TECHNICAL APPENDIX
Electoral economics: maximizing local representation under proportionality

WERNER ANTWEILER*

June 21, 2019

This Technical Appendix is available at https://wernerantweiler.ca/public/PROAC-appendix.pdf, and accompanying data and simulation code is available at https://wernerantweiler.ca/public/PROAC-data.zip.

A The Case for Dual Optimal Assignment

The electoral system proposed in the paper suggests the use of dual optimal assignment with proportional representation (dual PR-OAC). Dual optimal assignment of constituencies is a rather novel feature among electoral systems. It entails two overlapping district assignments for each member of parliament rather than a two-member district. Thus, it is important to distinguish this model of two parliamentarians representing a single district from the competing model of a two-member (larger) district.[1] Each member of parliament is responsible for two districts, one as a 'senior' representative and one as a 'junior' representative depending on the ranking of the number of votes for each of the two parties representing each district.

The case where one party has a majority of seats results in the same party representing some districts twice. These will be the districts in which other parties received the fewest votes. Among the assignments, parties will assign different individual members of parliaments to each constituency so that no member will be both junior and senior member of a district.

Each member of parliament can be senior or junior member for two constituencies. Because of the ranking of votes, a specific member of parliament may be senior member for two constituencies, senior member for one and junior member for

*Corresponding address: Prof. Werner Antweiler, University of British Columbia, Sauder School of Business, 2053 Main Mall, Vancouver, BC, V6T 1Z2. Phone: +1-604-822.8484. E-mail: werner.antweiler@ubc.ca.

[1] Specifically, dual PR-OAC must not to be confused with the "dual-member proportional representation" (DMP) model proposed by Sean Graham. There are entirely different.

another constituency, or junior member for two constituencies. The first scenario is likely for a majority party, whereas the third scenario is likely for a small/fringe party.

The notion of a senior and junior member for a constituency is somewhat reminiscent of the notion of senior and junior senators in the US Senate. There, the senior and junior status reflects the incumbency period in a two-member district (a US state) rather than overlapping assignments for a single district.

A.1 Advantages and Disadvantages

Dual PR-OAC has a number of unique advantages and perhaps several disadvantages. On balance, this author believes that the advantages significantly outweigh the disadvantages. There are six key advantages:

i. **Improved local representation**. Voters in a constituency are represented by a senior and junior member, where the senior remember typically represents a large (often governing) party, and the junior member represents a smaller party. Simulation results show that the representation is drastically and unambiguously larger than under FPTP.

ii. **Redundancy of representation**. Having two members of parliament responsible for a jurisdiction means that if one representative is less available to the community, for example because of involvement in the government or when a member becomes ill or indisposed, then the second member for the jurisdiction has an opportunity to maintain representation of the jurisdiction in parliament. It also means that there are always two members who have familiarity with each jurisdiction and can complement each other in their effort, and in more parliamentary committees. Even if members are from different parties, they can also back up each other for the numerous local functions they need to attend. Redundancy also provides for a ″division of labor.″

iii. **Competition for representation**. The junior and senior members for a constituency can compete for effective and meaningful representation. This may help especially the junior members to obtain a stronger electoral profile rather than having no role at all. Rather than one member having exclusive rights to represent a constituency (the local representation ″monopoly″ under FPTPT), each constituency has at least two voices that will seek to stand up for local interests.

iv. **Continuity of representation**. Under FPTP many (marginal) constituencies change hands along with the electoral swing. Constituencies that change party representation may suffer discontinuities in representation along with the change in political preferences. Under dual PR-OAC there is a significantly higher chance that representation from either the senior or junior member carries over from one electoral period to the next. In some cases the role of senior and junior member may simply swap roles along with the ranking of votes in a jurisdiction.

v. **Beneficial knowledge spillovers**. A member of parliament who has to represent two constituencies learns about a greater variety of issues, which may deepen expertise and proficiency and lead to positive knowledge externalities across the two jurisdictions.

vi. **Governing-Party Access**. Dual PR-OAC enhances the likely access to representation by a member of the governing party, giving more districts a direct channel to voices in government to support their local interests. Conversely, governing-party members will find it more difficult to abandon local interests of districts that would not be represented by them under FPTP or single PR-OAC.

