European Journal of Applied Sciences – Vol. 13, No. 03
Publication Date: May 25, 2025
DOI:10.14738/aivp.1303.18750.
Savadogo, Z. & Pagbelguem, R. (2025). Implementation of the New Metric Procedure of Multidecisions Makers Choice for Use in Large-scale Voting Cases. European Journal of Applied Sciences, Vol - 13(03). 34-46.

Implementation of the New Metric Procedure of Multidecisions Makers Choice for Use in Large-scale Voting Cases

Zoïnabo Savadogo
Joseph KI-ZERBO University

Rasmané Pagbelguem
Joseph KI-ZERBO University

ABSTRACT

The problem of aggregating group preferences into a collective or consensual preference has been tackled since the 18th century by Borda and Condorcet. Since then, several authors have proposed aggregation methods based on social choice theory and metric procedures. However, the use of these metric procedures in decision-making processes today is very limited, and this is explained by the fact that their algorithms recommend multiple complex computations if the number of decision-makers (voters) and the number of alternatives (candidates) is high. To facilitate and popularize the use of metric procedures, it is therefore necessary to equip them with tools or computer programs. It is in this context that we propose in this article a python implementation of the New metric procedure of multidecisions makers choice. In this way, we have been able to set up a program that perfectly integrates the NPMCD algorithm. It is proving to be very efficient, as it allows us to obtain results in a very short time.

Keywords: Social choice theory, Social choice function, Algorithm, Implementation.

INTRODUCTION

Social choice theory is concerned with group decision-making. According to Denis BOUYSSOU [5], it includes the study of voting based on social choice functions. In the literature, there are a large number of voting methods. But many of them produce results that are often controversial and have therefore been widely criticised by certain authors. This is particularly true of the first-past-the-post system with one or two rounds of voting used in the French electoral system. The concern to find a voting system that is participatory, inclusive and whose results reflect popular interest remains a major preoccupation for researchers. In this search for a method that best reflects voters' preferences, Zoïnabo SAVADOGO and al. [14] have developed a voting procedure called the " New metric procedure of multidecisions makers choice ", which is based on approval voting and uses a distance function to aggregate voters' preferences. This New Metric Procedure is highly innovative and has produced very appreciable results. However, its application under certain conditions is very complicated due to the multiple calculations and operations recommended by its algorithm. It is therefore necessary to find tools or calculation programs that allow this method to be used to solve large-scale problems. For this reason, in this paper we propose a computer implementation of this new Metric Procedure in order to broaden its scope and make it more operational, starting with a brief review of the literature on metric procedures. We will then describe the principle of the new metric procedure for

multi-decider choice and propose the implementation of its algorithm. Finally, we will apply the programme to some examples.

STATE OF THE ART

Metric Procedures

Metric procedures are decision-making processes or methods based on calculations of distances or disagreements. In the case of vots, they are used to determine a consensu arrangement from the individual arrangements. Consensus is based on a distance functio that determines a storage unit at the minimum distance from the individual storage units Metric procedures were first introduced in 1962 by Kemeny & Snell [15], and this wa followed by work of several heights, some of which we will bore you with.

Kemeny & Snell Method

Individual storage units are represented in matrix form by $M^{(t)} = \left[m_{ik}^{(t)}\right], t \in T$ (voters set) with

$$m_{ik}^{(t)} = \begin{cases} 1 & \text{if } x_i \succ^{(t)} x_k \\ 1/2 & \text{if } x_i \approx^{(t)} x_k \\ 0 & \text{if } x_k \succ^{(t)} x_i \end{cases}$$

Aggregation involves determining a matrix $B = [b_{ik}]$ which is the most representative of transitive matrices $M^{(t)} = \left[m_{ik}^{(t)}\right], t \in T$. Consensus is achieved by minimising the number of disagreements between B et $M^{(t)}$ i.e.:

$$\min\left\{\sum_{t=1}^{s}\ \sum_{i=1}^{m}\ \sum_{k=1}^{m}\ \left|b_{ik} - m_{ik}^{(t)}\right| : B \in WO\right\} = \left\{\min \sum_{t=1}^{s}\ \delta_{KS}\left(B, M^{(t)}\right) : B \in WO\right\} \quad (1)$$

where WO is the set of complete pre-orders. The optimal matrix B* corresponds to the matrix representation of a complete pre-order. This optimisation problem cannot be solved by classical mathematical programming methods because of the transitivity of the matrix B; but various solution algorithms have been proposed by Leese, Norman and Cook & Saipe (see [15] and [10]).

