Majority Judgment over a Convex Candidate Space

Chiwei Yan[a,b], Prem Swaroop[d], Michael O. Ball[d], Cynthia Barnhart[a,c], Vikrant Vaze[e]

[a]*Operations Research Center, Massachusetts Institute of Technology, USA*
[b]*Department of Industrial and Systems Engineering, University of Washington, USA*
[c]*Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, USA*
[d]*Robert H. Smith School of Business and Institute for Systems Research, University of Maryland, USA*
[e]*Thayer School of Engineering, Dartmouth College, USA*

Abstract

Most voting methods can only deal with a finite number of candidates. In practice, there are important voting applications where the candidate space is continuous. We describe a new voting method by extending the Majority Judgment voting and ranking method to handle a continuous candidate space which is modeled as a convex set. We characterize the structure of the winner determination problem and present a practical iterative voting procedure for finding a (or the) winner when voter preferences are unknown.

Keywords: Voting, Majority Judgment, Mixed-Integer Convex Optimization, Computational Social Choice

1. Introduction

Voting is concerned with aggregating evaluations over a multitude of voters, in ways that the final outcome has appeal to a large cross-section of the decision-makers. Over centuries, investigators seeking a fool-proof voting system have been riddled by a challenge characterized in Arrow's Impossibility Theorem (Arrow, 1951). Majority Judgment (MJ) (Balinski and Laraki, 2010, 2014) involves grading — instead of preference ranking — of each candidate. With this richer preference elicitation, MJ, in a sense, bypasses Arrow's Impossibility Theorem. The method also enjoys enhanced strategy-proofness due to its median-seeking criterion.

MJ has already been practiced in many contests and juries around the world, as well as in a few political elections (Balinski and Laraki, 2011, 2010). It takes as input the "grades" given by the voters, and produces a *majority grade* for each candidate as an output. The majority grade can be used to select a winner (called a *majority winner*) and/or compute rank-orderings (called *majority ranking*). If voters have equal weights, then the majority grade of a candidate is the highest grade such that an absolute majority of the voters grade that candidate at least as highly as that grade. If we order voters' grades in a descending order, in the case of an odd number of voters, it is the median of the grades; if there are even number of voters, then it is the lower of the two middlemost grades. On the other hand, if voters have unequal weights, then the majority grade is the highest grade such that a subset of voters with an absolute majority of the total weight (i.e., the sum of their weights being strictly greater than half of the total weight) grade that candidate at least as highly as that grade.

We illustrate MJ using the example in Table 1. Suppose there are six voters with equal weights, voting on three candidates: C_1, C_2, C_3. Each voter assigns to each candidate one of these five grades: 5 (Excellent), 4 (Very Good), 3 (Good), 2 (Passable), 1 (Reject). The grades thus obtained for each candidate by voting are then sorted from best to worst, as given in Table 1. The majority grade for each candidate (marked "Majority Grade") is the fourth grade from the top, because an absolute majority of (four out of six) voters would give at least that grade to the candidate. The majority ranking for the example is: $C_1 \succ C_3 \succ C_2$, according to the order of the majority grades. C_1 is the majority winner. The method involves additional

Candidates:	C_1	C_2	C_3
Best Grade:	5 (Excellent)	3 (Good)	4 (Very Good)
.	4 (Very Good)	3 (Good)	4 (Very Good)
.	4 (Very Good)	3 (Good)	4 (Very Good)
Majority Grade:	4 (Very Good)	1 (Reject)	3 (Good)
.	2 (Passable)	1 (Reject)	1 (Reject)
Worst Grade:	2 (Passable)	1 (Reject)	1 (Reject)

Table 1: MJ Example

procedures for breaking ties when ranking candidates and when deciding majority winner. We refer interested readers to Balinski and Laraki (2010) for more details. MJ requires a common language accepted by all voters for grading the candidates. Grades may be either discrete (as in Table 1) or continuous. A continuous grading language could be from 0 to 1, where 0 is commonly understood as "unacceptable", and 1 as the "most favorable".