Naturally, there are also downsides to dual PR-OAC. Four specific disadvantages are:

i. **Dilution of representation**. As each member of parliament has to look after two constituencies, and the member of parliament is dividing time and attention across two jurisdictions. There are now potentially twice as many events and functions a member of parliament may need to or want to attend. In parliament, the member may be less effective focusing on specific local interests.

ii. **Spanning cost.** Having to cover a larger geographic are, the dually-assigned member of parliament may have to travel more often across the two constituencies. If the jurisdictions are geographically distant to each other—especially in rural areas—the member of parliament may need to maintain two offices rather than one, which in turn may result in higher cost for office space and personnel. Higher costs caused by dual representation would likely require larger funding for parliamentarians.

iii. **Conflict of representation**. The interests of two constituencies may not always align perfectly. What benefits one constituency may in some cases hurt the other jurisdiction. While such eminent conflicts may be rare in practice, they are not inconceivable.

iv. **Fringe representation**. Proportional representation gives seats in parliament to small parties.[2] Dual assignment would give fringe parties a 'local mandate' even though their vote shares may be quite small. Fringe representation may be a problem largely in the single PR-OAC model with unique seat assignments, but it is not entirely absent in the dual PR-OAC model. Fringe parties sometimes represent only rather narrow interests, or they may hold extreme views that are objectionable to most members of an electoral district.

The aforementioned disadvantages may have mitigating factors that may dampen their significance. The dilution effect is compensated by the fact that each constituency has two representatives who share the workload to some extent. Also, if the two constituencies are in proximity, it may be possible to avoid maintaining two offices; there may even be economies of scale. This will be mostly beneficial in urban districts where distances are small. Spanning costs that require more funding may also be attractive for many parliamentarians.

Conflicts of representation may be rare in practice when constituencies are in relative proximity to each other. Spatial correlation in election results would tend

[2] Some countries have electoral thresholds for obtaining seats in parliament. For example, Germany's Bundestag requires 5%, and Denmark's Folketing requires 2%.

to favour such proximity. However, smaller parties may only obtain constituency assignments in regions that are far apart geographically and thus may have a higher probability of encountering misaligned constituency objectives. The possible permutations of assignments within each party may quickly resolve such conflicts. If a party has two seats it will represent four constituencies, and assignments can be packaged in such a way that maximizes similarity and proximity. Conflict of representation can be reduced through optimal party-internal assignment, explored in the next section. Once each party has been assigned constituencies, they need to assign *specific* members of parliament to each of these constituencies.

Representation by fringe parties was flagged as a potential risk. It may also have a benefit. Fringe parties that are effectively forced to represent a local constituency cannot escape their responsibility to get involved in a broad range of local issues and articulate positions. It has often been lamented that fringe parties tend to be "single issue" parties, but if they gain seats in parliament, their exposure to local issues may help them broaden their experience—or reveal their incompetence to represent local issues and lose support in the next election. Dual representation, as opposed to single optimal assignment, mitigates the fringe representation problem drastically because fringe-party members will be junior members and there remains a senior member as the go-to local representative.

Constituency assignments have a unique advantage whether by FPTP or single/dual PR-OAC. Many jurisdictions facilitate recall petitions by voters. Usually after a minimum statutory exclusion periods, voters can petition to have a member of parliament removed. Under FPTP this triggers a by-election. Under PR-OAC it would trigger replacement of the representative by the political party which holds the seat under PR. Electoral challenges of this type would not open the opportunity to overturn the election result, but merely eject a representative for "due cause." There are several degrees of sanction that are conceivable in such a scenario: censure that would mark the representative with a 'red flag' at the next election; removal from local representation while maintaining a seat in parliament; or ejection from parliament so that the party who holds the seat has to draw a new member of parliament from its electoral list.

A.2 Party-Internal Assignments

The OAC mechanism assigns political parties to constituencies, not individual members of parliament. It is therefore necessary to define a mechanism that further assigns members of parliament to the constituencies that have been assigned to particular political parties. There are several possible mechanisms.

The first allocation mechanism may be the most obvious. It could be left entirely to the party's leadership to assign members of parliament to the party's assigned constituencies. Naturally, this would allow for elements of incumbency, seniority and proximity to particular constituencies to be reflected. It would also give party leaderships clout to reward experience, loyalty, or other desirable traits with priority access to choosing constituencies. As the available assignments are not known prior to the election, it is not possible to announce constituency assignments prior to the election. However, based on previous election results, there is a significant likelihood that certain constituencies will be represented by particular political parties, and indeed incumbents. Parties could announce that incumbents would continue to represent a constituency if a particular constituency ends up being reassigned to them based on the election outcome. This makes it possible that parties may benefit from an incumbency advantage similar to conventional FP.

The second allocation mechanism involves rules defined prior to the election, set by law, and thus announced publicly. Under PR, parties maintain lists that rank their candidates for the election. The ranking gives precedence to high-profile candidates. One way to use a formulaic assignment could be to match the party list ranking with the ranking of votes that this party received in each of the constituencies it has been assigned to represent. Thus the top-listed candidate would represent the constituency in which the party received the most votes, and the marginal candidate at the cut-off for the party list would represent the constituency in which the party received the least votes. This type of rank-order assignment is easy to codify. However, this rank-order mechanism would tend to disregard strong ties that individual candidates have formed with local communities.