The Cook & Seiford Method

The Cook & Seiford model ([15]) calculates the distance between two ranks by taking the difference between the ranks of the shares. Consensus storage $R^* = (r_1^* \ldots r_m^*)$ is the one that minimises differences $|r_i^t - r_i^*|, t \in T$ i.e.:

$$\min\{\sum_{t=1}^{s}\ \sum_{i=1}^{m}\ |r_i^t - r_i^*| : (R^* \in SO), (R^t \in WO)\} \quad (2)$$

Note that the various previous codifications were developed with the sole aim of finding a way of solving these minimum distance problems, because they are mathematical problems with integer variables, which as we know are trickier to solve than problems with continuous variables, but also because the constraints relating to transitivity make them difficult to solve.

The Blin Method

In 1976, Blin proposed a model (see [?]) which consists of determining a linear consensual arrangement from linear individual arrangements (strict order relations). The rankings are represented in matrix form:

$$P^t = \left[p_{ir}^{(t)}\right], t \in T, \text{ with}$$

$$p_{ik}^{(t)} = \begin{cases} 1 & \text{if } x_i \text{ occupies the rank } r \in \{1; 3/2; 2; \dots, m-1; m-1/2; m\} \\ 0 & \text{sinon} \end{cases}$$

This representation is not limited to strict order relations, and can be extended to total pre-orders (Amstrong & a1, [4]). Blin's approach consists in determining a consensus of arrangements noted here $Q = [q_{it}]$ and showing the minimum of disagreements with the matrix: $P^t = [p_{ir}^t], t \in T$. So, we need to solve:

$$\min\{\sum_{t=1}^{s} \sum_{i=1}^{m} \sum_{k=1}^{m} |q_{ik} - p_{ik}^t| : Q \in SO\} \quad (3)$$

With:
⋄SO: The set of relations of strict order,⋄ s: all decision-makers or voters, ⋄ m: is the number of candidates.

Noting:

$$\delta_{BL}(Q, P^t) = \sum_{i=1}^{m} \sum_{k=1}^{m} |q_{ik} - p_{ik}^t| \quad (4)$$

The problem is written as:

$$\min\{\sum_{t=1}^{s} \delta_{BL}(Q, P^t) : Q \in SO\} \quad (5)$$

The problem is thus transformed into an assignment problem whose optimal solution is obtained by applying a classic solution algorithm: the Hungarian method.

DESCRIPTION OF THE NEW METRIC PROCEDURE OF MULTIDECISIONS MAKERS CHOICE

Notions

New metric procedure of multidecisions makers choice (NMPMD) was proposed by Zoïnabo Savadogo et al in 2020 [14]. This procedure uses the approval voting system but with a multi-choice preference with four levels of indifference. The idea is to rank the candidates, with the possibility of having more than two candidates in the same class.

In this method,

- Individual storage units are represented in matrix form by:

$$M^t = \left[m_{ir}^{(t)}\right], t \in T \text{ (voters set) with:}$$

$$m_{ir}^{t} = \begin{cases} 1 & \text{if } c_i \text{ is at } 1^{\text{st}} \text{ choice} \\ 1/2 & \text{if } c_i \text{ is at } 2^{\text{nd}} \text{ choice} \\ 0 & \text{if } c_i \text{ is at } 3^{\text{rd}} \text{ choice} \\ -1/2 & \text{if } c_i \text{ is at } 4^{\text{th}} \text{ choice} \\ -1 & \text{elsewhere} \end{cases} \quad \text{(E)}$$

- Minimum disagreement distances are calculated in the same way as in Blin's method.