The computation and ranking of majority grades require a complete evaluation of all candidates from all voters. We study important social choice problems where such evaluation is impossible to execute due to the continuous candidate space. One motivation for our work is a problem in air traffic flow management (ATFM) where a consensus input is required from a set of airlines in order to design and implement certain ATFM programs (see Ball et al. (2014, 2017); Evans et al. (2014); Swaroop and Ball (2012) for a description of this application and some preliminary ideas on the approach discussed in this paper). The required input is a numerical vector from a convex feasible region that specifies to an air navigation service provider how important tradeoffs among performance criteria should be made. A second, more widely known application requires the use of group decision-making mechanisms to find a capital budget allocation. There are many capital budgeting application contexts, particularly in the public sector, where a consensus must be reached among multiple decision-makers. In a basic problem statement of this type, there are n projects and a total available budget, C. The desired outcome is a feasible budget allocation: m_i for each project $i \in \{1, \cdots, n\}$. The set of feasible allocations is then defined by $P = \{(m_1, \cdots, m_n) : \sum_{i=1}^{n} m_i \leq C, \ m_i \geq 0, \ \forall i \in \{1, \cdots, n\}\}$. Viewing each feasible solution as an MJ candidate, the number of candidates is uncountably infinite, making the direct application of MJ impossible. In this paper, we develop models and computational approaches to apply MJ to this context.

1.1. Relevant Research

Extensive research has been conducted on the subjects of voting and elections. The most widely used methods include Approval Voting, Point Summing, Borda Count, among others. For the reasons discussed above we prefer MJ for our target applications. Therefore, although these other methods also have the potential to be extended to handle candidate spaces with infinite size, we do not explore them in this paper. Instead, our singular focus is extension of MJ method to continuous candidate spaces.

Aside from voting systems which only determine a single winner, there are group-ranking approaches that take ranking input from voters and output the group preference rankings of all candidates. Kemeny and Snell (1962) define the group-ranking problem when each voter's input is an ordinal preference ranking, and the group ranking is the one that minimizes the deviation from individual voters' rankings based on some distance measure. The analytic hierarchy process (AHP) developed by Saaty (1977) for multicriteria decision making has also been used in group-ranking problems. It requires intensity ranking (a pairwise comparison that provides the magnitude of the degree of preference) input from voters. AHP turns an intensity ranking matrix into a vector of weights for all candidates. Since its invention, it has been successfully applied to numerous areas to evaluate, rank or select candidates. However, like most of the

group-ranking methods including the one by Kemeny and Snell (1962), AHP builds upon the assumption that a full-ranking list for all candidates is provided by each voter. This prevents its direct application in our problem context. There is perhaps also a philosophical difference between our approach using MJ and the approaches using AHP. The AHP literature typically assumes that participants do not have precise knowledge of their own preferences, and preferences are quantified through an iterative process using pairwise comparisons. In our MJ-based approach, we address applications where voters can precisely estimate their grade functions, which map all candidates into numerical grades. For example, in our capital budgeting application, it is possible to relate every feasible capital allocation to a numerical grade that measures voter's preference, such as expected returns on investment. Similarly, in our ATFM application, it is possible (at least conceptually) to relate an airline's grade function to that airline's expected financial performance under every candidate vector in question.

Hochbaum and Levin (2006) propose generalized optimization approaches to produce group rankings where voters' inputs are also intensity rankings. Different from previous approaches, their approaches are amenable to partial preference lists: each voter provides a partial list that evaluates and compares only a subset of all the candidates. The partial preference lists are motivated by the fact that different voters may have different areas of expertise in reviewing candidates, and voters may not have the capability to review all the candidates in time, the latter of which is also the case in our problem. Note here that although the feature in Hochbaum and Levin (2006) of allowing partial lists is relevant to our problem context, it is difficult to apply this approach to our context where the candidate space is modeled as a convex set. Especially when the dimension of the candidate space is high, identifying good candidates for voting in this approach seems non-trivial.

Finally, our work is related to the computational social choice literature. In particular, Xia (2011) investigates "combinatorial voting" where the candidates have a multi-attribute structure. There is a set of attributes and a candidate is identified by the values these attributes take. Combinatorial voting faces somewhat similar challenges as our problem – the number of candidates is exponentially large. Xia (2011) thus develops new preference elicitation and aggregation methods to solicit preferences and compute winner efficiently. However, these methods cannot be directly applied to our context because they rely on the requirement that each attribute takes a value from a finite set, while in our problem context, it could take any continuous value from an interval. There is also similar literature in the field of combinatorial auctions (Conen and Sandholm, 2001) where efficient preference elicitation methods need to be developed for bundle evaluations.