Another formulaic assignment could take into account where candidates maintain their primary residence. Candidates residing in a constituency would automatically represent that constituency if it is assigned to their party. To achieve optimal local representation, strong ties to the local community are very desirable. This procedure will leave some, perhaps many, candidates who need to represent constituencies away from their home. To maintain the "proximity principle", candidates should then be assigned to constituencies in order of their distance from the community. Effectively, assignments would rely on a distance matrix of communities, and candidates are assigned either in strict order of proximity (distance between constituencies) or, again, through an optimization algorithm. The objective is similar to travel cost minimization in the classic assignment problem, assigning n resources to n tasks. The *Hungarian method* is a well-known algorithm which finds an optimal assignment for a given cost (distance) matrix. The outcome of the assignment is that candidates are optimally matched to assigned constituencies, minimizing their travel time and travel cost. More generally, let c_{kl} denote the travel distance between constituencies k and l. The objective is to assign, uniquely, a candidate living in constituency k to representing a constituency l. Let $x_{kl} \in \{0, 1\}$ denote these assignments. Then the assignment problem can be expressed as a linear program with the objective function $\sum_k \sum_l x_{kl} c_{kl}$ to be minimized subject to the two constraints $\sum_k x_{kl} = \sum_l x_{kl} = 1$ that impose uniqueness of assignment.

Whichever method is used for assigning elected members of parliament to the constituencies that are optimally assigned to their respective parties, it is very likely that it will foster stronger links between elected representatives and their communities, and thus enhance PR.

B Implementation and Numerical Complexity

The total number of possible combinations of allocating parties to constituencies is

$$H = \frac{n!}{\prod_{j=1}^{m} s_j!} \tag{1}$$

With only two parties ($m = 2$), H reaches a maximum when both parties have equal number of seats. Using Stirling's approximation for the log factorial, $\ln(n!) \approx n\ln(n) - n$, a useful connection can be made between the log of the complexity measure and the entropy measure of dispersion for the seats won by parties $j \in \{1, .., m\}$:

$$D \equiv \exp\left[-\sum_{s_j>0}\left(\frac{s_j}{n}\right)\ln\left(\frac{s_j}{n}\right)\right] \tag{2}$$

The dispersion measure can also be viewed as the 'effective number of parties' (Laakso and Taagepera, 1979). When the size of parties is perfectly equal and $s_j/n \rightarrow 1/m$, then $D = m$. For example, in a three-party system with two roughly equal-sized major parties and one very small minor party, D would be just a little over 2.

Using Sterling's approximation, it follows that $\ln(H) \approx n \cdot \ln(D)$. Complexity is largest when seats are distributed equally; when $\forall j : s_j = n/m$, then $H \approx m \cdot \exp(n)$. Ccomplexity increases linearly with the number of parties and exponentially with the number of seats. It would be computationally infeasible to inspect each solution individually when there are dozens of seats. However, branch-and-bound algorithms for solving a binary integer programming problem of the type described here are readily available.

C Simulation

The simulation consists of 20,000 random parliamentary outcomes, with eight election profiles simulated 2,500 times with varying degrees of district heterogeneity.

C.1 Simulation Objectives and Methods

The purpose of the simulation is to determine the degree of local representation R for a variety of election outcomes L, iterated over M repetitions where the level of "noise" is gradually increased between a minimum and maximum in order to capture smaller or larger variation in outcomes across the constituencies in the target jurisdiction. There are thus a total of $L \cdot M$ simulation runs. To keep the analysis compact, $L = 8$ election outcomes are modelled with 2, 3 and 5 parties. The 2-party scenario has a medium and low dispersion case, while the 3 and 5-party cases have high, medium, and low dispersion cases each.

Local representation is measured by the percentage share of voters who are represented by a candidate whose party they voted for in their constituency. For First-Past-the-Post (FPTP) this is simply the number of voters who voted for the candidate who won the constituency with a simple plurality of votes. For the single and dual optimal assignment system, local representation is measured by votes for the parties that are assigned the jurisdiction for representation.

The simulation assumes 100 seats in parliament—a number sufficiently large to cover typical parliamentary scenarios, including those with larger numbers of members but regional separation in proportional representation as those found in several federal parliaments (e.g., the European parliament, or Germany's Bundestag).

The simulation relies on the Dirichlet distribution to model proportion outcomes and commences by determining the level of "noise" for the variation of outcomes across constituencies. The precision α_0 (described in detail in the next section) is varied between 5 and 25 as the simulation proceeds from 1 to M. The low level (5) entails a "noisy" election result in which proportions change significantly from one electoral district to the next. A high level (25) entails much less variation. This range was determined through estimating a variety of real-world election outcomes in the United States, Canada, and Germany.[3] This analysis found precision parameters in the range of 5–10 for the US House of Representatives[4] and in the range of 15–25 for Canadian federal elections, province by province. The difference between Canada and the US is due to two factors. The U.S. has greater heterogeneity, exacerbated by gerrymandering; it also has only two major parties whereas Canadian provinces typically have 3-5 political parties with significant vote shares. Mathematically,