In other words:

$$\sum_{t=1}^{s} \delta_{BL}(Q, M^t) : Q \in SO \quad (6)$$

- The best choice (solution) is the arrangement that allows the minimum distance of disagreement, i.e. $\min\{\sum_{t=1}^{s} \delta_{BL}(Q, M^t) : Q \in SO\}$

Didactic Example

To explain the various stages of the method, let's consider the case of five candidates and three voters whose choices are as follows:

Table 1: Didactic example

	1st choice	**2nd choice**	**3rd choice**	**4th choice**
$R^{(1)}$	$\{c_2, c_3\}$	$\{c_1\}$	$\{c_4\}$	$\{c_5\}$
$R^{(2)}$	$\{c_2\}$	$\{c_1\}$	$\{c_4\}$	$\{c_3, c_5\}$
$R^{(3)}$	$\{c_1\}$	$\{c_3\}$	$\{c_2, c_4\}$	$\{c_5\}$

(a) Step 1: Storage Matrices:

Applying the scale defined above, the storage matrices for these three voters are respectively:

$$M^{(1)} = \begin{pmatrix} \text{class} & 1^{\text{st}} \text{ choice} & 2^{\text{nd}} \text{ choice} & 3^{\text{rd}} \text{ choice} & 4^{\text{th}} \text{ choice} \\ c_1 & -1 & 1/2 & -1 & -1 \\ c_2 & 1 & -1 & -1 & -1 \\ c_3 & 1 & -1 & -1 & -1 \\ c_4 & -1 & -1 & 0 & -1 \\ c_5 & -1 & -1 & -1 & -1/2 \end{pmatrix}$$

$$M^{(2)} = \begin{pmatrix} \text{class} & 1^{\text{st}} \text{ choice} & 2^{\text{nd}} \text{ choice} & 3^{\text{rd}} \text{ choice} & 4^{\text{th}} \text{ choice} \\ c_1 & -1 & 1/2 & -1 & -1 \\ c_2 & 1 & -1 & -1 & -1 \\ c_3 & -1 & -1 & -1 & -1/2 \\ c_4 & -1 & -1 & 0 & -1 \\ c_5 & -1 & -1 & -1 & -1/2 \end{pmatrix}$$

$$M^{(3)} = \begin{pmatrix} \text{class} & 1^{\text{st}}\ \text{choice} & 2^{\text{nd}}\ \text{choice} & 3^{\text{rd}}\ \text{choice} & 4^{\text{th}}\ \text{choice} \\ c_1 & 1 & -1 & -1 & -1 \\ c_2 & -1 & -1 & 0 & -1 \\ c_3 & -1 & 1/2 & -1 & -1 \\ c_4 & -1 & -1 & 0 & -1 \\ c_5 & -1 & -1 & -1 & -1/2 \end{pmatrix}$$

(b) Step 2: Set of Possible Choices:

We determine all the possible arrangements a voter can make with these five candidates. This involves generating all the p -lists (here $p = 5$) of the list: {1,2,3,4} where 1 denotes 1^{st} choice, 2 denotes 2^{th} choice, etc. The set of possible choices is a list containing all these rankings. Generally speaking, its cardinal is $4^{\text{numberof candidates}}$. In our example, we have $4^5 = 1024$ possible choices. The overview of the matrix form of this list is:

$$SO = \begin{pmatrix} c_1 & c_2 & c_3 & c_4 & c_5 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 2 \\ & & & & \\ & . & . & . & . \\ 4 & 4 & 4 & 4 & 4 \end{pmatrix}$$

(c) Étape 3: Matrice de Stockage des Choix Possibles:

For each element in the set of possible choices (which corresponds to a possible arrangement a voter can make), we generate the corresponding storage matrix using the scale (E).

For example, for the first element [11111] we have the following matrix:

$$\begin{pmatrix} \text{class} & 1^{\text{st}}\ \text{choice} & 2^{\text{nd}}\ \text{choice} & 3^{\text{rd}}\ \text{choice} & 4^{\text{th}}\ \text{choice} \\ c_1 & 1 & -1 & -1 & -1 \\ c_2 & 1 & -1 & -1 & -1 \\ c_3 & 1 & -1 & -1 & -1 \\ c_4 & 1 & -1 & -1 & -1 \\ c_5 & 1 & -1 & -1 & -1 \end{pmatrix}$$

And for the second element [11112], we have the following matrix:

$$\begin{pmatrix} \text{class} & 1^{\text{er}}\ \text{choix} & 2^{\text{eme}}\ \text{choix} & 3^{\text{eme}}\ \text{choix} & 4^{\text{eme}}\ \text{choix} \\ c_1 & 1 & -1 & -1 & -1 \\ c_2 & 1 & -1 & -1 & -1 \\ c_3 & 1 & -1 & -1 & -1 \\ c_4 & 1 & -1 & -1 & -1 \\ c_5 & -1 & 1/2 & -1 & -1 \end{pmatrix}$$

And so on.