1.2. Summary of Paper

The contribution of this paper is a practical approach to applying MJ where the set of candidates is of infinite size and represented by a convex set. Section 2 defines a mixed integer program that solves our problem assuming "perfect information", namely an explicit knowledge of all voter grade functions. Section 3 employs the results from Section 2 in creating a practical, iterative mechanism that generates an approximate solution while only requiring that the voters grade a limited number of candidates. Section 4 gives experimental results using our approach to address a capital budgeting application.

2. Winner Determination Problem

In this section, we discuss the winner determination problem of finding the majority winner over a convex candidate space given perfect information on voter preferences. The general context for the problem involves a group of voters, N, and a central planner who seek to select a candidate. These voters are not necessarily cooperative nor do they necessarily have common goals. The form of the candidate that we seek is a numerical vector $\mathbf{m} \in \mathbb{R}^n$ that is within

a candidate space P, which is modeled as a convex set. The dimension of space P is n. We assume that each voter $i \in N$ has a grade function $g_i(\cdot)$ that maps each candidate $\mathbf{m} \in P$ to a real number (common language across all voters) that represents the value of $\mathbf{m}$ to voter i. Each voter $i \in N$ also has a weight $w_i \in \mathbb{R}^+$. We apply MJ in this setting, i.e., we aim to ensure that the final selected candidate $\mathbf{m}$ has the highest majority grade among all candidates in P. We now describe some assumptions regarding the structure of the candidate space and the grade functions:

Assumption 1. *The candidate space P is a bounded convex set.*

We require convexity of the candidate space to enable proper definitions of grade functions over P in Assumption 2. We require P to be bounded to model the context where the range of candidates is in general limited.

Assumption 2. *Each voter's grade function $g_i(\mathbf{m})$ is continuous, concave and component-wise non-decreasing in $\mathbf{m}$, the last one implying that all partial derivatives of $g_i(\mathbf{m})$ are non-negative. Without loss of generality, we assume grades are continuous in $[0, 1]$, where a higher grade implies better acceptability by a voter.*

Continuity would be a reasonable assumption for many applications: very small changes in a candidate's component values should not induce jumps in grades. The non-decreasing assumption implies that higher values of the individual components of a candidate $\mathbf{m}$ are as good as or better than lower values. The concavity assumption expresses the diminishing returns property.

Under these assumptions, we describe an optimization model whose solution is the majority winner (i.e., the candidate in P that has the highest majority grade), given complete information of voters' preferences $g_i(\cdot)$ and weights w_i. First, we define *majority set* as the following,

Definition 1 (Majority Set)**.** *A majority set B is a subset of voters which satisfies the following property:*

$$\sum_{i \in B} w_i > \frac{\sum_{i \in N} w_i}{2},$$

i.e. the sum of weights of all the voters in B is strictly greater than one half of the total weight. We further define the set of all possible majority sets as $\mathcal{B}$.

Voter	Weight	Grade
1	23	1.00
2	46	0.90
3	31	0.85

(a) Voters' grades on one candidate

Majority Set	Voter with Minimal Grade	Minimum Grade
{1,2}	2	0.90
{1,3}	3	0.85
{2,3}	3	0.85
{1,2,3}	3	0.85

(b) All possible majority sets and their minimum grades

Table 2: MJ Illustration

We start the analysis with an example of three voters grading one single candidate (Table 2a). The total weight of the three voters sums up to 100. In this example, we can easily see

that the set of all possible majority sets $\mathcal{B}$ is $\{\{1,2\},\{2,3\},\{1,3\},\{1,2,3\}\}$. In Table 2b, we list all possible majority sets in the first column. For each majority set, we specify the voter which gives the minimum grade in the majority set (column 2) and the corresponding minimum grade (column 3). We would like to show that the maximum value of column 3, 0.90 in this case, is indeed the majority grade of this candidate. We formalize this result in the following lemma.