$$\hat{\alpha}_0 = (1-\chi)\cdot 5 + \chi \cdot 25 \quad \text{with} \quad \chi = \frac{m-1}{M-1} \quad \text{with} \quad m \in \{1,\dots,M\} \tag{3}$$

where $M = 2,500$ is the total number of repetitions. After determining the precision parameter, K random numbers (eaqual to the number of parties in the specific election scenario) are drawm from a gamma distribution so that $\hat{\alpha}_i \sim \Gamma(\mu_i \hat{\alpha}_0)$ with the mean μ_i given by the assumption about the jurisdiction-wide percentage shares for individual parties. To check for the quality of the resulting local outcomes it is useful to make use of the dispersion index

$$N_i \equiv \exp\left(-\sum_{k=1}^{K} \ln(p_{ik})\cdot p_{ik}\right) \tag{4}$$

based on the entropy statistic. This measure $1 \le N_i \le K$ can be interpreted as

[3]The STATA package `dirifit` from Maarten L. Buis, Nicholas Cox, and Stephen Jenkins was used to estimate these results. This package can be installed by typing in Stata: `ssc install dirifit`.

[4]Data was obtained from the MIT Election Data and Science Lab, 2017, "U.S. House 1976-2018," https://doi.org/10.7910/DVN/IG0UN2. Districts in which a candidate ran unopposed were ignored.

the “effective number of parties” represented in each riding. Each simulation step follows the following sequence:

1. Generate a random election outcome based on the target number of parties and jurisdiction-wide percentage shares. Outcomes for each constituency have increasing ”noise” as the iteration proceeds from 1 to M as explained above.
2. Determine the FPTP results along the way of generating the election outcome.
3. Determine the proportional representation for the election outcome based on the Webster-Sainte-Laguë divisor method. (An alternative version allows for d’Hondt.)
4. Generate a CPLEX input file for the single optimal assignment problem.
5. Generate a CPLEX input file for the dual optimal assignment problem, allowing for the case of a majority outcome in which the majority party represents all jurisdictions at least once, and the case of a minority outcome in which all parties compete equally for representation.
6. Run CPLEX once for each assignment problem, allowing for the dual assignment case with majority that requires using the single-assignment procedure for the remaining seats.[5]
7. The results from the optimization runs are read and merged, and the representation shares for the single and dual optimal assignment cases are calculated.
8. The results of the iteration are averaged over the number of constituencies and saved.

The final step is to average the results over the L scenarios and calculate means and standard errors for the local representation shares (FPTP, single PR-OAC, dual PR-OAC). The SAS code for the entire simulation is documented in a separate section below.

C.2 Distributional Assumptions

In order to model a variety of electoral outcomes in individual constituencies for a particular set of jurisdiction-wide vote shares, it is necessary to introduce distributional assumptions from which to draw random numbers. As it is necessary to obtain percentage shares that sum up to one hundred percent, a suitable basis is the the Dirichlet distribution, the most basic model for proportional data. The Dirichlet distribution is a multivariate generalization of the beta distribution. Specifically, the Dirichlet distribution of order K with parameters $\boldsymbol{\alpha} \equiv \{\alpha_1, ..., \alpha_K\}$ has a probability density function

$$\phi(\boldsymbol{x}; \boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} x_i^{\alpha_i - 1}}{B(\boldsymbol{\alpha})} \quad \text{with} \quad \sum_{i=1}^{K} x_i = 1 \quad \text{and} \quad B(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)} \tag{5}$$

[5]After the majority party receives a seat for each constituency, the remaining seats are assigned so that the majority party receives $2s_i - S$ seats and the minority parties receive $2s_i$ seats, where s_i is the number of seats for party i and S is the total number of seats.

for $x_i \geq 0$ and $\boldsymbol{\alpha} \equiv \{\alpha_1, ..., \alpha_K\}$. The mean of this distribution is given by

$$\mu_i = \frac{\alpha_i}{\alpha_0} \quad \text{with} \quad \alpha_0 \equiv \sum_{i=1}^{K} \alpha_i \tag{6}$$

and the variance and covariances are

$$\sigma_i^2 = \frac{\mu_i(1-\mu_i)}{1+\alpha_0} \quad \text{and} \quad \sigma_{ij} = -\frac{\mu_i \mu_j}{1+\alpha_0} \tag{7}$$

The parameter α_0 is understood as the precision.