We then have 1024 storage matrices corresponding to the 1024 possible choices.

(d) Step 4: Calculating Disagreements (delta Blin):

For each storage Q of the sete SO, we calculate the detuning, also known as delta Blin (δ_{BL}) which is the distance of the symmetrical difference between this matrix and each of the voter storage matrices.

The distance of the symmetrical difference between two storage matrices $M^{(1)}$ and $M^{(2)}$ is calculated by the following formula:

$$\delta_{BL}\left(M^{(1)}, M^{(2)}\right) = \sum_{i=1}^{m} \sum_{k=1}^{m} \left|M_{ik}^{(1)} - M_{ik}^{(2)}\right| \quad (7)$$

m being the number of candidates.

For example, considering the storage matrices of the three voters $R^{(1)}, R^{(2)}$ and $R^{(3)}$ above we have:

$$\begin{aligned}\delta_{BL}\left(R^{(1)}, R^{(2)}\right) &= \sum_{i=1}^{m} \sum_{k=1}^{m} \left|M_{ik}^{(1)} - M_{ik}^{(2)}\right| \\ = \; & |-1+1| + |1/2 - 1/2| + |-1+1| + |-1+1| \\ & + |1-1| + |-1+1| + |-1+1| + |-1+1| \\ & + |1+1| + |-1+1| + |-1+1| + |-1+1/2| \\ & + |-1+1| + |-1+1| + |0-0| + |-1+1| \\ & + |-1+1| + |-1+1| + |-1+1| + \left|-\frac{1}{2} + \frac{1}{2}\right| = 2{,}5\end{aligned}$$

In the same way we have:

$$\begin{aligned}\delta_{BL}\left(R^{(1)}, R^{(3)}\right) &= \sum_{i=1}^{m} \sum_{k=1}^{m} \left|M_{ik}^{(1)} - M_{ik}^{(3)}\right| \\ &= 10\end{aligned}$$

and

$$\begin{aligned}\delta\left(R^{(2)}, R^{(3)}\right) &= \sum_{i=1}^{m} \sum_{k=1}^{m} \left|M_{ik}^{(2)} - M_{ik}^{(3)}\right| \\ &= 8{,}5\end{aligned}$$

Or to determine the disagreement δ_{BL} of the three voters on the choice of arrangement:

$$Q = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

We have: $\delta_{BL}\left(Q, M^{(t)}\right) = \sum_{i=1}^{m} \sum_{k=1}^{m} \left|q_{ik} - m_{ik}^{(t)}\right|$, with ici $t = 1,2,3$.

$$\Rightarrow \delta_{BL}\left(Q, M^{(t)}\right) = \delta\left(Q, M^{(1)}\right) + \delta\left(Q, M^{(2)}\right) + \delta\left(Q, M^{(3)}\right)$$

$$\begin{aligned}\delta_{BL}\left(Q, M^{(1)}\right) = \quad & |1+1| + |-1-1/2| + |-1+1| + |-1+1| \\ & +|1-1| + |-1+1| + |-1+1| + |-1+1| \\ & +|1-1| + |-1+1| + |-1+1| + |-1+1| \\ & +|1+1| + |-1+1| + |-1-0| + |-1+1| \\ & +|1+1| + |-1+1| + |-1+1| + |-1+1/2| \\ & = 9\end{aligned}$$

$$\begin{aligned}\delta_{BL}\left(Q, M^{(2)}\right) = \quad & |1+1| + |-1-1/2| + |-1+1| + |-1+1| \\ & +|1-1| + |-1+1| + |-1+1| + |-1+1| \\ & +|1+1| + |-1+1| + |-1+1| + |-1+1/2| \\ & +|1+1| + |-1+1| + |-1-0| + |-1+1| \\ & +|1+1| + |-1+1| + |-1+1| + |-1+1/2| \\ & = 11{,}5\end{aligned}$$