Lemma 1. *The majority grade $g_{MJ}(\mathbf{m})$ of a candidate $\mathbf{m}$ can be equivalently calculated as follows:*

$$g_{MJ}(\mathbf{m}) = \max_{B \in \mathcal{B}} \left\{ \min_{i \in B} g_i(\mathbf{m}) \right\} \tag{1}$$

Proof. This result follows directly from the definition of the majority grade: it is the highest grade such that there exists a majority set where all voters in the majority set grade that candidate at least as highly as that grade. The inner minimization in Eq. (1) is based on the fact that $\min_{i \in B} g_i(\mathbf{m})$ is the highest grade such that every voter in majority set B will give a grade greater than or equal to it. The outer maximization in Eq. (1) then aims to maximize that grade by searching over all possible majority sets. $\square$

Lemma 1 inspires us to develop a mathematical program to search over all possible majority sets $\mathcal{B}$ in order to find the majority winner. As before, we use $\mathbf{m}$ to define a vector of continuous variables representing an MJ candidate vector. Also, we define binary variables $x_i = 1, \forall i \in N$ if voter i is chosen to be in the corresponding majority set. Similarly, we define binary variables $y_i = 1, \forall i \in N$ if voter i assigns the lowest grade to $\mathbf{m}$ among all voters in the majority set indicated by $\mathbf{x}$. Continuous variable $v_i, \forall i \in N$ represents the grade given by voter i to candidate $\mathbf{m}$. Finally, auxiliary continuous variable z is equal to the majority grade of candidate $\mathbf{m}$ as we will prove next in Theorem 1. Theorem 1 introduces a mathematical program that computes the majority winner over the entire candidate space P.

Theorem 1. *Given each voter's grade function $g_i(\cdot)$, and weight w_i, and feasible candidate space P, the candidate with the highest majority grade g_{MJ}^* is the optimal solution $\mathbf{m}^*$ of the following mathematical program:*

$$g_{MJ}^* = \max_{\mathbf{x},\mathbf{y},\mathbf{v},\mathbf{m},z} \quad z \tag{2}$$

$$s.t. \quad \sum_{i \in N} w_i x_i \geq \frac{\sum_{i \in N} w_i}{2} + \epsilon_0 \tag{3}$$

$$\sum_{i \in N} y_i = 1 \tag{4}$$

$$x_i \geq y_i, \qquad\qquad\qquad \forall i \in N \tag{5}$$

$$\mathbf{m} \in P \tag{6}$$

$$v_i = g_i(\mathbf{m}), \qquad\qquad\qquad \forall i \in N \tag{7}$$

$$v_j \leq G^{max}(1 - x_i) + G^{max}(1 - y_j) + v_i, \quad \forall i \in N, j \in N : i \neq j \tag{8}$$

$$z \leq v_i + G^{max}(1 - y_i), \qquad\qquad \forall i \in N \tag{9}$$

$$x_i, y_i \in \{0, 1\}, \qquad\qquad\qquad \forall i \in N \tag{10}$$

where G^{max} is the maximum possible grade (under our assumptions $G^{max} = 1$) and ϵ_0 can be any positive number smaller than $\min_{\left\{ \mathbf{x} \in \{0,1\}^{|N|}: \ \sum_{i \in N} w_i x_i > \frac{\sum_{i \in N} w_i}{2} \right\}} \left(\sum_{i \in N} w_i x_i - \frac{\sum_{i \in N} w_i}{2} \right)$ which is the smallest positive difference between $\sum_{i \in N} w_i x_i$ and $\frac{\sum_{i \in N} w_i}{2}$ for all $\mathbf{x} \in \{0,1\}^{|N|}$.

Proof. Constraint (3) indicates that the sum of weights in the selected majority set should exceed one half of the total weight. Constraint (4) ensures that we select one voter as the one