C.3 Simulation Code: CPLEX

The PR-OAC algorithm has been implemented with IBM CPLEX®. The code below describes the use of CPLEX with the OPL interface for carrying out the maximization problem described in the paper. The code is executed by invoking the "oplrunsample" script in the OPL interface of CPLEX and passing the filenames of the code file (first below) and the data file (second below) on the command line. Input is redirected from a null device, output is redirected to a log file for further processing.

```
// Entities
{string} Parties = ...;
{string} Ridings = ...;
int Seats[Parties] = ...;
// Assignment
tuple Assignment {
  string party;
  string riding;
  int votes;
}
{Assignment} Nodes = ...;
// Decision Variables
dvar boolean X[Nodes];
// Objective Function
maximize sum(k in Nodes) (k.votes*X[k]);
// Constraints
subject to {
forall(p in Parties)
   sum(k in Nodes: k.party == p) (X[k])==Seats[p];
forall(r in Ridings)
   sum(k in Nodes: k.riding == r) (X[k])==1;
};
execute DISPLAY_RESULTS{
  for(var k in Nodes)
    if (X[k]>0)
      writeln("ASSIGNED: ",k.riding," ",k.party," ",k.votes);
}
```

The example data file below contains information from the October 2015 election in four Prince Edward Island ridings. The island is the smallest Canadian province.

```
Parties = {"LIB","CPC","NDP"};
Ridings = {"11001","11002","11003","11004"};
Seats = [2,1,1]; // seats for parties based on PR
Nodes = {
  <"LIB","11001",14621>, <"CPC","11001",3632>, <"NDP","11001",2503>,
  <"LIB","11002",11910>, <"CPC","11002",3136>, <"NDP","11002",4897>,
  <"LIB","11003",10521>, <"CPC","11003",6185>, <"NDP","11003",4097>,
  <"LIB","11004",13950>, <"CPC","11004",3947>, <"NDP","11004",2509>};
```