$$\begin{aligned}\delta_{BL}\left(Q, M^{(3)}\right) = & |1-1| + |-1+1| + |-1+1| + |-1+1| \\ & +|1+1| + |-1+1| + |-1-0| + |-1+1| \\ & +|1+1| + |-1-1/2| + |-1+1| + |-1+1| \\ & +|1+1| + |-1+1| + |-1-0| + |-1+1| \\ & +|1+1| + |-1+1| + |-1+1| + |-1+1/2| \\ & = 12\end{aligned}$$

Hence $\delta_{BL}\left(Q, M^{(t)}\right) = 32{,}5$ .So for the 1024 possible choices we will also have 1024 disagreements.

(e) Step 5: Solution:

Once the disagreements are known, we identify the choice with the fewest disagreements. This choice constitutes a ranking of the candidates on which the three voters had the least disagreement. It is therefore a consensual ranking, since it is a result approved by a majority of voters. The candidate with the highest ranking is then the winner. Continuing the calculations with our python program, we see that for this example, the best choice is: $\begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 \\ 2 & 1 & 4 & 3 & 4 \end{bmatrix}$ with a disagreement of 11. The winner is candidate c_2.

ALGORITHMIC IMPLEMENTATION OF THE NEW METRIC PROCEDURE OF MULTIDECISIONS MAKERS CHOICE

Procedure Algorithm

- *Initialization* on input, we need a data table showing the ranking of candidates by each voter. The data (values) in this table form a matrix that we call Data.
- *Storage Matrix:* The choice of each voter is converted into a storage matrix: $M = [m_{ir}]$ with:

$$m_{ir} = \begin{cases} 1 & \text{if } c_i \text{ is at } 1^{\text{st}} \text{ choice} \\ 1/2 & \text{if } c_i \text{ is at } 2^{\text{nd}} \text{ choice} \\ 0 & \text{si } c_i \text{ is at } 3^{rd} \text{ choice} \\ -1/2 & \text{si } c_i \text{ is at } 4^{\text{th}} \text{ choice} \\ -1 & \text{elsewhere} \end{cases} \quad \text{(E)}$$

We denote MS the list of all storage matrices.

- *Set of possible choices:* Generate all the possible arrangements a voter can make. This can easily be done using the product function in the Python library (version 3 or version 5). We call ensChoix the list of all possible choices.
- *Matrix of possible choices:* For every $i \in$ ensChoix, we generate a storage matrix Q_i. All these matrices are then stored in a set (list) noted SO.
- *set of disagreements:* Determine the set of disagreements deltablin which is:

$$\{\sum_{t=1}^{s}\sum_{i=1}^{m}\sum_{k=1}^{m}|q_{ik} - p_{ik}^{t}|: Q \in SO\}$$

- *Solution:* Identify the choice whose storage matrix $Q \in SO$ results in the fewest disagreements ($min\{$ *deltablin* $\}$).

Python Implementation of the Procedure

Implementing an algorithm involves writing it down in a machine-executable version using a programming language, i.e. writing a program. We have used the Python programming language to program our algorithm. This choice is justified by the fact that Python is a powerful and easy-to-learn programming language. It features high-level data structures and allows a simple but effective approach. Python is also ideal for scripting and rapid application development in many fields and on most platforms.

Pseudo Code:

The pseudo code of an algorithm is its writing in natural language, i.e. without recourse to a given programming language. The pseudo code of the NMPMD algorithm is as follows:

Algorithm 1 NMPMD

```
Inputs : Data Matrix showing voters' preferences V_i about the candidates c_j
  List : MS, ensChoix, SO, delta_BL empty
  variables number of voters  (n), number of candidates  (m): wholes, k: real
    For i from 1 to n Do
      MS← matrice_stockage (Data)
    End For
  rank =[1,2,3,4]
  ensChoix ← product(rank, repeat =4)
    For Choice in ensChoix Do
      SO ← matrice_stockage (Choice)
    End For
    For i from 1 to size of ensChoix Do
      dist_BL ←  0
          For j from 1 to size of MS Do
            dist_BL ←  \sum_{i=1}^m \sum_{k=1}^m|S O[i]-M S[j]|
          End For
        Add dist_BL to delta_BL
      End For
    k ← \min (delta_B L)
    If dist_BL=k Then
        bestchoice ← S O[i]
      End If
```