who assigns the minimum grade to $\mathbf{m}$ among all voters in the selected majority set. There could be multiple voters who give a grade equal to this minimum grade, in which case any one of them can be chosen. Constraints (5) make sure that if voter i is selected to be the one to give the minimum grade, then voter i is included in the majority set. Constraint (6) restricts $\mathbf{m}$ to be selected from the feasible candidate space P. Constraints (7) state that v_i is the grade that voter i gives to candidate $\mathbf{m}$ according to its grade function $g_i(\cdot)$. Constraints (8) ensure that if voter j is selected to be the one to give the minimum grade to candidate $\mathbf{m}$ (i.e., if $y_j = 1$), then its grade v_j should indeed be less than or equal to other grades in the selected majority set. To see this, note that if $y_j = 1$ and $x_i = 1$, which means voter i is included in the majority set and voter j is selected to be the one who gives the minimum grade, then constraints (8) become $v_j \leq v_i$; otherwise, constraints (8) are redundant. Finally, constraints (9) along with objective (2) ensure that auxiliary variable z equals the minimum grade of $\mathbf{m}$ in the selected majority set. To see that, note that if $y_i = 1$, then constraints (9) are reduced to $z \leq v_i$; and if $y_i = 0$, then constraints (9) are redundant. Now we can see that the mathematical program described above is the same as $\max_{\mathbf{m} \in P, B \in \mathcal{B}} \{\min_{i \in B} g_i(\mathbf{m})\}$. Finally, from Lemma 1, we know that this is equivalent to $\max_{\mathbf{m} \in P} g_{\mathrm{MJ}}(\mathbf{m})$. $\square$

Note that in theory multiple candidates in P could have the same highest majority grade. In this situation certain tie-breaking rules, as developed by Balinski and Laraki (2010) in the original MJ theory, need to be imposed. However, we don't explicitly model tie-breaking rules in the formulation (2) - (10) because ties rarely happen in the contexts of interest since we allow voters to grade continuously between 0 and 1, which is an uncountably infinite space.

Although Theorem 1 allows us to formulate the problem of finding a majority winner over a continuous convex candidate space, the resultant formulation is not computationally approachable because of the non-convexity introduced by constraints (7). Since the grade functions $g_i(\cdot), \forall i \in N$ are concave, these equality constraints make the formulation a mixed-integer non-convex program. Fortunately, Corollary 1 below shows that constraints (7) can be relaxed without loss of optimality so that the resultant formulation becomes a mixed-integer convex program. Corollary 1 also exhibits a simplification of the formulation. Proof of Corollary 1 can be found in Appendix C.

Corollary 1. *Constraints (5) are redundant and constraints (7) can be relaxed to $v_i \leq g_i(\mathbf{m}), \forall i \in N$. Thus the following mixed-integer convex programming formulation MJ-OPT is equivalent to the formulation described in (2)-(10).*

$$
\begin{aligned}
(\text{MJ-OPT}) \quad \max \quad & z \\
s.t. \quad & (3) - (4), (6) \\
& v_i \leq g_i(\mathbf{m}), \quad \forall i \in N \qquad (11) \\
& (8) - (10)
\end{aligned}
$$

3. An Iterative Voting Approach

The mathematical development in Section 2 enables the computation of a majority winner over a continuous convex candidate space if we have perfect knowledge of every voter's grade function. However, this information is typically private, known only to the voters themselves. In this section, we develop an iterative voting approach which employs Theorem 1 but does not require explicit knowledge of grade functions. Note that we assume that voters submit their inputs truthfully according to their grade functions, i.e., there's no strategic behavior.

To start, the voters are provided an initial set of candidates and are asked to grade these candidates. The central planner then statistically estimates each voter's grade function based on that voter's submitted grades. We denote the estimated grade functions as $\hat{g}_i(\mathbf{m}), \forall i \in N$. We don't make any assumptions on the central planner's knowledge of the functional form of voters' true grade functions, apart from knowing that they are concave and component-wise

non-decreasing functions. So the central planner cannot directly estimate the grade functions' parameters. Instead, we use a non-parametric regression method called "convex regression" (Kuosmanen, 2008) to fit the best concave and component-wise non-decreasing function to estimate each voter's true grade function. This best-fit function turns out to be a piecewise-linear function with the number of pieces equal to the number of graded candidates. It also has desirable consistency property (Lim and Glynn, 2012). Suppose the central planner provides k candidates to the voters to grade. For each voter i, the estimation problem takes two types of inputs: (1) the set of candidates, and (2) voter i's grade associated with each of these candidates. The estimation problem outputs the best-fit piecewise-linear function's coefficients $(\mathbf{c}_i^j, d_i^j)$ for each piece $j \in \{1, \cdots, k\}$. Thus, the form of the best-fit piecewise-linear concave function for voter i is $\hat{g}_i(\mathbf{m}) = \min_{j \in \{1, \cdots, k\}} \left\{ (\mathbf{c}_i^j)^T \mathbf{m} + d_i^j \right\}$. The detailed formulation is presented in Appendix A.

