MATH0461-2 Introduction to numerical optimization

Faculty of Applied Science

Randomized Condorcet Voting System

Teacher :
Quentin LOUVEAUX

Assistant :
Adrien BOLLAND

Students :
Romain LAMBERMONT, s190931
Arthur LOUIS, s191230

December 8, 2022

Contents

1 Model **1**
1.1 Implementation of the model in linear programming 1
1.2 Application of the RCVS to an example 1
1.3 Discussion of the dual variables and optimal dual basis 2
1.4 Solution of the linear in a linear system 2
1.5 *Bonus* : Comparison of the RCVS with an alternative voting system 2

2 Linear Robust Formulation **3**
2.6 Electors change their votes 3
2.7 Linear Approximation 4
2.8 Implementation as a linear program 5

3 Quadratic Robust Formulation **5**
3.11 SOCP Approximation 5

List of Figures

1 Example of voting graph 1
2 Duels graph 3

List of Tables

1 Lottery probabilities for each candidate 2
2 Dual variables values 2
3 Preferences of voters 3
4 Winning lottery values 3
5 Values of E for some n 4

1 Model

1.1 Implementation of the model in linear programming

In order to compute the winning distribution law of the RCVS, one can implement a linear formulation of the model. The linear formulation is based on the following variables:

- π the voting matrix where $\pi_{i,j}$ represents the results of a duel between the i-th and j-th candidates. The elements of the matrix are computed following this rule : for each voter, if the i-th canditate is ranked higher than the j-th candidate the element $\pi_{i,j}$ is incremented and the element $\pi_{j,i}$ is decremented.
- p the probability vector where p_i represents the probability that the i-th candidate wins.
- e the vector of ones.

The linear formulation of the model is the following:

$$\begin{aligned} &\max_p 1 \\ \text{s.t. } & p^T e = 1 \\ & p \geq 0 \\ & p^T \pi \geq 0 \end{aligned}$$

This formulation was implemented in the file `q1_model.jl` using the `JuMP` and `Gurobi` packages.

1.2 Application of the RCVS to an example

One can apply the Condorcet winning system to the following example where edges goes from loser to winner with relatives voting preferences :

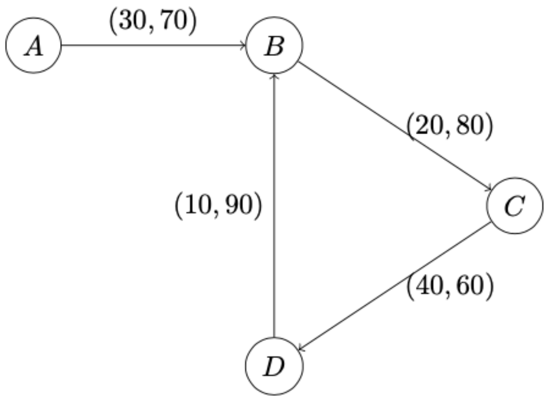

Figure 1: Example of voting graph

In this example, there will no be any Condorcet winner because the graph is not a directed acyclic graph, indeed there is a cycle between the candidates B, C and D. The RCVS is then needed to solve this problem. We simply need to compute the π matrix respecting the graph represented in figure 1 following the rule described in section 1.1:

$$\pi = \begin{pmatrix} 0 & -40 & 0 & 0 \\ 40 & 0 & -60 & 80 \\ 0 & 60 & 0 & -20 \\ 0 & -80 & 20 & 0 \end{pmatrix}$$

When launching the `q1_model.jl` file with this matrix as input, we obtain the following lottery for the RCVS :

Candidate	A	B	C	D
Probability	0.0	0.125	0.5	0.375

Table 1: Lottery probabilities for each candidate

1.3 Discussion of the dual variables and optimal dual basis

By using the following objective function :

$$\max_p 1$$

We ensure that the algorithm will stop at the first iteration it finds a feasible basis and that the five dual variables are equal to 0. We can easily check this affirmation by using the function `solution_summary` and as expected the dual variables values are the following :

Variable	$c_1[1]$	$c_1[2]$	$c_1[3]$	$c_1[4]$	c_2
Value	0	0	0	0	0

Table 2: Dual variables values

This insures that the optimal was found because the dual variables represent the costs necessary to beat a candidate in a election and the optimal solution beats all the candidates. More formally, as our objective function is a constant, the relative costs c^T are always equal to 0 in each election.

1.4 Solution of the linear in a linear system

By considering the following variables :

$$A = \begin{pmatrix} 0 & -40 & 0 & 0 \\ 40 & 0 & -60 & 80 \\ 0 & 60 & 0 & -20 \\ 0 & -80 & 20 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}, \; x = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix}, \; b = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \; d = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{pmatrix}, \; c = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

We can create a system of 9 equations with 9 variables by considering that the 2 vectors of variables will be feasible for the primal and dual problems and optimal by considering :

$$\begin{cases} \left(A_i^T x - b_i\right) d_i = 0, & \forall i \\ x_i \left(c_i - d^T A_i\right) = 0, & \forall i \end{cases}$$

1.5 *Bonus* : Comparison of the RCVS with an alternative voting system

One could compare the Randomized Condorcet Voting System with a much more simpler voting system as the two-round system used in a lot of countries to determine the winner of an election. This system is separated, as the name suggests, in two rounds :

- In the first round, one could use his vote for every possible candidate on the list. The votes are then counted to determine the two candidates with the most votes.
- In the second round, all the voters can vote either for one or the other candidate previously selected in the first round. The winner of the election would be the one with the most votes.