Output : bestchoice

The Python code for the NMPMD algorithm is as follows:

Fonction 2: ***Function used to generate storage matrices***

```
This function takes the "initial array" matrix as a parameter and returns
the list of MS storage matrices
def storage_matrix(Matrix):
   Nb_voters, Nb_candidates = len(Data), len(Data[0]|
   MS = []
   M = np.zeros((Nb_candidates,4))-1
   for i in range(Nb_voters):
      for j in range(Nb_candidates):
         if Data[i,j] == 1:
            M[j,0] = 1
         if Data[i,j] == 2:
            M[j,1] = 1/2
         if Data[i,j] == 3:
            M[j,2] = 0
         if Data[i,j] == 4:
            M[j,3] = -1/2
      MS.append(M)
      M = np.zeros((Nb_candidates,4))-1
   return MS
MS=storage_matrix(Data)
```

Fonction 3 ***Function for generating possible choices***

```
This function (product) is imported from the Python version 5 library.
Nb _candidates =len(Data[0])
m=Nb _candidates
rang=[1, 2, 3,4]
ensChoix=np.array(list([i for i in itertools.product(rang,repeat=m)]))
```

Fonction 4: ***Function for generating matrices of possible choices***

```
def choice_matrix(Matrix):
   Nb_candidates =len(Data[0])
   SO = []
   Q = np.zeros((Nb_candidates,4))-1
   for i in range(len(ensChoix)):
      for j in range(len(ensChoix[0])):
         if ensChoix[i,j] == 1:
            Q[j,0] = 1
         if ensChoix[i,j] == 2:
            Q[j,1] = 1/2
         if ensChoix[i,j] == 3:
            Q[j,2] = 0
```

```
        if ensChoix[i,j] == 4:
          Q[j,3] = -1/2
      SO.append(Q)
      Q = np.zeros((Nb_candidates,4))-1
    return SO
SO=choice_matrix(ensChoix)
```

Fonction 5 ***Function for calculating disagreements and identifying the best choice (solution)***

```
def deltablin (SO, MS):
  SO=choice_matrix(ensChoix)
  MS=storage_matrix (Data)
  delta_BL=[]
  for i in range(len(ensChoix)):
    dist_BL=0
    for j in range(len (MS)):
      dist_BL+=sum(abs((SO[i])-MS[j]))
    delta_BL.append(sum(dist_BL))
  return delta_BL
sol=deltablin(SO,MS)
K=min(sol)
bestChoice=ensChoix[sol.index(K),:]
matbestChoice=SO[sol.index(min(sol))]
```

APPLICATIONS

Example 1: 4 Candidates and 3 Voters

The following table shows the choices made by voters:

Table 2: Preference of 3 voters out of 4 candidates

	1st choice	2nd choice	3rd choice	4th choice
$V^{(1)}$	$\{c_1\}$	$\{c_2\}$	$\{c_3\}$	$\{c_4\}$
$V^{(2)}$	}	$\{c_1, c_2\}$	}	$\{c_3, c_4\}$
$V^{(3)}$	$\{c_4\}$	$\{c_1\}$	$\{c_3\}$	$\{c_2\}$

After modelling, we obtain the following table:

Table 3: Preference of 3 voters out of 4 candidates

	c_1	c_2	c_3	c_4
$V^{(1)}$	1	2	3	4
$V^{(2)}$	2	2	4	4
$V^{(3)}$	2	4	3	1

By entering this data into our program, we obtain the following results, which are displayed on the console.