C.4 Simulation Code: SAS

The code below documents the full simulation study described in the paper. The code can be easily adapted to investigate any number of additional scenarios. The macro SIMULATE is invoked with the number of parties, a model name, and a list of vote shares for the jurisdiction-wide result.

```
*-----------------------------------------------------------------------+
| PR-OAC Electoral System - Simulation and Evaluation                   |
| (C) 2019 by Prof. Werner Antweiler, University of British Columbia    |
+-----------------------------------------------------------------------;
options nocenter ls=80;
libname here '.';
%LET PRINT=0;
%LET METHOD=WEBSTER;
%LET SEATS=100;
%LET ITERATIONS=2500;

%MACRO SIMULATE(PARTIES,MODEL,VLIST);
%DO ITER=1 %TO &ITERATIONS;
data election;
  *-----------------------------------------------+
  | generate a randomized election outcome        |
  | draw from Dirichlet distribution              |
  | with mean mu[i]=alpha[i]/xi                   |
  | xi=sum(alpha[i],i=1..k)                       |
  | and variance sigma[i]^2=mu[i]*(1-mu[i])/(1+xi) |
  +-----------------------------------------------;
  seats=&SEATS;
  parties=&PARTIES;
  array votes[1:&PARTIES] v1-v&PARTIES;
  array alpha[1:&PARTIES] alpha1-alpha&PARTIES;
  array mu[1:&PARTIES] _temporary_ (&VLIST);
  *call streaminit(12345);
  array noise[0:1] _temporary_ (5 25);
  if &ITERATIONS=1 then chi=0.5;
  else chi=(&ITER-1)/(&ITERATIONS-1);
  xi=(1-chi)*noise[0]+chi*noise[1];
  call symput('XI',put(xi,best10.));
  length seat $4;
  do r=1 to seats;
    seat='R'||put(r,Z3.);
    *-----------------------------------------------+
    | draw {k} random numbers from Gamma distribution |
    | and then transform into Dirichlet distribution  |
    +-----------------------------------------------;
```

```
    do i=1 to parties;
      alpha[i]=rand('GAMMA',xi*mu[i]/100);
    end;
    vsum=sum(of alpha[*]);
    entropy=0;
    do i=1 to parties;
      alpha[i]=alpha[i]/vsum;
      entropy=entropy-log(alpha[i])*alpha[i];
    end;
    *---------------------------------------------------+
    | Determine First-Past-the-Post result and          |
    | turn shares into integer results based on         |
    | 100,000 voters per constituency                   |
    +---------------------------------------------------;
    winner=0; maxvotes=0; allvotes=0;
    do i=1 to parties;
      votes[i]=floor(1E5*alpha[i]+0.5);
      if votes[i]>maxvotes then do;
        winner=i;
        maxvotes=votes[i];
      end;
      allvotes=allvotes+votes[i];
    end;
    represented0=maxvotes;
    output;
  end;
  keep seat v1-v&PARTIES entropy winner represented0 allvotes;
  label seat='Seat/Constituency/Riding Identifier'
    entropy='Entropy Index of Dispersion'
    winner='Party of Winner in FPTP'
    represented0='Voters represented under FPTP'
    allvotes='Sum of votes (needed because of rounding)';
proc sort data=election;
  by seat;
data PR_seats;
  set election end=verylast;
  retain print &PRINT;
  if print then  file 'simulation.txt';
  parties=&PARTIES;
  *---------------------------------------------------+
  | Determine Proportional Representation results    |
  +---------------------------------------------------;
  array votes[1:&PARTIES] v1-v&PARTIES;
  array seats[1:&PARTIES] _temporary_;
  array total[0:&PARTIES] _temporary_;
  if _N_=1 then do;
    total[0]=0;
    do i=1 to parties;
      seats[i]=0;
      total[i]=0;
    end;
  end;
  if print then put seat $4. ': ' allvotes comma7.0 ' :' @;
  do i=1 to parties;
    pct=100*votes[i]/allvotes;
    if print then  put '   ' pct 5.2 @;
    total[i]=total[i]+votes[i];
  end;
  total[0]=total[0]+allvotes;
  seats[winner]=seats[winner]+1;
  if print then do;
    code=substr('ABCDEFGHIJK',winner,1);
    put ' : ' code $1. @;
    N=exp(entropy);
    put ' : ' N 5.3;
  end;
  drop i pct code;
  if verylast then do;
```

```
    if print then  do;
      put; put 'First Past the Post';
      do i=1 to parties;
        code=substr('ABCDEFGHIJK',i,1);
        put 'Party ' code $1. ':' @;
        share=100*seats[i]/&SEATS;
        put ' ' seats[i] 3.0 ' (' share 5.2 '%) seats' @;
        share=100*total[i]/total[0];
        put ' ' share 5.2 '% votes';
      end;
    end;
    *--------------------------------------------------+
    | Gallagher Index of Disproportionality            |
    +--------------------------------------------------;
    gallagher=0;
    do i=1 to parties;
       s_share=100*seats[i]/&SEATS;
       v_share=100*total[i]/total[0];
       gallagher=gallagher+(s_share-v_share)**2;
    end;
    gallagher=sqrt(gallagher/2);
    label gallagher='Gallagher Index of Disproportionality';
    if print then do;
      put 'Gallagher Index: ' gallagher 8.3;
    end;
    *--------------------------------------------------+
    | Allocation of seats based on d_Hondt             |
    | or by Webster/Sainte-ëLagu method                |
    +--------------------------------------------------;
    array x[&PARTIES] _temporary_;
    array z[&PARTIES] _temporary_;
    array s[&PARTIES] PR_seats1-PR_seats&PARTIES;
    do i=1 to parties;
      x[i]=total[i];
      s[i]=0;
      z[i]=x[i];
    end;
    do j=1 to &SEATS;
      maxz=0; maxi=0;
      do i=1 to parties;
        if z[i]>maxz then do; maxz=z[i]; maxi=i; end;
      end;
      s[maxi]=s[maxi]+1;
      if "&METHOD"='DHONDT' then z[maxi]=x[maxi]/(1+s[maxi]);
      else if "&METHOD"='WEBSTER' then z[maxi]=x[maxi]/(1+2*s[maxi]);
      else put 'unspecified method';
    end;
    length majority $1;
    majority='X';
    threshold=ceil(&SEATS/2);
    do i=1 to &PARTIES;
      if s[i]>=threshold then majority=substr('ABCDEFGHIJK',i,1);
    end;
    call symput('MAJORITY',majority);
    if print then do;
      put; put 'Proportional Representation';
      do i=1 to parties;
        code=substr('ABCDEFGHIJK',i,1);
        put 'Party ' code $1. ':' @;
        share=100*s[i]/&SEATS;
        put ' ' s[i] 3.0 ' (' share 5.2 '%) seats' @;
        share=100*total[i]/total[0];
        put ' ' share 5.2 '% votes';
      end;
      if majority='X' then put 'Plurality but no majority';
      else put 'Party ' majority $1. ' has majority of seats';
    end;
    output;
```

```
  end;
  keep majority PR_seats1-PR_seats&PARTIES gallagher;
data cplex;
  set election end=verylast;
  file 'election1.dat';
  *--------------------------------------------------+
  | Generate input file for CPLEX/OPL software       |
  | Standard version for single-seat PR-OAC          |
  | and dual-seat PR-OAC without majority            |
  +--------------------------------------------------;
  length code $1;
  if _N_=1 then do;
    put '/*--------------------------------------------------';
    put '  CPLEX OPL Maximization Problem';
    put '  Simulated Election Data';
    put '  Werner Antweiler, University of British Columbia';
    put '--------------------------------------------------*/';
    put 'Parties = {' @;
    do i=1 to &PARTIES;
      code=substr('ABCDEFGHIJKLM',i,1);
      put '"' code $1. '"' @;
      if i<&PARTIES then put ',' @;
    end;
    put '};';
    put 'Ridings = {' @;
    do i=1 to &SEATS;
      put '"R' i Z3.0 '"' @;
      if i=&SEATS then put '};';
      else if mod(i-1,10)=0 then put ',';
      else put ',' @;
