`nopp` 1.0.0 (Nash Optimal Party Positions): An R-Package

Luigi Curini* Stefano M. Iacus†

October 16, 2012

Abstract

`nopp` is a package for R which enables to compute party/candidate ideological positions that correspond to a Nash Equilibrium along a one-dimensional space. It accommodates alternative motivations in (each) party strategy while allowing to estimate the uncertainty around their optimal positions through two different procedures (bootstrap and MC).

Keywords: spatial theory of voting, Nash equilibrium, voter choice, political science

1 Introduction

Since Downs' seminal work, spatial theory of voting has been extremely useful in deepening our understanding of party strategic interactions. However, at least in multiparty context the expectations in terms of equilibrium that the theory provides, in particular the high degree of policy convergence (Adams 1999), usually do not accord very well with actual parties behaviours. In this respect, a number of works have recently aimed at filling the gap between the theoretical predictions and the empirical world of party systems. Among

*Department of Social and Political Sciences, University of Milan, Via Passione 13, I-20123 Milan, Italy. E-mail: `luigi.curini@unimi.it`

†Department of Economics, Management and Quantitative Methods, University of Milan, Via Conservatorio 7, I-20123 Milan, Italy. E-mail: `stefano.iacus@unimi.it`

the others, Schofield and Sened (2006), Merrill and Adams (2001), Adams et al. (2005) and Adams and Merrill (2003), have developed models that by recognizing the importance of non-policy variables in affecting voters' behaviours, are able to produce optimal parties strategies that appear much more in line with real-world elections.

Following this logic, `nopp` computes party/candidate ideological positions along a one-dimensional space that correspond to a Nash Equilibrium. It employs a two-step procedure:

1. First: it uses survey data to estimate individual respondents' vote choice through an empirical model.

2. Second: it uses the parameters estimates of such empirical model to search for an equilibrium configuration in party locations. To this end `nopp` implements the Merrill and Adams (MA) iterative algorithm (see Merill and Adams 2001, Adams, Merrill and Grofman 2005).

Main advantages of `nopp`:

- Easy to use: two lines of commands to run the entire procedure.

- It takes advantage of the flexibility of `mlogit` package with respect to the choice models that can be estimated.

- It accommodates alternative motivations in (each) party strategy.

- It allows to estimate the uncertainty around the optimal position of parties through two different procedures (bootstrap and MC).

- It also incorporates the possibility to produce a graphical representation of the findings.

2 The algorithm

A typical voter utility function with both policy and nonpolicy factors can be expressed as following (see Merill and Adams 2001, Adams et. al. 2005, Calvo and Hellwig 2011):

$$U_{ik}(s,a) = -a(x_i - s_k)^2 + \beta t_{ik} + \delta z_i + \epsilon_{ik} \tag{1}$$

where U_{ik} is the utility of voter i to vote for party k. With respect to the policy component in U_{ik}, x_i and s_k are respectively the ideal point of elector i and party k's location on the underlying policy dimension, a describes the weight, or salience, of the voter's proximity preference, and $(x_i - s_k)^2$ is a quadratic term measuring the ideological proximity of voter i to party k. With respect to the nonpolicy component in U_{ik}, β describes the weight that the voter gives to the nonspatial components of her vote choice relative to party k, such as partisanship or the assessments about the valence endowment of that party, t_{ik}; z_i is a vector of i's individual attributes (sex, education, etc.) while δ describes their corresponding salience in voter's choice. Finally ϵ_{ik} describes a stochastic error term. One plausible assumption (among the others: see below) is that the values of ϵ are generated independently from a type I extreme-value distribution – the assumption that characterizes the Conditional Logit (CL) model.

The choice model maximizes the random utility function in (1) to estimate the probability a voter i will select party k (i.e., P_{ik}):

$$P_{ik}(s,a) = \frac{\exp\{U_{ik}(s,a)\}}{\sum_{k=1}^{K} \exp\{U_{ik}(s,a)\}}, \quad \forall i,k. \tag{2}$$

Adams et al. (2005) show that the random utility model in equations (1) and (2) can be used to search for a Nash equilibrium in parties location, i.e., a combination of strategies used such that each party k cannot increase its vote share by changing strategy. Starting with the estimated model in (1), the vote share for party k is given by the expected value:

$$EV_k(s^*,a) = \sum_i P_{ik}(s^*,a) \tag{3}$$

where the expected vote share, EV_k, is the sum of the probabilities of voting for party k given all parties' strategic locations, $s_k^* \in s^*$. At a Nash equilibrium, the partial derivative of EV_k with respect to s_k must be zero for all k. That is, for all k:

$$\frac{\partial}{\partial s_k} EV_k(s^*,a) = 0 \tag{4}$$

From this, and solving for s_k^* we obtain the location at which party k maximizes its vote share:

$$s_k^* = \frac{\sum_i P_{ik}(s^*,a)\,(1 - P_{ik}(s,a))\,x_i}{\sum_i P_{ik}(s^*,a)\,(1 - P_{ik}(s,a))} \tag{5}$$

By iteratively solving for each party's preferred location, MA provides an algorithm to find each party's equilibrium s^* (see Merrill and Adams, 2001, for a proof of the conditions that guarantee the existence and uniqueness of Nash equilibria for each party k). In particular, the iterative algorithm updates the spatial location of each party in response to changes in the location of all other parties, with fixed proximity and nonproximity parameters a, β and δ as they result from the empirical model of equation (1). Estimating the equilibrium strategy of party k does not therefore require information about the actual vote choice of the respondents as in equation (2).

3 An application: the 2006 Italian general election

3.1 First step: the empirical choice model

`italy2006` included in `nopp` package contains data from the 2006 Italian General Election survey (source: CSES - Comparative Study of Electoral Systems: `http://www.cses.org/`). In this survey respondents were asked to indicate which party they voted for in the 2006 Election. The data concerns 5 parties: UL (Ulivo), RC (Communist Refoundation party), FI (Forza Italia), AN (National Alliance) and UDC (Union of Christian Democrats).

To work `nopp` requires the data frame to be in a long-format (i.e., each row is an alternative) rather than in a wide-format (i.e., each row is an observation). The `mlogit.data` routine in `mlogit` (Croissant 2012) allows to transform a wide-data frame into a long one in an easy way. In the case of `italy2006`, the original dataset presents a wide format.