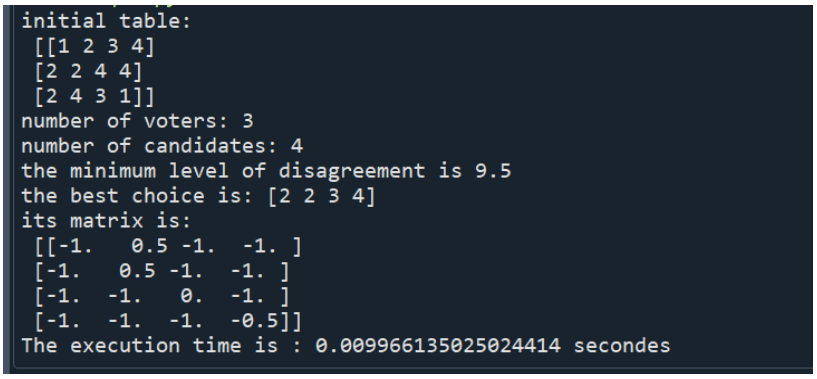

Fig 1: Results for example 1

So the winners are candidate c1 and candidate c2

Example 2: 6 Candidates and 5 Voters

Table 4: Preference of 5 voters out of 6 candidates

	1st choice	2nd choice	3rd choice	4th choice
$V^{(1)}$	$\{c_1, c_2\}$	$\{c_6\}$	$\{c_3, c_4\}$	$\{c_5\}$
$V^{(2)}$	}	$\{c_3\}$	$\{c_1, c_2\}$	$\{c_4, c_5, c_6\}$
$V^{(3)}$	$\{c_3, c_5\}$	$\{c_1, c_6\}$	$\{c_2\}$	$\{c_4\}$
$V^{(4)}$	$\{c_1\}$	$\{c_2\}$	$\{c_3, c_6\}$	$\{c_4, c_5\}$
$V^{(5)}$	$\{c_1, c_6\}$	$\{c_2\}$	$\{c_3\}$	$\{c_4, c_5\}$

After modelling, we obtain the following table:

Table 5: Preference of 5 voters out of 6 candidates

	c_1	c_2	c_3	c_4	c_5	c_6
$V^{(1)}$	1	1	3	3	4	2
$V^{(2)}$	3	3	2	4	4	4
$V^{(3)}$	2	3	1	4	1	2
$V^{(4)}$	1	2	3	4	4	3
$V^{(2)}$	1	2	3	4	4	1

By entering this data into our program, we obtain the following results, which are displayed on the console:

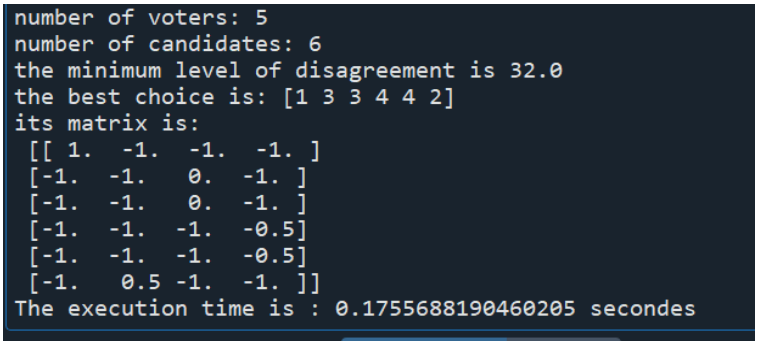

Fig 2: Results of example 2

The winner is candidate c1

Example 3: 5 Candidates and 20 Voters

Table 5: Preferences of 20 voters on 5 candidates

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9	V_{10}	V_1	V_{12}	V_{13}	V_{14}	V_{15}	V_{16}	V_{17}	V_{18}	V_{19}	V_{20}
c_1	3	1	4	2	2	3	2	3	4	4	1	1	1	2	2	1	1	4	1	4
c_2	3	1	4	3	4	1	2	1	1	3	2	2	2	2	4	2	1	2	1	1
c_3	1	2	2	4	4	2	3	1	2	4	2	4	4	2	4	4	4	2	4	2
c_4	4	2	4	1	4	4	4	3	2	3	3	1	4	3	1	4	1	2	1	1
c_5	1	4	2	1	1	1	1	4	2	2	2	2	2	1	4	4	4	1	1	1