The central planner then generates new candidates based on the estimated grade functions $\hat{g}_i(\mathbf{m})$, $\forall i \in N$. It is desirable that the generated candidates are of good quality, in the sense that they have relatively high majority grades. Thus we propose model $\widehat{\text{MJ-OPT}}$ as a building block for this iterative approach. Model $\widehat{\text{MJ-OPT}}$, which is presented next, produces the majority winner with the estimated grade functions. The only difference between this formulation and formulation MJ-OPT is that we replace $v_i \leq g_i(\mathbf{m})$, $\forall i \in N$ in constraints (11) with $v_i \leq \hat{g}_i(\mathbf{m})$, $\forall i \in N$. Note that $v_i \leq \hat{g}_i(\mathbf{m})$, $\forall i \in N$ is equivalent to constraints (12) since $\hat{g}_i(\mathbf{m}) = \min_{j \in \{1, \cdots, k\}} \left\{ (\mathbf{c}_i^j)^T \mathbf{m} + d_i^j \right\}$.

$$
\begin{aligned}
(\widehat{\text{MJ-OPT}}) \quad \max \quad & z \\
\text{s.t.} \quad & (3) - (4), (6) \\
& v_i \leq (\mathbf{c}_i^j)^T \mathbf{m} + d_i^j, \quad \forall i \in N, \ j \in \{1, \cdots, k\} \quad (12) \\
& (8) - (10)
\end{aligned}
$$

Based on the discussion above, we develop an algorithm to iteratively generate new candidates, refine the estimation of voter grade functions and thus approach the true majority winner over the entire candidate space. The overall flow of this process is summarized in Algorithm 1.

Algorithm 1 An Iterative Approach

---

  initialize
      Select a set of initial candidates, S.
      num_iter $= 1$
  while num_iter $\leq n_{\max}$ **do**
      Obtain each voter's grade for all newly added candidates in set S.
      Estimate each voter's grade function by running the estimation problem using the
      obtained grades of all candidates in set S.
      Generate a new candidate $\mathbf{m}_{\text{new}}$ by solving $\widehat{\text{MJ-OPT}}$.
      if $\|\mathbf{m}_{\text{new}} - \mathbf{m}\|_2 \geq \epsilon$, $\forall \mathbf{m} \in S$ **then**
         Add $\mathbf{m}_{\text{new}}$ to set S.
      else
         Break
      end if
      num_iter $=$ num_iter $+ 1$
  end while
  return the candidate with the highest majority grade in set S.

Algorithm 1 starts with a set of initial candidates. The algorithm then enters into a loop to (1) first obtain the grades (from the voters) of the candidates that haven't been graded yet,

then (2) estimate each voter's grade function by running the convex regression model, then (3) generate a new candidate $\mathbf{m}_{\text{new}}$ by solving $\widehat{\text{MJ-OPT}}$, and finally (4) add the new candidate $\mathbf{m}_{\text{new}}$ to the set S. The algorithm terminates either after certain pre-specified number (n_{max}) of iterations or if the newly generated candidate after solving $\widehat{\text{MJ-OPT}}$ is sufficiently similar to any of the existing candidates in S. The similarity here is measured in terms of their euclidean distance being less than or equal to a certain threshold ϵ. Upon termination, the algorithm returns the candidate with the highest majority grade among the candidates in set S.

4. A Capital Budgeting Example

In this section, we introduce a case study that considers a simple capital budgeting problem, i.e., the one involving allocation of a fixed amount of money (C) to n projects. Let $m_1, \cdots, m_n$ denote, respectively, the amounts of money allocated to projects 1 through n. Then the feasible candidate space P of all possible allocations is:

$$\left\{ (m_1, \cdots, m_n) : \sum_{i=1}^{n} m_i \leq C, m_i \geq 0, \forall i \in \{1, \cdots, n\} \right\}.$$

Consistent with previous discussion, the type of capital budgeting problems that we are interested in are group decision-making problems, i.e., the ones where the final budget allocation scheme is not determined by a single decision-maker but rather by a group. Suppose that the final allocation outcome $\mathbf{m}^* = (m_1^*, \cdots, m_n^*)$ is to be determined by N voters. Each voter has a grade function and a weight. Consistent with Assumption 2, the grade functions are concave. Concavity reflects a decreasing rate of return on investment in each project.