    end;
    set PR_seats;
    array s[&PARTIES] PR_seats1-PR_seats&PARTIES;
    put 'Seats = [' @;
    do i=1 to &PARTIES;
      put s[i] +(-1) @;
      if i<&PARTIES then put ',' @;
      else put '];';
    end;
    put 'Nodes = {';
  end;
  array v[&PARTIES] v1-v&PARTIES;
  do i=1 to &PARTIES;
    code=substr('ABCDEFGHIJKLM',i,1);
    put '  <"' code $1. '","' seat $4. '",' v[i] +(-1) '>' @;
    if i<&PARTIES then put ',';
  end;
  if verylast then put '};';
  else put ',';
  drop _ALL_; delete;
data cplex;
  set election end=verylast;
  file 'election2.dat';
  *--------------------------------------------------+
  | Generate input file for CPLEX/OPL software       |
  | special case for dual-seat PR-OAC with majority  |
  +--------------------------------------------------;
  length code $1;
  length majority $1;
  retain majority;
  if _N_=1 then do;
    majority=symget('MAJORITY');
    put '/*--------------------------------------------------';
    put '   CPLEX OPL Maximization Problem';
    put '   Simulated Election Data';
    put '   Werner Antweiler, University of British Columbia';
    put '--------------------------------------------------*/';
    put 'Parties = {' @;
```

```
    do i=1 to &PARTIES;
      code=substr('ABCDEFGHIJKLM',i,1);
      put '"' code $1. '"' @;
      if i<&PARTIES then put ',' @;
    end;
    put '};';
    put 'Ridings = {' @;
    do i=1 to &SEATS;
      put '"R' i Z3.0 '"' @;
      if i=&SEATS then put '};';
      else if mod(i-1,10)=0 then put ',';
      else put ',' @;
    end;
    set PR_seats;
    array s[&PARTIES] PR_seats1-PR_seats&PARTIES;
    put 'Seats = [' @;
    do i=1 to &PARTIES;
      code=substr('ABCDEFGHIJKLM',i,1);
      if majority='X' then put s[i] +(-1) @;
      else do;
        if majority^=code then needed=2*s[i];
        else needed=2*s[i]-&SEATS;
        put needed +(-1) @;
      end;
      if i<&PARTIES then put ',' @;
      else put '];';
    end;
    put 'Nodes = {';
  end;
  array v[&PARTIES] v1-v&PARTIES;
  do i=1 to &PARTIES;
    code=substr('ABCDEFGHIJKLM',i,1);
    put '  <"' code $1. '","' seat $4. '",' @;
    *----------------------------------------------------+
    | a majority party gets one seat in each riding      |
    | and remaining seats are based on zero extra        |
    | representation                                     |
    +----------------------------------------------------;
    if code=majority then put '0' @;
    else put v[i] +(-1) @;
    put '>' @;
    if i<&PARTIES then put ',';
  end;
  if verylast then put '};';
  else put ',';
  drop _ALL_; delete;
run;

data null;
  *----------------------------------------------------+
  | run the simulation with CPLEX                      |
  +----------------------------------------------------;
  call system("/usr/local/bin/cplex-opl election1.mod election1.dat </dev/null >election1.log");
  call system("/bin/gawk -f election.awk election1.log >election1.opt");
  *----------------------------------------------------+
  | for dual-seat PR-OAC, run two-seat model for case|
  | without majority, and single-seat truncated case |
  | when one party has a majority and gets at least  |
  | one seat in each riding                          |
  +----------------------------------------------------;
  if symget('MAJORITY')='X' then
    call system("/usr/local/bin/cplex-opl election2.mod election2.dat </dev/null >election2.log");
  else
    call system("/usr/local/bin/cplex-opl election1.mod election2.dat </dev/null >election2.log");
  call system("/bin/gawk -f election.awk election2.log >election2.opt");
```

```
run;

data results1;
  *--------------------------------------------------------+
  | Read results from single-seat PR-OAC simulation       |
  +--------------------------------------------------------;
  infile "election1.opt" dsd delimiter=',';
  if _N_=1 then do;
    input optimum;
    retain optimum;
  end;
  else do;
    length seat $4;
    length assigned $1;
    input seat $ assigned $ represented1;
    output;
  end;
  keep seat assigned represented1;
proc sort data=results1;
  by seat;
data results2;
  *--------------------------------------------------------+
  | Read results from dual-set PR-OAC simulation          |
  +--------------------------------------------------------;
  infile "election2.opt" dsd delimiter=',';
  if _N_=1 then do;
    input optimum;
    retain optimum;
  end;
  else do;
    length seat $4;
    length party $1;
    input seat $ party $ votes;
    output;
  end;
  keep seat party votes;
proc sort data=results2;
  by seat descending votes;
data results2;
  set results2;
  by seat descending votes;
  retain represented2 rank;
  length junior senior $1;
  if first.seat then do;
    represented2=0; rank=0;
  end;
  represented2=represented2+votes;
  rank=rank+1;
  if first.seat=last.seat then do;
    * majority case ;
    senior=symget('MAJORITY');
    junior=party;
  end;
  else do;
    if rank=1 then senior=party;
    else if rank=2 then junior=party;
    else put 'ERROR: rank>2';
  end;
  if last.seat then output;
  keep seat junior senior represented2;
data election;
  *--------------------------------------------------------+
  | Merge results and analyze                             |
  +--------------------------------------------------------;
  merge election results1 results2;
  by seat;
  *--------------------------------------------------------+
  | adjust total representation for dual-seat PR-OAC |
```

```
  | when one party has a majority of seats                    |
  +-----------------------------------------------------------;
  length majority $1;
  majority=symget('MAJORITY');
  array votes[1:&PARTIES] v1-v&PARTIES;
  if majority^='X' then do;
    do i=1 to &PARTIES;
      code=substr('ABCDEFGHIJKLM',i,1);
      if code=majority   /* majority party votes */
      and junior^=senior /* do not double count */
      then represented2=represented2+votes[i];
    end;
  end;
data summary;
  set election end=verylast;
  by seat;
  retain rep0 rep1 rep2 total 0;
  rep0=rep0+represented0;
  rep1=rep1+represented1;
  rep2=rep2+represented2;
  total=total+allvotes;
  *put seat $4.
     ' ' represented0 comma7.0
     ' ' represented1 comma7.0
     ' ' represented2 comma7.0;
  if verylast then do;
    set PR_seats;
    length model $1;
    model="&MODEL";
    seats=&SEATS;
    parties=&PARTIES;
    noise=input(symget('XI'),10.);
    rep0=100*rep0/total;
    rep1=100*rep1/total;
    rep2=100*rep2/total;
    iteration=&ITER;
    output;
  end;
  keep iteration noise seats parties model noise
    rep0 rep1 rep2 gallagher majority;
proc append base=simulation data=summary;
run;
%END;
%MEND;

*-----------------------------------------------------------------------+
| SIMULATE(PARTIES,MODEL,VOTESHARELIST)                                 |
+-----------------------------------------------------------------------;
options nosource nonotes;
%SIMULATE(2,A,60 40);
%SIMULATE(2,B,52 48);
%SIMULATE(3,A,55 35 10);
%SIMULATE(3,B,45 35 20);
%SIMULATE(3,C,40 35 25);
%SIMULATE(5,A,50 20 15 10  5);
%SIMULATE(5,B,40 25 20 10  5);
%SIMULATE(5,C,30 25 20 15 10);
options source notes;

proc sort data=simulation out=here.simulation;
  by parties model iteration;
proc print data=here.simulation;
  id parties model iteration;
  var rep0 rep1 rep2;
run;
endsas;
```