Integrating this table into the program gives the following results at the console:

```
number of voters: 20
number of candidates: 5
the minimum level of disagreement is 150.5
the best choice is: [4 4 4 4 1]
its matrix is:
 [[-1.  -1.  -1.  -0.5]
 [-1.  -1.  -1.  -0.5]
 [-1.  -1.  -1.  -0.5]
 [-1.  -1.  -1.  -0.5]
 [ 1.  -1.  -1.  -1. ]]
The execution time is : 0.1388380527496338 secondes
```

Fig 3: results of example 3

The winner is candidate c5

CONCLUSION

The New metric procedure of multidecisions makers choice uses a distance function as an aggregation function, more precisely the Blin distance, which calculates the disagreement of voters on a given choice. Calculating these disagreements requires several operations and the manipulation of a large number of matrices, especially if the number of candidates and voters is high. The implementation we propose here removes this difficulty and extends the scope of application of this method. Applying this program to the examples discussed in [14] gives the same results in most cases in less than a second. However, certain difficulties associated with adapting the data structures into arrays or matrices that can be run on Python did not make the task any easier.

References

[1] A. Baujard, R. Blanch, S. Bouveret, H. Igersheim, A. Laruelle, J.-F. Laslier and I. Lebon, Report on the experiment & VOTER AUTREMENT >during the first round of the French presidential election on 23 April 2017 in Allevardles-Bains, Crolles, Grenoble, Hérouville-Saint-Clair, Strasbourg and on the internet. Working Papers (2017). University of Lyon, Groupe d'Analyse et de Théorie Economique Lyon St-Étienne (GATE Lyon St-Étienne), 48.

[2] Borda, J.-Ch., Mémoire sur les élections au scrutin, Memoirs of Académic Sciences, Paris 1785.

[3] Condorcet, M. J. A. M. Caritat, Marquis D, Essay on the application of analysis to the probability of decisions rendered the plurality of ways, Imprimerie Royale, Paris 1785.

[4] Denis Bouyssou, Thierry Marchant, Marc Pirlot, Patrice Perny, Alexis Tsoukiàs, and Philippe Vincke. Evaluation and decision models: a critical perspective. Kluwer Academic Publishers, 2000

[5] Hatem Smaoui, Dominique Lepelley,2013, The three-tier note voting system: Study of a new voting method. Review of Political Economy, 2021 (Vol.131)

[6] Jacques Teghem. Operational Research, volume 2: Production management, Random models, Multicriteria support. Ellipses, Bruxelles, 2012.

[7] J. P. Barthélemy, Axiomatic characterisation of the distance of the symmetrical difference between binary relations, Mathematics and Humanities 17 (1979), 85-113.

[8] J. G. Kemeny, Mathematics without numbers, Daedalus 88(4) (1959), 577-591. [20] W. D. Cook and L. M. Seiford, Priority ranking and consensus formation, Management Science 24 (1978), 1721-1 732.

[9] Pomerol, J.-Ch. et Barba-Romero, S. (1993), Multi-criteria choice in business, Hermès, Paris.

[10] R. D. Armstrong, W. D. Cook and L. M. Seiford. Priority ranking and consensus formation, Management Science 28 (1982), 638-648.

[11] Ruffin-Benoît M. Ngoie. Social choice and fair sharing: a mathematical a posteriori analysis of the 2006 and 2011 presidential and legislative elections in the DRC. Master's thesis, National Pedagogical University, Kinshasa, 2012. http://mpra.ub.uni-muenchen.de/64915/.

[12] S. Durand. On some paradoxes in social choice theory and decision support. Doctoral thesis (2000). Joseph-Fourier University, Grenoble1, 148.

[13] SLIM Ben Khélifa. Multicriteria support for group decision-making: the sythesis outranking approach, Thèse, LAVAL UNIVERSITY Quebec Avril 1998.

[14] Zoinabo Savadogo, Pegdwindé Ousséni Fabrice Ouedraogo, Kounhinir Some, Ousséni So, Berthold Ulungu and Blaise Somé. New metric procedure of multidecisions makers choice, Far East J. Appl. Math. 95(5) (2016), 329-341

[15] Zoinabo Savadogo et Blaise Somé. Voting methode based on approval voting and aritmetic mean. International Journal of Applied Mathematical Research (2020)