Let there be an election between 3 candidates with 100 voters, the preferences of voters are summarized in the table below :

Preference	Voters
$A > B > C$	30
$A > C > B$	5
$B > A > C$	5
$B > C > A$	25
$C > A > B$	25
$C > B > A$	10

Table 3: Preferences of voters

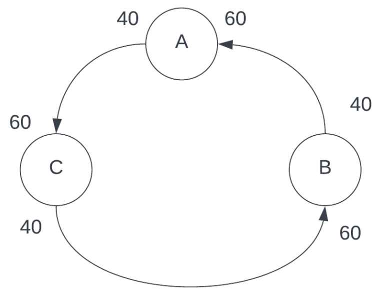

Figure 2: Duels graph

On one hand, if the system used was the two-round system, the two candidates selected in the first round would be A and C and if the preferences remained the same between rounds, the candidate C would be elected with 60% of the votes.

On the other hand, if the system used was the RCVS, we would obtain a Condorcet paradox (a cycle would appear in the duels graph). The resulting lottery values would be :

Candidate	Probability
$p(A)$	0.33
$p(B)$	0.33
$p(C)$	0.33

Table 4: Winning lottery values

2 Linear Robust Formulation

2.6 Electors change their votes

One can compute the number of elections relative to the number of maximum changes as the following formula with n representing the number of maximum changes :

$$E = \sum_{i=0}^{n} 12^i$$

Here are some values for E :

n	0	1	2	3	4
E	1	13	157	1885	22621

Table 5: Values of E for some n

The number 12 comes from the number of possible changes for a system with 4 candidates. If a voter changes his mind, it could result in 12 different modifications of the matrix π. One could create a set of 12 matrices representing the different possibilities :

$$\begin{bmatrix} 0 & \pm2 & 0 & 0 \\ \pm2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & \pm2 & 0 \\ 0 & 0 & 0 & 0 \\ \pm2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & \pm2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \pm2 & 0 & 0 & 0 \end{bmatrix}$$
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \pm2 & 0 \\ 0 & \pm2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \pm2 \\ 0 & 0 & 0 & 0 \\ 0 & \pm2 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \pm2 \\ 0 & 0 & \pm2 & 0 \end{bmatrix}$$

To compute the matrix containing all the possible elections, one simply needs to apply a combination of all the different change matrices. The complete π matrix becomes :

$$\pi = \begin{pmatrix} 0 & -40 & 0 & 0 \\ 40 & 0 & -60 & 80 \\ 0 & 60 & 0 & -20 \\ 0 & -80 & 20 & 0 \\ \\ 0 & -38 & 0 & 0 \\ 38 & 0 & -60 & 80 \\ 0 & 60 & 0 & -20 \\ 0 & -80 & 20 & 0 \\ \\ 0 & -40 & 2 & 0 \\ 40 & 0 & -60 & 80 \\ -2 & 60 & 0 & -20 \\ 0 & -80 & 20 & 0 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{pmatrix}$$

This implementation of the generation of the π matrix is done in the `q2_LRF.jl`. To compute this matrix. One can implement the function `get_change` which takes as inputs `chan`, the matrix containing the 12 possibilities of changes explained earlier and `i`, an index which is decoded into base 12, to assign the correct changes to the initial duel matrix. For example, for $i = 58$ which translates into 4 11 in base 12, means we need to apply the changes 4 and 11 to the initial matrix.

2.7 Linear Approximation

To implement a linear problem in standard form to compute the approximation of the previously discussed problem, one can use a simplex formulation. The objective function is the following :

$$\sum_{i=1}^{E} \left(\mathbb{1} \left(\sum_{j=1}^{4} \left(p^T \times \pi_i\right)_j > 0 \right) \right)$$

This function is the l_0 norm of the vector fill with the l_1 norm of each vector $p^T \times \pi$.

$$\begin{aligned} & \max \quad ||\mathbf{l}||_0 \\ \text{s.t.} \quad & l_i = ||\mathbf{p}^T * \boldsymbol{\pi}_i||_1 \\ & \mathbf{p} * \mathbf{1} \quad \text{where } \mathbf{1} \text{ is a vector of 4 times one} \\ & \boldsymbol{\pi}_i * p \geq 0 \\ & p_i \geq 0 \end{aligned}$$

2.8 Implementation as a linear program

The implementation can be found in the `q2_LRF` file, but we are aware that the results obtained are not the one expected, as the results should be much closer to the original lottery in the table 1.

3 Quadratic Robust Formulation

3.11 SOCP Approximation

To approximate the previous problem, we can use the l_2 norm instead of the l_1 norm. This give us the following standard formula :

$$\begin{aligned} & \max \quad ||\mathbf{l}||_0 \\ \text{s.t.} \quad & l_i = ||\mathbf{p}^T * \boldsymbol{\pi}_i||_2 \\ & \mathbf{p} * \mathbf{1} \quad \text{where } \mathbf{1} \text{ is a vector of 4 times one} \\ & \boldsymbol{\pi}_i * p \geq 0 \\ & p_i \geq 0 \end{aligned}$$

This is a rapid way to make a conic problem.