`nopp` is loaded using :

```
> library(nopp)
```

To load `italy2006`:

```
> data(italy2006)
> str(italy2006)
```

```
'data.frame':	438 obs. of  18 variables:
 $ country    : chr  "Italy2006" "Italy2006" "Italy2006" "Italy2006" ...
 $ id         : num  1 2 3 4 5 6 7 8 9 10 ...
```

```
 $ vote       : Factor w/ 5 levels "FI","UL","AN",..: 2 2 2 2 2 5 2 4 3 1 ...
 $ self       : int  4 4 3 4 2 0 2 9 9 6 ...
 $ prox_FI    : num  -14.2 -14.2 -22.7 -14.2 -33.3 ...
 $ prox_UL    : num  -0.211 -0.211 -0.292 -0.211 -2.373 ...
 $ prox_AN    : num  -18.3 -18.3 -27.8 -18.3 -39.4 ...
 $ prox_UDC   : num  -3.95 -3.95 -8.92 -3.95 -15.9 ...
 $ prox_RC    : num  -4.34795 -4.34795 -1.1776 -4.34795 -0.00725 ...
 $ partyID_FI : num  0 0 0 0 0 0 0 1 0 1 ...
 $ partyID_UL : num  0 1 0 1 1 0 1 0 0 0 ...
 $ partyID_AN : num  0 0 0 0 0 0 0 0 1 0 ...
 $ partyID_UDC: num  0 0 0 0 0 0 0 0 0 0 ...
 $ partyID_RC : num  0 0 0 0 0 1 0 0 0 0 ...
 $ sex        : int  1 0 1 0 0 1 1 0 0 1 ...
 $ age        : int  4 6 3 5 6 5 1 4 2 1 ...
 $ education  : int  1 0 1 0 4 2 4 2 2 2 ...
 $ gov_perf   : int  3 4 3 3 3 4 4 2 2 2 ...
```

where: `id` (respondent identifier), `vote` (the party voted), `self` (self-placement of respondent on a 0 to 10 left-right scale), `prox_`$*$ (ideological distance between the respondent and a party placement – estimated as $-(\texttt{self}_i - s_k)^2$, where s_k is the survey respondents' mean party k placement used as party k actual position), `partyID_`$*$ (`partyID_`$* =$ `1` if the respondent declares to feel herself close to that party, 0 otherwise).

To transform the data frame in a long-format, we would write:

```
> colnames(italy2006)

 [1] "country"     "id"          "vote"        "self"
 [5] "prox_FI"     "prox_UL"     "prox_AN"     "prox_UDC"
 [9] "prox_RC"     "partyID_FI"  "partyID_UL"  "partyID_AN"
[13] "partyID_UDC" "partyID_RC"  "sex"         "age"
[17] "education"   "gov_perf"

> election <- mlogit.data(italy2006, shape="wide", choice="vote",
           varying=c(5:14), sep="_")
```

The compulsory arguments are `choice`, which is the variable that indicates the choice made (in our case variable `vote`), the `shape` of the original `data.frame` (in our case: `wide`) and, if there are some alternative specific variables (in our case: `prox_*` and `partyID_*`) `varying` which is a numeric vector that indicates which columns contains alternative specific variables. This argument is then passed to reshape that coerced the original `data.frame` in long format. Further arguments may be passed to reshape. For example, if the names of the variables are of the form `var_alt` (as in our case), one should add `sep` = "_".

```
> head(election)

        country id  vote self sex age education gov_perf
1.AN  Italy2006  1 FALSE    4   1   4         1        3
1.FI  Italy2006  1 FALSE    4   1   4         1        3
1.RC  Italy2006  1 FALSE    4   1   4         1        3
1.UDC Italy2006  1 FALSE    4   1   4         1        3
1.UL  Italy2006  1  TRUE    4   1   4         1        3
2.AN  Italy2006  2 FALSE    4   0   6         0        4
      alt        prox partyID chid
1.AN   AN -18.2756214       0    1
1.FI   FI -14.1908979       0    1
1.RC   RC  -4.3479486       0    1
1.UDC UDC  -3.9490445       0    1
1.UL   UL  -0.2111659       0    1
2.AN   AN -18.2756214       0    2
```

The result is a `data.frame` in long format with one line for each alternative. The choice variable is now a logical variable and the individual specific variables (`sex`, `age`, `education` and `gov_perf`) are repeated 5 times. An index attribute is added to the data, which contains the two relevant indexes: `chid`, the choice index, and `alt` index.

As the first step of our analysis, we need to estimate equation (1). Let's assume as an illustrative example that the choice of voter i can be represented as a function of two properties connecting voter i to party k (her ideological proximity to party k (`prox`) and her `partyID`) and four individual characteristics: `sex` (equals to 1 for female), `age` (1 "18-24 years", 2 "25-34", 3 "35-44", 4 "45-54", 5 "55-64", 6 "65 +"), `education` (0 "up to primary school", 1 "incomplete secondary ", 2 "secondary completed", 3 "post-secondary trade ", 4 "university undergraduate degree inc", 5 "university undergraduate degree comp") and voter's judgment of the incumbent government performance – `gov_perf` (1 "very good job", 2 "good job", 3 "bad job", 4 "very bad job"). We also add an intercept for each party to capture all the other unmeasured non-policy sources of voters' party evaluations, including voters' valence-related evaluations of the parties (such as the judgment related to party leaders' competence, integrity, or charisma).

