_wght)/ENEP_wght)

national.level.mk_n<- data.b %>%
  distinct(ctr, yr, mn, pty, cst, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn, cst)%>%
  mutate(
    new_vv1=sum(pv1, na.rm=TRUE),
    share2 = (pv1/new_vv1) * (pv1/new_vv1),
    denom = sum(share2, na.rm=TRUE),
    ENEP_cst = 1/denom)%>%
  left_join(national.level.nat2)%>%
  mutate(
    I_i = (( ENEP_nat-ENEP_cst)/ ENEP_cst) * 100)%>%
  group_by(ctr, yr, mn)%>%
  mutate(
    alpha = .5,
    beta = .25,
    gamma = .5,
    n_districts = max(cst),
    nat_vote = sum(new_vv1, na.rm=TRUE),
    cst_vote_proportion = new_vv1/nat_vote,
    product =  ENEP_cst *  cst_vote_proportion,
    sum_cst = sum(product, na.rm=TRUE),
    denominator = n_districts * sum_cst,
    W_tilde = ENEP_cst/ denominator,
    I_w = ((ENEP_nat - sum_cst)/sum_cst)*100,
    numerator = (I_i -  I_w)^2 * W_tilde,
    sum_numerator = sum(numerator, na.rm=TRUE),
    coeff_var_I_i = sqrt(sum_numerator)/I_w,
    numerator2 = ((I_i - I_w)^4) * W_tilde,
    sum_numerator2 = sum(numerator2, na.rm=TRUE),
    denominator2 = ((I_i - I_w)^2) * W_tilde,
    sum_denominator2 = sum(denominator2, na.rm=TRUE),
    sq_sum_denominator2 = (sum_denominator2)^2,
    kurtosis_I_i = sum_numerator2/sq_sum_denominator2,
    D =  (coeff_var_I_i)^gamma * (kurtosis_I_i)^(1 - gamma),
    MK_N = ((I_w)^alpha) * (D^(1 - alpha)),
    MK_N_two = ((I_w)^alpha) *  ((coeff_var_I_i)^beta) * ((kurtosis_I_i)^(1 - alpha -
beta)))%>%
  arrange(ctr, yr, mn, cst, pty)%>%
    group_by(ctr, yr, mn)%>%
  mutate(
    nat_vv1 = sum(pv1, na.rm=TRUE))%>%
  group_by(ctr, yr, mn, pty)%>%
  mutate(
    pty_vv1 = sum(pv1, na.rm=TRUE))%>%
  select(-c(cst,cst_n, pty_n))%>%
  distinct(ctr, yr, mn, pty, .keep_all = TRUE)%>%
  left_join(party.level)%>%
  arrange(ctr, ctr_n, yr, mn, pty, pty_n)%>%
  mutate(
    weight = pty_vv1/nat_vv1)%>%
  group_by(ctr, yr, mn)%>%
  mutate(
    PSNS = sum(PNS * weight, na.rm=TRUE),
    PSNS_s = sum(PNS_s * weight, na.rm=TRUE),
    PSNS_w = sum(PNS_w * weight, na.rm=TRUE),
    PSNS_sw = sum(PNS_sw *weight, na.rm=TRUE),
    PSNS = ifelse(PSNS == 0|PSNS > 1, NA, PSNS),
    PSNS_s= ifelse(PSNS_s == 0 | PSNS_s > 1, NA,PSNS_s),
    PSNS_w = ifelse(PSNS_w == 0 | PSNS_w > 1, NA, PSNS_w),
    PSNS_sw = ifelse(PSNS_sw == 0 | PSNS_sw > 1, NA, PSNS_sw),
    MK_N_two = as.numeric(sprintf("%.3f", MK_N_two)),
    MK_N = as.numeric(sprintf("%.3f", MK_N)))%>%
  select(ctr,  yr, mn, PSNS, PSNS_s, PSNS_w, PSNS_sw, MK_N, MK_N_two)%>%
  summarise_each(funs(mean(., na.rm=TRUE)))%>%
  distinct(ctr, yr, mn, .keep_all = TRUE)

national.level.psns<-data.c %>%
  distinct(ctr, yr, mn, pty, cst, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn)%>%
  mutate(
    nat_vote = sum(pv1, na.rm=TRUE))%>%
  group_by(ctr, yr, mn, cst)%>%
  mutate(
    seat_cst = sum(seat, na.rm=TRUE),
    party_vote_proportion =  pv1/nat_vote)%>%
  arrange(ctr, yr, mn, cst, pty)%>%
  group_by(ctr, yr, mn)%>%
  mutate(
    seat_total = sum(seat, na.rm=TRUE))%>%
  group_by(ctr, yr, mn, pty)%>%
  mutate(
    seat_contest = sum(seat_cst, na.rm=TRUE),
    seat_proportion = seat_contest/seat_total,
    local_E =  party_vote_proportion * seat_proportion)%>%
  group_by(ctr, yr, mn)%>%
  mutate(
    local_E = sum(local_E, na.rm=TRUE))%>%
  select(ctr, ctr_n, yr, mn, local_E)%>%
  distinct(ctr_n, ctr, yr, mn, .keep_all = TRUE)