D Disproportionality Index

The paper employs the Gallagher (1991); Gallagher and Mitchell (2005) index to measure the disproportionality of election outcomes. Disproportionality arises from the concentration of overall vote shares in combination with the heterogeneity of constituencies (the α_0 parameter of the Dirichlet distribution, section C.1 above). The most widely used statistic for this purpose in the Political Science literature is the Gallagher index ($\mathcal{G}$), which is defined as

$$\mathcal{G} \equiv 100 \cdot \sqrt{\frac{1}{2}\sum_{i=1}^{n}\left(\frac{s_i}{s} - \frac{v_i}{v}\right)^2}$$

where s_i/s and v_i/v are the seat and vote shares for party i, and n is the number of parties. The larger the differences between the vote share and seat share summed over all parties, the larger the Gallagher index, and thus in turn the larger the disproportionality. There are a number of alternatives measures described in Karpov (2008).

It is useful to compare these measures with actual election results. The US House of Representatives typically has values around 5, while Canada's more fractured political landscape reveals values of about 10-15. The $\mathcal{G}$ values in the table correspond to that for the narrow outcomes typical of US elections (second line for $n = 2$) and Canada's 3-party system (third line of $n = 3$). When vote share are highly concentrated, The $\mathcal{G}$ can get very large and more volatile.

Reporting the Gallagher index serves two purposes. First, it provides a reference point for identifying which simulation groups correspond to more "typical" election outcomes. Second, it underlines that higher degrees of vote share concentration generate higher levels of disproportionality under First-Past-The-Post (FPTP). It further demonstrates the cost of FPTP: very small increases in average voter representation come at the cost of very high disproportionality.

References

Gallagher, Michael (1991). Proportionality, disproportionality, and electoral systems. Electoral Studies, 10(1), 33–51.

Gallagher, Michael and Mitchell, Paul (2005). The politics of electoral systems. Cambridge University Press.

Karpov, Alexander (2008). Measurement of disproportionality in proportional representation systems. Mathematical and Computer Modelling, 48(9-10), 1421–1438.

Laakso, Markku and Taagepera, Rein (1979). "Effective" Number of Parties: a Measure with Application to West Europe. Comparative Political Studies, 12(1), 3–27.