To estimate individual respondents' vote choice `nopp` calls the `mlogit` package. We run in this first example a Conditional Logit model (we also select the Forza Italia party as the reference alternative – this is optional):

```
> m <- mlogit(vote~prox+partyID | gov_perf+sex+age+education,
       election, reflevel = "FI")
> summary(m)

Call:
mlogit(formula = vote ~ prox + partyID | gov_perf + sex + age +
    education, data = election, reflevel = "FI", method = "nr",
    print.level = 0)

Frequencies of alternatives:
      FI       AN       RC      UDC       UL
0.242009 0.178082 0.100457 0.077626 0.401826

nr method
7 iterations, 0h:0m:0s
g'(-H)^-1g = 0.000248
successive fonction values within tolerance limits

Coefficients :
                  Estimate Std. Error t-value  Pr(>|t|)
AN:(intercept)    0.647570   1.262720  0.5128 0.6080651
RC:(intercept)   -4.927774   2.222642 -2.2171 0.0266177 *
UDC:(intercept) -0.996713   1.375212 -0.7248 0.4685926
UL:(intercept)   -3.668816   1.415796 -2.5913 0.0095602 **
prox              0.080615   0.012912  6.2436 4.276e-10 ***
partyID           4.115841   0.389591 10.5645 < 2.2e-16 ***
AN:gov_perf       0.071072   0.410497  0.1731 0.8625446
RC:gov_perf       1.890192   0.628987  3.0051 0.0026546 **
UDC:gov_perf     -0.054156   0.418196 -0.1295 0.8969623
UL:gov_perf       1.415211   0.405426  3.4907 0.0004818 ***
```

```
AN:sex          -0.621747   0.481125 -1.2923 0.1962613
RC:sex          -1.501041   0.783660 -1.9154 0.0554385 .
UDC:sex         -0.340194   0.490517 -0.6935 0.4879692
UL:sex          -0.260747   0.482777 -0.5401 0.5891295
AN:age          -0.196572   0.156697 -1.2545 0.2096702
RC:age          -0.243988   0.246333 -0.9905 0.3219400
UDC:age          0.251847   0.161889  1.5557 0.1197853
UL:age           0.139813   0.158847  0.8802 0.3787672
AN:education     0.071070   0.188558  0.3769 0.7062364
RC:education    -0.450525   0.314049 -1.4346 0.1514092
UDC:education    0.169480   0.184193  0.9201 0.3575103
UL:education     0.176575   0.182907  0.9654 0.3343541
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-Likelihood: -243.73
McFadden R^2:  0.61524
Likelihood ratio test : chisq = 779.45 (p.value = < 2.22e-16)
```

As can be seen, increasing the ideological proximity between voter i and party k increases the probability to vote for that party. A similar result applies if voter i displays a party identification for party k. On the other hand, as the judgment about the performance of the centre-right incumbent government gets worst, so the chance to vote for any of the opposition parties (relative to FI - the party headed by the incumbent Prime Minister Silvio Berlusconi) increases. Finally given that the constants are all relative to the score of FI, which is normalized to be zero, we can see that there are unmeasured sources of voters' party evaluations that benefit in a statistical significant way FI while penalizing the opposition parties (UL and RC).

Note that by employing `mlogit` we can estimate a large variety of choice models. For example, Conditional Logit has been criticized in the literature because it imposes the independence of irrelevant alternatives (IIA) property on voter choice (meaning that the relative odds of selecting between two parties is independent of the addition or subtraction of other alternatives from the choice set: see Alvarez and Nagler, 1998). Dow and Endersby, (2004) show that for most application the IIA assumption is not as restrictive as it might appear. `mlogit` allows however to run a number of models that relax the above IIA assumptions, such as the heteroscedastic logit model, the general extreme value model, the nested logit model, the multinomial probit model.

Alternatively, we could also decide to run a model with party-varying proximity coefficient, rather than with a single proximity coefficient for all parties, by typing:

```
> m2 <- mlogit(vote~ partyID | gov_perf+sex+age+education | prox,
         election, reflevel = "FI")
> summary(m2)

Call:
mlogit(formula = vote ~ partyID | gov_perf + sex + age + education |
    prox, data = election, reflevel = "FI", method = "nr", print.level = 0)

Frequencies of alternatives:
      FI       AN       RC      UDC       UL
0.242009 0.178082 0.100457 0.077626 0.401826

nr method
7 iterations, 0h:0m:0s
g'(-H)^-1g = 4.43E-05
successive fonction values within tolerance limits

Coefficients :
                 Estimate Std. Error t-value  Pr(>|t|)
AN:(intercept)   0.831976   1.267999  0.6561 0.5117384
RC:(intercept)  -5.335890   2.139541 -2.4939 0.0126333 *
UDC:(intercept) -0.715348   1.400122 -0.5109 0.6094083
UL:(intercept)  -3.125356   1.487789 -2.1007 0.0356698 *
partyID          4.179460   0.395892 10.5571 < 2.2e-16 ***
AN:gov_perf     -0.040633   0.424475 -0.0957 0.9237383
RC:gov_perf      1.758049   0.597427  2.9427 0.0032537 **
UDC:gov_perf    -0.097656   0.432567 -0.2258 0.8213886
UL:gov_perf      1.236409   0.423356  2.9205 0.0034948 **
AN:sex          -0.663719   0.485018 -1.3684 0.1711738
RC:sex          -1.849938   0.788415 -2.3464 0.0189556 *
UDC:sex         -0.452395   0.500718 -0.9035 0.3662654
UL:sex          -0.392115   0.507260 -0.7730 0.4395188
```

```
AN:age          -0.215370   0.158295 -1.3606 0.1736538
RC:age          -0.188365   0.242111 -0.7780 0.4365613
UDC:age          0.269604   0.165212  1.6319 0.1027080
UL:age           0.142644   0.165471  0.8620 0.3886622
AN:education     0.066177   0.190695  0.3470 0.7285693
RC:education    -0.391942   0.309933 -1.2646 0.2060148
UDC:education    0.175535   0.186911  0.9391 0.3476613
UL:education     0.202944   0.190928  1.0629 0.2878115
FI:prox          0.103039   0.033629  3.0640 0.0021839 **
AN:prox          0.061046   0.024028  2.5407 0.0110641 *
RC:prox          0.011895   0.020846  0.5706 0.5682712
UDC:prox         0.158010   0.049688  3.1801 0.0014723 **
UL:prox          0.114600   0.030862  3.7133 0.0002046 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Log-Likelihood: -236.53
McFadden R^2:  0.6266
Likelihood ratio test : chisq = 793.85 (p.value = < 2.22e-16)
```

As can be seen, now the salience of voter's proximity is higher for Ulivo, FI and the centrist UDC, while it is not significant for the radical party RC.

Note that we can also directly estimating a model starting from a wide-format data set by typing

```
> m3 <- mlogit(vote~ prox+partyID  | gov_perf+sex+age+education,
       italy2006, shape="wide", choice="vote",
       varying=c(5:14), sep = "_", reflevel = "FI")
```

All the previous models, regardless of their form, can then be passed to `nopp` to estimate the optimal party positions configuration.

3.2 Second step: estimating party optimal positions

Having estimated empirically all parameters of interest related to the individual respondents' vote choice, now we have to fed them into the MA algorithm. To this aim we have to use the equilibrium command within `nopp`. Starting from previous estimated model `m` this can be easily done by typing:

```
> nash.eq <- equilibrium(model=m, data=election)
```

The mandatory arguments are `model`, which is the `mlogit` model analysis, and `data`, that is the data set (in a long format).

```
> nash.eq

================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
6.408 4.717 6.465 6.291 2.300

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.393 0.183 0.073 0.108
```

`nopp` returns both the optimal party positions as well as the vote-share of parties at that optimal configuration.