national.level<-national.level.nat2 %>%
  left_join(national.level.nat)%>%
  right_join(national.level.mk_n)%>%
  left_join(national.level.psns)%>%
  rename(inflation1=Cox, inflation2=MK_I, inflation3=MK_I_w, inflation4=MK_N,
nvvi=indicator,
           ENP_nat=ENEP_nat, ENP_avg=ENEP_avg, ENP_wght=ENEP_wght)%>%
  mutate(
    nvvi = replace(nvvi, is.na(nvvi), 0))%>%
  mutate(
    inflation1 = ifelse(inflation1==0, NA, inflation1),
    inflation2 = ifelse(inflation2==0, NA, inflation2),
    inflation3 = ifelse(inflation3==0, NA, inflation3),
    inflation4 = ifelse(inflation4==0, NA, inflation4),
    local_E = ifelse(local_E == 0, NA, local_E),
    ENP_nat = ifelse(ctr == 144 & yr == 1947 | ctr == 144 & yr == 1952 | ctr == 144 & yr ==
1956 | ctr == 144 & yr == 1960 | ctr == 144 & yr == 1965 | ctr == 144 & yr == 1970 | ctr == 144
& yr == 1977, NA, ENP_nat),
    ENP_avg = ifelse(ctr == 144 & yr == 1947 | ctr == 144 & yr == 1952 | ctr == 144 & yr ==
1956 | ctr == 144 & yr == 1960 | ctr == 144 & yr == 1965 | ctr == 144 & yr == 1970 | ctr == 144
& yr == 1977, NA, ENP_avg),
    ENP_wght = ifelse(ctr == 144 & yr == 1947 | ctr == 144 & yr == 1952 | ctr == 144 & yr ==
1956 | ctr == 144 & yr == 1960 | ctr == 144 & yr == 1965 | ctr == 144 & yr == 1970 | ctr == 144
& yr == 1977, NA, ENP_wght))%>%
    right_join(data.grid)%>%
    distinct(ctr_n, ctr, yr, mn, .keep_all = TRUE)%>%
    select(ctr_n, ctr, yr, mn, nvvi, ENP_nat, ENP_avg, ENP_wght, inflation1, inflation2,
inflation3,
          inflation4, PSNS, PSNS_s, PSNS_w, PSNS_sw, local_E)%>%
    arrange(ctr_n, ctr, yr, mn, nvvi, ENP_nat, ENP_avg, ENP_wght, inflation1, inflation2,
inflation3,
            inflation4, PSNS, PSNS_s, PSNS_w, PSNS_sw, local_E)
write.csv(national.level, "national.level.csv")
print(paste("National Level Dataset has been sucessfully saved to the folder ", getwd(), sep=""),
quote=FALSE)
```

4. Constituency level of aggregation

```
**************************************************
*Party Nationalization Measures - Constituency Level Dataset
*****************************************************

*Note: this part of the file uses CLEA data to create several alternate measures of party nationalization identified in the literature */

print("Computing Constituency Level Dataset...", quote=FALSE)

cst.level<-data.b %>%
  distinct(ctr, yr, mn, cst, pty, .keep_all = TRUE)%>%
  group_by(ctr, yr, mn, cst) %>%
  mutate(
    new_vv1 = sum(pv1,na.rm=TRUE),
    share2 = (pv1/new_vv1)^2,
    denom = sum(share2),
    ENP_cst = 1/denom,
    cvvi = 0,
    cvvi = ifelse(vv1!= new_vv1, 1, cvvi))%>%
  select(ctr, yr, mn, cst, cst_n,  ENP_cst, cvvi)%>%
  distinct(ctr, yr, mn, cst, .keep_all = TRUE)%>%
  left_join(national.level)%>%
  mutate(
    inflation5 = (ENP_nat - ENP_cst)/ENP_cst,
    ENP_cst=ifelse(yr < 1834 & ctr == 840, NA, ENP_cst))%>%
  right_join(data.grid)%>%
  mutate(
    nvvi = ifelse(is.na(nvvi), 0, nvvi),
    cvvi = ifelse(is.na(cvvi), 0, cvvi))%>%
  select(ctr_n, ctr, yr, mn, cst_n, cst, nvvi, cvvi, ENP_cst, ENP_nat, ENP_avg,
         ENP_wght, inflation1, inflation2, inflation3, inflation4, inflation5,
         PSNS, PSNS_s, PSNS_w, PSNS_sw, local_E)%>%
  arrange(ctr_n, ctr, yr, mn, cst_n, cst, nvvi, cvvi, ENP_cst, ENP_nat, ENP_avg,
          ENP_wght, inflation1, inflation2, inflation3, inflation4, inflation5,
          PSNS, PSNS_s, PSNS_w, PSNS_sw, local_E)%>%
  distinct(ctr, yr, mn, cst, .keep_all = TRUE)
write.csv(cst.level, "cst.level.csv")
print(paste("Constituency Level Dataset has been sucessfully saved to the folder ", getwd(),
sep=""), quote=FALSE)
cat("The entire computation took ", proc.time()[1]-start.timer[1], "secs \n")
print("Done!", quote=FALSE)
}
#FUNCTION ENDS HERE
```