Cabinet formation and portfolio distribution in European multi-party systems

Josh Cutler
Department of Political Science, Duke University
josh.cutler@duke.edu

Scott de Marchi
Department of Political Science, Duke University
demarchi@duke.edu

Max Gallop
Department of Political Science, Duke University
max.gallop@duke.edu

Florian M. Hollenbach
Department of Political Science, Duke University
florian.hollenbach@duke.edu

Michael Laver
Department of Politics, New York University
michael.laver@nyu.edu

Matthias Orlowski
Berlin Graduate School of Social Sciences, Humboldt-University Berlin
mace84@gmail.com

APPENDIX 1

As with any algorithm for calculating MIWs, we must first generate all minimum winning coalitions; we do this recursively:

```
#define recursive coalition combinatorics generator
#note this finds ALL coalitions, not just MWC's or min. weight coal.'s
def combo(tcoal, tlim, n, L1, tl):
    # tcoal is coalition list, tlim is # of parties in coal, n is starting party
    # and L1 is list of coalitions used to fill dictionary coal_list
    for j in range(n,T-tlim+1):
        tl.append(j)
        L1.append(tcoal[j])
        if tlim-1>0:
            combo(tcoal, tlim-1, j+1, L1, tl)
        if tlim==1:
            tsum=0
            for z in range(0,len(L1)): tsum=tsum+L1[z]
            coal_list[tuple(L1)]=tsum
            if E%2==0 and tsum==(u-1):
                tie_list.append(tuple(tl))
            elif tsum>=u:
                pulp_list.append(tuple(tl))
                for k in L1:
                    if tsum - k >= u:
                        pulp_list.pop()
                        break
            else:
                loss_list.append(tuple(tl))
        L1.pop()
        tl.pop()
    return 0
```

Solving for MIWs is a straightforward application of linear programming and we use the PuLP library to implement our algorithm (http://packages.python.org/PuLP/).[1]

[1] For the best verbal description of the algorithm, see Strauss (2003). His code is not, however, open-source, though the java program can be run from http://www.mindlessphilosopher.net/weights/. Our version of the algorithm chooses a slightly different set of assumptions than Strauss (see p. 6 of this paper), is not prone to mistakes, but is slower for very large numbers of parties. Our code is available from the authors.

APPENDIX 2

Table A2.1: Descriptive statistics of key variables

Variable	N	Mean	SD	Min	Max
cabinet_seat_share	3729	0.127393	0.250542	0	1
seats_share	3737	0.133264	0.14457	0.001518	0.635812
miw_share	3709	0.132549	0.164013	0	1
election_year	3737	1977.327	18.53296	1939	2011
bicameral	3737	0.618411	0.485842	0	1

APPENDIX 3

To facilitate comparison with previously published results, we now present estimates derived from OLS regressions and including all observations used in other studies – despite the fact that these models are mis-specified for the reasons we have given. The OLS estimates in Tables A 3.1 and A3.2 are largely in accord with the zero inflated beta models presented above, but a few additional details are worth pointing out. Support for the role played by the two independent variables of interest (i.e., MIW's and raw weights) is provided by the bootstrapped estimates in Table A3.1 which demonstrates that MIW's are in fact utilized by parties when one includes both stages of bargaining (i.e., admission to a coalition and then the subsequent division of perquisites). Table A3.2, accordingly, shows that MIW's are far less predictive of cabinet seat allocations once one constrains the sample, as previous work has done, to only those parties that are in the government. In qualitative terms, this supports our contention that MIW's predict entry to coalitions, especially in the low complexity case of eight or fewer effective parties. But, raw weights are dominant in predicting seat shares once the coalition is established.

Table A3.1: OLS Regression Full Sample

Bootstrap				**N**	3701
Replications	1000			**Wald chi2(4)**	2724.47
				Prob > chi^2	0
				R^2	0.57
				Adj R^2	0.57
				Root MSE	0.164

DV: cabinet_share	**Coef.**	**Std. Err.**	**z**	**P>|z|**	**[95% Conf. Interval]**	
miw_share	0.72959	0.033393	21.85	0	0.6641414	0.7950377
seats_share	0.548392	0.045642	12.01	0	0.4589345	0.6378494
italy	0.010611	0.024198	0.44	0.661	-0.0368161	0.0580389
bicameral	0.009772	0.005844	1.67	0.094	-0.0016814	0.0212252
_cons	-0.0485	0.004868	-9.96	0	-0.0580426	-0.0389625

Table A3.2: OLS Regression excluding non-cabinet members

Bootstrap				**Number of obs**	1137
Replications	1000			**Wald chi2(4)**	4059.16
				Prob > chi^2	0
				R^2	0.77
				Adj R^2	0.77
				Root MSE	0.142

DV: cabinet_share	**Coef.**	**Std. Err.**	**z**	**P>|z|**	**[95% Conf. Interval]**	
miw_share	0.199748	0.038367	5.21	0	0.124549	0.2749466
seats_share	1.347314	0.054709	24.63	0	1.240086	1.454541
italy	0.026754	0.03744	0.71	0.475	-0.0466275	0.1001354
bicameral	-0.03683	0.009039	-4.07	0	-0.0545451	-0.0191141
_cons	0.071515	0.00835	8.56	0	0.0551486	0.0878817