The user can also decide to pass to `nopp` a list of "true" party positions (as they arise for example from mass or expert surveys) as well as a list of "true" vote-share of parties (according to survey or to electoral results) and then comparing the equilibrium party configuration with the true one. For example, if we decide to use as the true party positions the survey respondents' mean party placements, and as the true vote-share of parties the vote-share they obtain in the survey, we would type:

```
> true.pos <- list(FI=7.76, UL=3.54, RC=1.91, AN=8.27, UDC=5.99)
> true.votes <- list(FI=.25, UL=.38, RC=.10, AN=.18, UDC=.07)
> nash.eq <- equilibrium(model=m, data=election,
       pos=true.pos, votes=true.votes)
> nash.eq

================
True
================
Party positions:
  FI   UL   AN  UDC   RC
```

```
7.76 3.54 8.27 5.99 1.91

Party shares:
  FI   UL   AN  UDC   RC
0.25 0.38 0.18 0.07 0.10


================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
6.408 4.717 6.465 6.291 2.300

Correlation True/Nash: 0.93
Average Absolute Distance: 1.00

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.393 0.183 0.073 0.108

Correlation True/Nash: 1.00
Average Absolute Distance: 0.67%
```

The results make clear that there are marked similarities between actual and optimal positions (Average Absolute Distance from actual party position: 1.00; R-Pearson: .93%). Besides, the projected vote-share of parties found in the Nash equilibrium looks remarkably close to actual vote-share (around 0.67% of difference as an average). This of course does not imply that all parties are simple-minded vote-maximizers, but still by assuming it we get to a close proxy to the actual, but unknown, utility function deployed by party leaders in the Italian 2006 electoral context.

`nopp` is also extremely flexible with respect to alternative motivations in parties' strategy. For example, two pre-electoral coalitions were competing in 2006 Italian general election: a centre-left coalition (UL and RC) and a centre-right coalition (FI, UDC and AN). In such circumstance we could expect that party's utility from the election is not merely linked to the vote-share it would get, but also, on varying degree, to the electoral success of its own coalition. This could affect party strategy on where to locate on the ideological space during the electoral campaign.

More formally, the utility that party k belonging to coalition c_j attaches to an electoral outcome can be considered as the weighted sum of its expected vote-share (EV_k) and the expected vote of its coalition partners:

$$U_k = EV_k + \alpha \sum_{i=c_j} EV_i \tag{6}$$

where i are the parties – other than k – belonging to coalition c_j (see Adams et al. 2005: 113). When $\alpha = 0$, parties are purely vote-seeking (like in the previous scenario), while if α increases parties start to weight both their own votes and those of their own coalition partners. At the extreme, when $\alpha = 1$, parties weight their own votes exactly as those of their coalition partners. An example:

```
> coal1 <- list(FI=1, UL=2, RC=2, AN=1, UDC=1)
```

With the above line of command, the parties are assigned to two different coalitions.

```
> alpha1 <- list(FI=0.7, UL=0.8, RC=0.1, AN=0.5, UDC=0.5)
```

With this second line of command, the values of α are defined for each party. Note that α can be different across coalitions as well as within the same coalition.

```
> nash.eq <- equilibrium(model=m, data=election,
                   coal=coal1, alpha=alpha1)
> nash.eq

================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
5.581 5.422 6.088 5.603 2.349

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.386 0.186 0.074 0.112
```

Also in this case, we could add to our command a list specifying the “true” party position and the “true” vote-share of parties.

```
> nash.eq <- equilibrium(model=m, data=election,
       pos=c(FI=7.76, UL=3.54, RC=1.91, AN=8.27, UDC=5.99),
       votes=c(FI=.25, UL=.38, RC=.10, AN=.18, UDC=.07),
       coal=coal1, alpha=alpha1)
> nash.eq
================
True
================
Party positions:
  FI   UL   AN  UDC   RC
7.76 3.54 8.27 5.99 1.91

Party shares:
  FI   UL   AN  UDC   RC
0.25 0.38 0.18 0.07 0.10


================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
5.581 5.422 6.088 5.603 2.349

Correlation True/Nash: 0.81
Average Absolute Distance: 1.41

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.386 0.186 0.074 0.112

Correlation True/Nash: 1.00
Average Absolute Distance: 0.69%
```

Compared to previous scenario in which parties are expected to be just vote-maximizer, an equilibrium in which parties care also about the votes obtained by their own coalition (as depicted above) produces equilibria positions that resemble less well the actual positions of Italian parties during the 2006 election (Average Absolute Distance: 1.00 vs. 1.41)[1].

A party k could also be motivated to maximize its margin relative to party j. The probability that parties act as margin-maximizer rather than vote-maximizer can be due for example to the type of electoral system employed (plurality vs. proportional systems). Let's assume that UL is interested to maximize its margin relative to FI. Then we would type:

```
> nash.eq <- equilibrium(model=m, data=election,
         margin=list(UL="FI"))
> nash.eq

================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
6.372 4.937 6.426 6.222 2.311

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.392 0.183 0.073 0.109
```

`margin` is not a symmetric command. Therefore, if both UL and FI are interested to maximize their vote margins relative to each other, we would write:

```
> nash.eq <- equilibrium(model=m, data=election,
         margin=list(FI="UL", UL="FI"))
> nash.eq

================
Nash equilibrium
================
```

[1] In this sense, one could also infer that the existence of pre-electoral coalitions seems to have played no relevant role in the spatial behaviors of parties during the 2006 Italian electoral stage, given that members of either blocs appear not focusing on how their policy strategies influence the collective appeal of their coalition bloc (see Curini and Iacus 2008).

```
Party positions:
   FI    UL    AN   UDC    RC
5.925 4.891 6.495 6.349 2.309

Party shares:
   FI    UL    AN   UDC    RC
0.242 0.392 0.184 0.074 0.109
```

Till now we have imposed no restrictions on party positioning. Therefore, parties are completely free to manipulate their policy images. This is an option that we can however relax in `nopp`. That is, party k can be modeled as being stuck (to a varying degree) to a specific party location for different reasons (fear to lose reputation among electors, uncertainty about popular preferences, activists' concern for established ideology, etc.). When this happens, in the final Nash configuration of party positions, the position of party k can be expressed as: $\texttt{fixed} * (1-\gamma) + \texttt{nash} * \gamma$, where `nash` is the optimal party position when that party is free to move and `fixed` is the "stuck" party position.