To demonstrate the effectiveness of our approach, we compare the performances of the following three methods. We define the *efficient frontier* $\bar{P}$ of the feasible candidate space P as the subset of the candidate space such that no candidate in P dominates any candidates in $\bar{P}$ in terms of the component values. In other words, there does not exist a candidate $\mathbf{m} \in P$ such that the following holds: $m_i \geq m_i'$, $\forall i \in \{1, \cdots, n\}$ and $\exists i' \in \{1, \cdots, n\}$ such that $m_{i'} > m_{i'}'$, for some $\mathbf{m}' \in \bar{P}$. We can restrict ourselves to the candidates on the efficient frontier because for any candidates not on the efficient frontier, we can always find one on the frontier that has higher or equal majority grade.

1. *Random Generation*:
   This is a baseline approach to find the majority winner. We randomly generate k candidates along the efficient frontier of the candidate space (detailed method is described in Appendix B), and perform one round of candidate grading by all the voters. The candidate with the highest majority grade among these k candidates is selected as the winner.

2. *Iterative Approach Starting with Voters' Best Candidates*:
   This is the iterative approach discussed in Algorithm 1. We initialize Algorithm 1 with a set of candidates which is made up of each voter's most preferred candidate. We report the performance of the majority winner among the candidates in the set S at each iteration.

3. *Iterative Approach Starting with Evenly Distributed Candidates*:
   The only difference between this approach and the previous one is that we initialize this approach with a set of evenly distributed candidates along the efficient frontier. The number of initial candidates is set to be equal to the number of voters so that we can have a fair comparison with the iterative approach starting with voters' best candidates.

We benchmark all solutions to $\mathbf{m}_{\text{exact}}$, which is the true majority winner (the candidate that has the highest majority grade based on voters' true grade functions) over the entire feasible space. It is obtained by solving formulation MJ-OPT. Denote its majority grade as

v_{exact}. We then use v_{exact} to report the optimality gap of all solutions which is calculated as $\big((v_{\text{exact}} - v)/v_{\text{exact}}\big) * 100\%$, where v is the true majority grade of the solution we want to evaluate. The lower the value of optimality gap the better is the performance.

Without loss of generality, we set the total budget C equal to 1. We set the number of voters to 10. We test two weighting schemes: (1) equal weights, where each voter has the same weight; (2) random weights, where each voter's weight is generated from a discrete uniform distribution between 1 and 10. We set to 10 the number of projects to be potentially funded. To compare the performances of the three approaches, we run 100 rounds of simulation in total. For each simulation, we randomly generate one set of all voters' true grade functions. Under random weighting scheme, we also randomly generate one set of voter weights for each simulation run.

For the random generation approach, we report the optimality gap under $10, 20, 30, 40$ candidates (denoted as Rand10, Rand20, Rand30 and Rand40, respectively). For each of the two iterative approaches we start with 10 candidates, and report the optimality gap upon termination and total number of candidates graded until termination. For the iterative approach starting with evenly distributed candidates (denoted as IterEven), we start with 10 unit vectors $\mathbf{e}_1, \cdots, \mathbf{e}_{10}$ where $\mathbf{e}_i$ is the vector with all components equal to zero except for the ith component which is equal to one. For the iterative approach starting with voters' best candidates (denoted as IterBest), we start with all 10 voters' most preferred candidates, one most preferred candidate per voter. In the iterative approach, we set $n_{\max} = 30$ so that at most 40 candidates will be evaluated. We also set $\epsilon = 0.01$ as another termination criteria. All the mathematical programs were written in Julia using JuMP as the algebraic modeling language (Dunning et al., 2015). All linear programming and mixed-integer linear programming problems were solved using Gurobi 7.5 (Gurobi Optimization Inc., 2018) and all nonlinear optimization problems were solved using Ipopt (Wächter and Biegler, 2006) and Bonmin (Bonami et al., 2008).
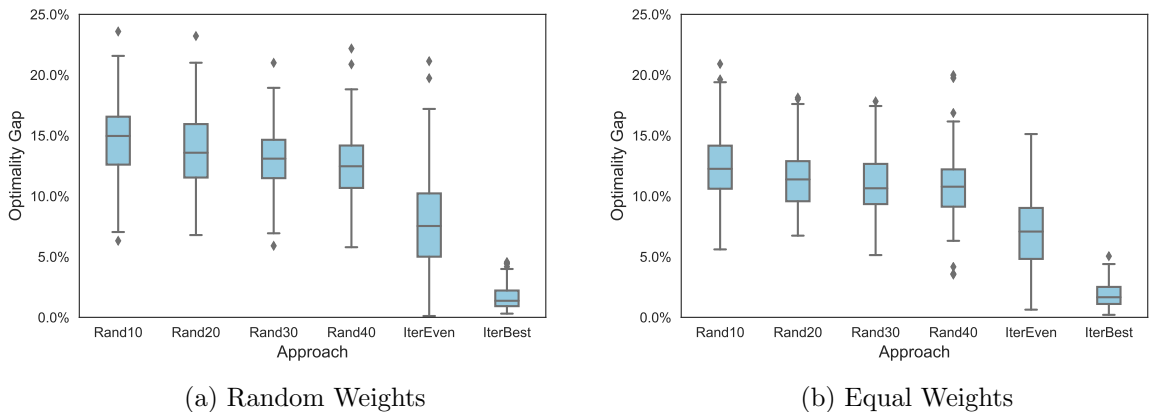