When $\gamma = 1$, party's policy choices can span with no spatial limits. When $\gamma = 0$, a party is fixed to the spatial position specified by the user. When $0 < \gamma < 1$, the final position of party k will depend on both considerations (maximizing its vote-shares on one hand and the concern to protect its ideological reputation on the other). For example, let's decide to fix RC party at 1.95 and let's assume that γ is equal to 0.5 for that party. Then we would type:

```
> nash.eq <- equilibrium(model=m, data=election,
             fixed=list(RC=1.95), gamma=0.5)
> nash.eq

================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
6.409 4.726 6.466 6.291 2.094

Party shares:
```

```
   FI    UL    AN   UDC    RC
0.243 0.393 0.183 0.073 0.108
```

Notice that through nopp we can also easily combine different party-motivations in a variety of way. Below we report two hypothetical scenarios:

Example 1:

```
> nash.eq <- equilibrium(model=m, data=election,
          margin=list(FI="UL", UL="FI"),
          fixed=list(RC=1), gamma=0.2)
```

Example 2:

```
> coal1 <- list(FI=1, UL=2, RC=2, AN=1, UDC=1)
> alpha1 <- list(FI=0.7, UL=0.8, RC=0.5, AN=0.5, UDC=0.5)
> nash.eq <- equilibrium(model=m, data=election, coal=coal1,
             alpha=alpha1, fixed=list(RC=1), gamma=0.6)
```

3.3 Linear utility function

In equation (1) we have assumed a quadratic utility function for the voters with respect to the policy component of U_{ik}. An alternative would be to assume a linear utility, i.e.,

$$U_{ik} = -a|x_i - s_k| + \beta t_{ik} + \delta z_i + \epsilon_{ik}. \qquad (7)$$

In the italy2006.lin dataset included in nopp the ideological proximity variable (labelled proxlin_$*$) has been computed as $-|\texttt{self}_i - s_k|$. This variable allows to estimate the empirical model by employing a linear utility function. From the empirical model, we can do the same with respect to the computation of the Nash equilibrium in party locations. Once reshaped in a long format italy2006.lin, we would type two lines of command to run the entire procedure:

```
> data(italy2006.lin)
> election <- mlogit.data(italy2006.lin, shape="wide",
     choice="vote", varying=c(5:14), sep="_")
> m3 <- mlogit(vote~proxlin+partyID | gov_perf+sex+age+education,
       election, reflevel = "FI")
> nash.eq <- equilibrium(model=m3, data=election, quadratic=FALSE)
```

The `FALSE` option after the quadratic option, tells `nopp` to adopt the MA algorithm for linear utility

```
> nash.eq

================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
6.913 4.335 7.140 5.978 2.125

Party shares:
   FI    UL    AN   UDC    RC
0.242 0.402 0.178 0.078 0.100
```

3.4 Adding uncertainty

`nopp` can also estimate the confidence intervals around the optimal party positions (as well as the party vote-share at that optimal configuration) in two different (and mutually excludable) way:

1. By means of a bootstrapping procedure. The observations are resampled and the Nash equilibrium is re-evaluated.

2. By means of a Monte Carlo simulation starting from the empirical estimation. The coefficients of the `mlogit` model are resampled independently from their asymptotic Gaussian distribution and Nash equilibrium is re-evaluated each time. Compared to the former method, this alternative appears as more "Bayesian" in its spirit, given that it takes the estimated coefficients of the `mlogit` model as its priors before simulation.

Note that for each simulation, there is a unique Nash equilibrium that is independent of the randomly generated starting points used in the algorithm for parties' initial placements.

To run a bootstrapping procedure, we just have to add the "`boot`′ option, specifying the number of boostrap replications, as follows:

```
> nash.eq.boot <- equilibrium(model=m, data=election, boot=1000)
```

```
> nash.eq.boot

================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
6.408 4.717 6.465 6.291 2.300

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.393 0.183 0.073 0.108

================
Bootstrap
================
Party positions:
        FI    UL    AN   UDC    RC
mean 6.406 4.711 6.459 6.284 2.295
sd   0.135 0.154 0.151 0.168 0.202

Party shares:
        FI    UL    AN   UDC    RC
mean 0.243 0.393 0.183 0.073 0.107
sd   0.012 0.013 0.010 0.003 0.009

Bootstrap replications: 1000
```

Similarly, to run a Monte Carlo procedure, we need to add the "MC" option, specifying the number of Monte Carlo replications.

```
> nash.eq.mc <- equilibrium(model=m, data=election, MC=1000)

> nash.eq.mc

================
Nash equilibrium
================
Party positions:
```

```
   FI    UL    AN   UDC    RC
6.408 4.717 6.465 6.291 2.300

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.393 0.183 0.073 0.108

================
Monte Carlo
================
Party positions:
        FI    UL    AN   UDC    RC
mean 6.882 4.442 6.906 6.455 2.417
sd   0.678 1.044 1.238 1.070 1.572

Party shares:
        FI    UL    AN   UDC    RC
mean 0.210 0.356 0.189 0.112 0.134
sd   0.072 0.187 0.138 0.136 0.139
```

4 Graphical display of the results

Through `nopp` is also possible to display graphically the results of the analysis with the `plot` command (see Figure 1) taking them out from

```
> nash.eq <- equilibrium(model=m, data=election)
```

and passing it to `plot`

```
> plot(nash.eq)
```

In this case `nopp` reports the equilibrium positions of parties, while the height of each line centered on each party optimal position is proportional to the amount of votes that each party gets in equilibrium.

Had we passed to `nopp` a list of "true" party positions and "true" vote-share of parties, we could also display the optimal party positions/party vote share against the true party positions/party vote share.

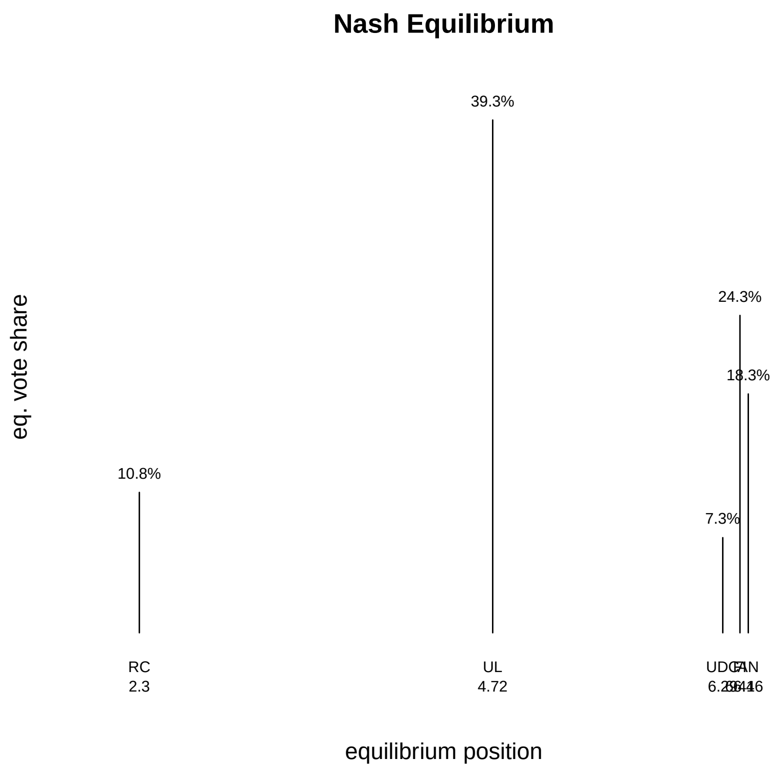

Figure 1: Plot of Nash equilibrium party positions.

```
> true.pos <- list(FI=7.76, UL=3.54, RC=1.91, AN=8.27, UDC=5.99)
> true.votes <- list(FI=.25, UL=.38, RC=.10, AN=.18, UDC=.07)
> nash.eq <- equilibrium(model=m, data=election,
        pos=true.pos, votes=true.votes)
```

and then plot the estimates via

```
> par(mfrow=c(3,1))
> plot(nash.eq)
```

The results are in Figure 2. Alternatively, to get the same graphs we could type:

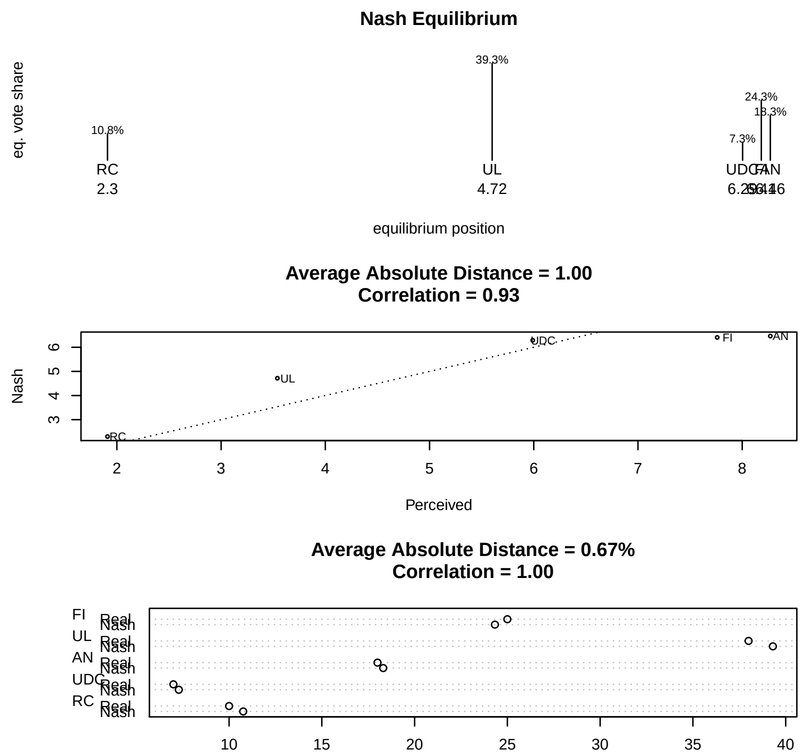

Figure 2: Plot of Nash equilibrium party positions.

```
> nash.eq <- equilibrium(model=m, data=election)
> plot(nash.eq,pos=list(FI=7.76, UL=3.54, RC=1.91, AN=8.27, UDC=5.99))
> plot(nash.eq,votes=list(FI=.25, UL=.38, RC=.10, AN=.18, UDC=.07))
```

After having run a bootstrap procedure, the plot command displays also the equilibrium position of parties with their corresponding confidence interval, as well as a graph contrasting the equilibrium positions as they arise from the normal Nash procedure and from the Bootstrap Nash procedure (see Figure 3) with

```
> plot(nash.eq.boot)
```

The same happens after having estimated a Monte Carlo procedure.

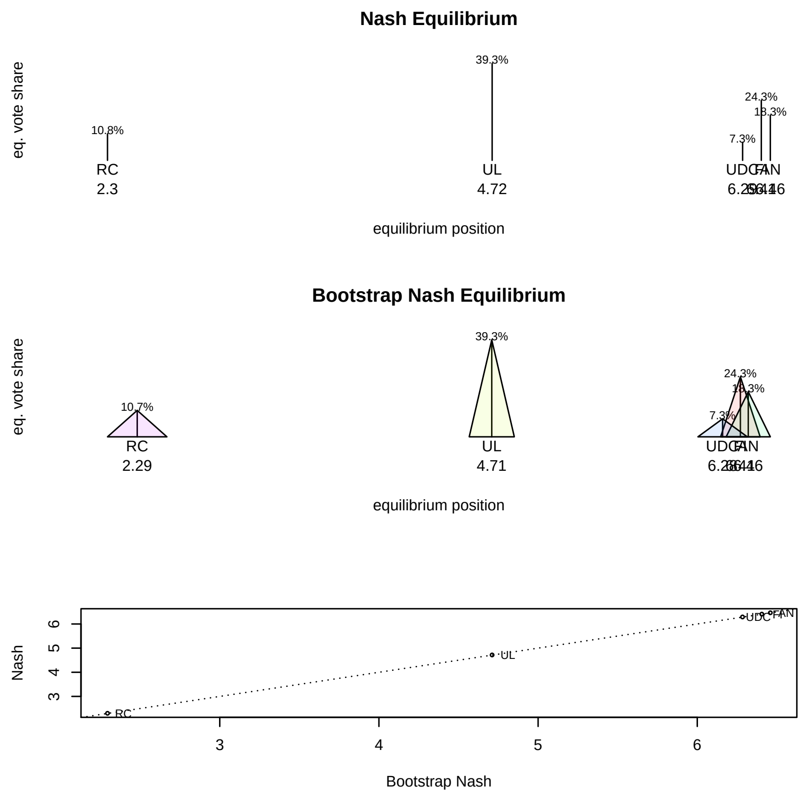

Figure 3: Plot of Nash equilibrium party positions with bootstrap confidence intervals.

5 What's to do in the next release

There could be a number of alternative specifications for U_{ik} that may be used in place of the utility model defined in (1) (a specification that is sometimes called in the literature "proximity unified model". See Adams et al. 2005), including a post-electoral policy preference model (Kedar 2005), a policy discounting model (Adams et al. 2005), a model that mixes directional and proximity theory (Merrill and Grofman 1999), a model in which the utility attached to voting for a party is affected by the expectations over the transmission of votes into representation in government (Calvo and Hellwig 2011), and so on.

Moreover, we must acknowledge the possibility of abstention in voters' choice if neither competitor is sufficiently attractive to merit their support (Hinich and Munger 1994).

In the next release of `nopp` (2.0) we will therefore introduce more flexibility in the specification of voters' utility function. Ideally the researcher should be free to specify (and to estimate) any voters' utility function in a "friendly" environment before passing it to `nopp`.

6 Appendix: preparing a dataset for `nopp`

In this appendix we briefly explain how to build a dataset in R that can be used in `nopp` starting from a format displayed by the majority of electoral survey data available on-line. The dataset that we will use is called *italy.wide* (source: `http://www.cses.org/`).

```
> data(italy2006.wide)
> head(italy2006.wide)

    country id self FI DS AN DL UDC RC pID age sex
1 Italy2006  1    3 NA NA NA NA  NA NA   0   1   0
2 Italy2006  2    4 NA NA NA NA  NA NA   0   4   1
3 Italy2006  3    4 NA NA NA NA  NA NA  23   6   0
4 Italy2006  4    3 NA NA NA NA  NA NA   0   3   1
5 Italy2006  5    4 NA NA NA NA  NA NA  23   5   0
6 Italy2006  6   10  1  9  9  7   6  1   3   4   1
  education vote gov_perf
1         1   UL       NA
2         1   UL        3
3         0   UL        4
4         1   UL        3
5         0   UL        3
6         4   AN       NA
```

where `vote` is a variable that identifies the party voted for by the respondent in the 2006 Italian general election, `self` is the self-placement of respondent on a 0 to 10 left-right scale, `pID` is a variable that identifies the partisanship of the respondent (where 0=stands for no partyID, 1 = FI partyID, 23 = UL partyID, 3 = AN partyID, 4 = UDC partyID, 6 = RC partyID), while the variables from `FI` to `RC` identify the placement of those parties as perceived by the respondent.

As the first step, we estimate the survey respondents' mean of each party placement used as party actual position.

```
> varlist <- c("FI","DS","AN","DL","UDC","RC")
> for(var in varlist)
   assign( sprintf("mean_%s",var),
      mean(italy2006.wide[,var],na.rm=TRUE) )
```

Note that in the 2006 Italian general election the two main parties of the centre-left coalition (DS and DL) presented a joint list for the Chamber of Deputies (under the name of "Ulivo - UL"). Therefore, in our analysis to estimate the placement of UL we just compute the average between DS and DL.

```
> mean_UL <- (mean_DS+mean_DL)/2
> mean_UL

[1] 3.540472
```

To compute the ideological distance between the respondent and each party placement according to a quadratic utility function (i.e., $-(\texttt{self}_i - s_k)^2$), we would type:

```
> varlist <- c("FI","UL","AN","UDC","RC")
> for(var in varlist)
   italy2006.wide[[sprintf("prox_%s",var)]] <-
       -(get(sprintf("mean_%s",var))-italy2006.wide$self)^2
```

On the other hand, to compute the same ideological distance but according to a linear utility function (i.e., $-|\texttt{self}_i - s_k|$.), we would type:

```
> for(var in varlist)
   italy2006.wide[[sprintf("proxlin_%s",var)]] <-
         -abs(get(sprintf("mean_%s",var))-italy2006.wide$self)
> str(italy2006.wide)

'data.frame':	524 obs. of  25 variables:
 $ country    : chr  "Italy2006" "Italy2006" "Italy2006" "Italy2006" ...
```

```
 $ id         : num  1 2 3 4 5 6 7 8 9 10 ...
 $ self       : int  3 4 4 3 4 10 2 0 8 2 ...
 $ FI         : int  NA NA NA NA NA 1 10 0 9 10 ...
 $ DS         : int  NA NA NA NA NA 9 3 3 1 4 ...
 $ AN         : int  NA NA NA NA NA 9 10 10 10 10 ...
 $ DL         : int  NA NA NA NA NA 7 6 7 1 5 ...
 $ UDC        : int  NA NA NA NA NA 6 6 6 NA 5 ...
 $ RC         : int  NA NA NA NA NA 1 2 0 1 0 ...
 $ pID        : int  0 0 23 0 23 3 23 6 1 23 ...
 $ age        : int  1 4 6 3 5 4 6 5 2 1 ...
 $ sex        : int  0 1 0 1 0 1 0 1 1 1 ...
 $ education  : int  1 1 0 1 0 4 4 2 2 4 ...
 $ vote       : Factor w/ 5 levels "FI","UL","AN",..: 2 2 2 2 2 3 2 5 1 2 ...
 $ gov_perf   : int  NA 3 4 3 3 NA 3 4 NA 4 ...
 $ prox_FI    : num  -22.7 -14.2 -14.2 -22.7 -14.2 ...
 $ prox_UL    : num  -0.292 -0.211 -0.211 -0.292 -0.211 ...
 $ prox_AN    : num  -27.8 -18.3 -18.3 -27.8 -18.3 ...
 $ prox_UDC   : num  -8.92 -3.95 -3.95 -8.92 -3.95 ...
 $ prox_RC    : num  -1.18 -4.35 -4.35 -1.18 -4.35 ...
 $ proxlin_FI : num  -4.77 -3.77 -3.77 -4.77 -3.77 ...
 $ proxlin_UL : num  -0.54 -0.46 -0.46 -0.54 -0.46 ...
 $ proxlin_AN : num  -5.28 -4.28 -4.28 -5.28 -4.28 ...
 $ proxlin_UDC: num  -2.99 -1.99 -1.99 -2.99 -1.99 ...
 $ proxlin_RC : num  -1.09 -2.09 -2.09 -1.09 -2.09 ...
```

Eventually we could also compute the quadratic proximity between `self` and the placement of each party as perceived by the same respondent, by typing:

```
> italy2006.wide$UL <- (italy2006.wide$DS+italy2006.wide$DL)/2
> varlist <- c("FI","UL","AN","UDC","RC")
> for(var in varlist)
    italy2006.wide[[sprintf("proxego_%s",var)]] <-
             -(italy2006.wide[[var]]-italy2006.wide$self)^2
```

Finally, we could want to compute a partisanship variable for each party that takes the value of 1 if a respondent is identified with a specific party. In this particular data set, the id which identify each party is a numerical value, so the following procedure is needed to create the `partyID` variables

```
> italy2006.wide$partyID_FI = 1*(italy2006.wide$pID==1)
> italy2006.wide$partyID_UL = 1*(italy2006.wide$pID==23)
> italy2006.wide$partyID_AN = 1*(italy2006.wide$pID==3)
> italy2006.wide$partyID_UDC = 1*(italy2006.wide$pID==4)
> italy2006.wide$partyID_RC = 1*(italy2006.wide$pID==6)
```

Now the dataset is ready to be used in `nopp`

```
> colnames(italy2006.wide)

 [1] "country"     "id"          "self"        "FI"
 [5] "DS"          "AN"          "DL"          "UDC"
 [9] "RC"          "pID"         "age"         "sex"
[13] "education"   "vote"        "gov_perf"    "prox_FI"
[17] "prox_UL"     "prox_AN"     "prox_UDC"    "prox_RC"
[21] "proxlin_FI"  "proxlin_UL"  "proxlin_AN"  "proxlin_UDC"
[25] "proxlin_RC"  "UL"          "proxego_FI"  "proxego_UL"
[29] "proxego_AN"  "proxego_UDC" "proxego_RC"  "partyID_FI"
[33] "partyID_UL"  "partyID_AN"  "partyID_UDC" "partyID_RC"

> election <- mlogit.data(italy2006.wide , shape="wide",
     choice="vote", varying=c(16:25, 27:36), sep="_")
> head(election)

        country id self FI DS AN DL UDC RC pID age sex
1.AN  Italy2006  1    3 NA NA NA NA  NA NA   0   1   0
1.FI  Italy2006  1    3 NA NA NA NA  NA NA   0   1   0
1.RC  Italy2006  1    3 NA NA NA NA  NA NA   0   1   0
1.UDC Italy2006  1    3 NA NA NA NA  NA NA   0   1   0
1.UL  Italy2006  1    3 NA NA NA NA  NA NA   0   1   0
2.AN  Italy2006  2    4 NA NA NA NA  NA NA   0   4   1
      education  vote gov_perf UL alt       prox   proxlin
1.AN          1 FALSE       NA NA  AN -27.825625 -5.275000
1.FI          1 FALSE       NA NA  FI -22.725059 -4.767081
1.RC          1 FALSE       NA NA  RC  -1.177602 -1.085174
1.UDC         1 FALSE       NA NA UDC  -8.923486 -2.987220
1.UL          1  TRUE       NA NA  UL  -0.292110 -0.540472
2.AN          1 FALSE        3 NA  AN -18.275625 -4.275000
      proxego partyID chid
```

```
1.AN       NA       0    1
1.FI       NA       0    1
1.RC       NA       0    1
1.UDC      NA       0    1
1.UL       NA       0    1
2.AN       NA       0    2
```

Note that in the `election` object there are also several missing values. This does not constitute a problem for the estimation of the empirical model. However, the missing values, where present, should be dropped from the dataset before calling the equilibrium routine, in the following way:

```
> m <- mlogit(vote~prox+partyID  | gov_perf+sex,
      election, reflevel = "UL")
> election2 <- election
> idx <- which( is.na(election2$gov_perf) |
     is.na(election2$prox) | is.na(election2$partyID) )
> election2 <- election2[-idx,]
> nash.eq <- equilibrium(model=m, data=election2)
> nash.eq

================
Nash equilibrium
================
Party positions:
   FI    UL    AN   UDC    RC
6.420 4.763 6.378 6.456 2.295

Party shares:
   FI    UL    AN   UDC    RC
0.243 0.393 0.183 0.073 0.108
```

References

[1] Adams J. (1999), Policy divergence in multicandidate probabilistic spatial voting, *Public Choice*, **100**, 103–22

[2] Adams, James F., Samuel Merrill III, and Bernard Grofman (2005). *A Unified Theory of Party Competition.* Cambridge: Cambridge University Press

[3] Alvarez, Michael R., and Jonathan Nagler (1998), When Politics and Models Collide: Estimating Models of Multiparty Elections, *American Journal of Political Science*, **42**: 1349–63

[4] Calvo, Ernesto, and Timothy Hellwig (2011). Centripetal and Centrifugal Incentives under Different Electoral Systems, *American Journal of Political Science*, **55**(1), 27–41

[5] Croissant, Yves (2012), Estimation of multinomial logit models in R : The `mlogit` Package. R package version 0.2-2. URL: `http://cran.r-project.org/web/packages/mlogit/vignettes/mlogit.pdf`

[6] Curini, L., Iacus. S.M. (2008). Italian spatial competition between 2006 and 2008: a changing party system?, paper presented at the XXII Congress of the Italian Political Science Society (SISP), Pavia, 5-8 September. URL: `http://www.socpol.unimi.it/docenti/curini/Papers/Italian%20Spatial%20Competition%202006-2008.pdf`

[7] Dow, Jay K, and James W. Endersby (2004), A Comparison of Conditional Logit and Multinomial Probit Models in Multiparty Elections, *Electoral Studies*, **23**, 107–22

[8] Hinich, Melvin J. and Michael C. Munger (1994), *Ideology and the Theory of Political Choice*, Ann Arbor: Michigan University Press

[9] Kedar, Orit (2005). WhenModerate Voters Prefer Extreme Parties: Policy Balancing in Parliamentary Elections. *American Political Science Review*, **99**(2), 185–99.

[10] Merrill, Samuel III, and James Adams (2001), Computing Nash Equilibria in Probabilistic, Multiparty Spatial Models with Nonpolicy Components, *Political Analysis*, **9**, 347–61

[11] Merrill, Samuel III, and Bernard Grofman (1999), *A Unified Theory of Voting: Directional and Proximity Spatial Models*, Cambridge: Cambridge University Press

[12] Schofield, Normand, and Itai Sened (2006), *Multiparty Democracy: Parties, Elections and Legislative Politics in Multiparty Systems*, Cambridge: Cambridge University Press