(a) Random Weights (b) Equal Weights

Figure 1: Optimality gap upon termination under all tested approaches

Figure 1 uses standard boxplots to depict the distributions of optimality gaps upon termination of the iterative approaches and benchmarks them against the random generation baselines with different number of candidates. Under both weighting schemes, iterative approach starting with voters' best candidates produces the best results, with an average optimality gap of 1.67% under random weights and 1.79% under equal weights. Iterative approach starting with evenly distributed candidates reaches an average optimality gap of 7.98% under random weights and 7.04% under equal weights. In comparison, for the random generation baselines, even with 40 candidates, the average optimality gap is as large as 12.58% under random weights and 8.48% under equal weights.

Figure 2 shows, using standard boxplots, the distributions of the number of candidates evaluated until termination for the two iterative approaches. The iterative approach starting with voters' best candidates evaluates on average 20.2 and 22.7 candidates under random and equal

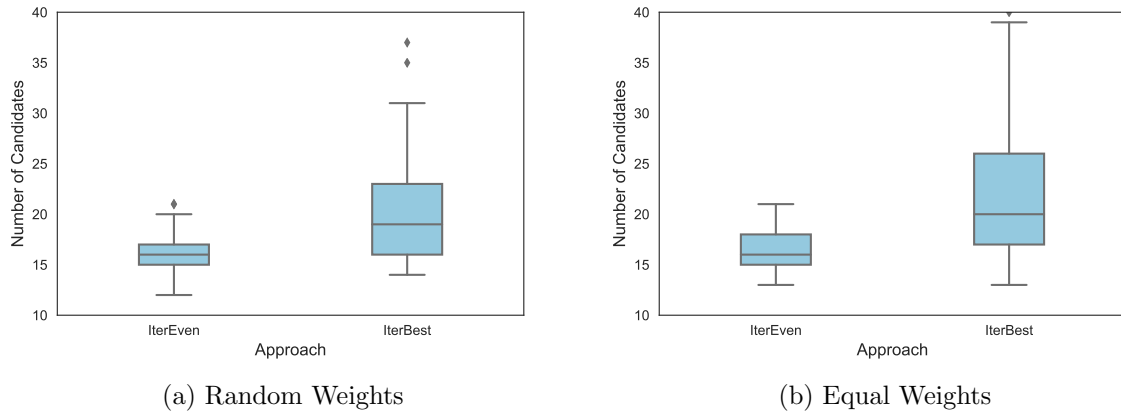|	
(a) Random Weights	(b) Equal Weights

Figure 2: Number of candidates graded until termination under the iterative approaches

weighting schemes. In contrast, iterative approach starting with evenly distributed candidates evaluates on average only 15.8 candidates under random weights and 16.8 candidates under equal weights. Both iterative approaches save a significant amount of grading effort compared to the random generation approach (which requires at least 30 to 40 candidates to be graded to obtain somewhat reasonable results).

In summary, these computational results demonstrate the superior performance of our iterative approaches compared to the baselines in terms of the optimality gaps and also in terms of the required grading effort.

Acknowledgements

References

Arrow, K., 1951. Individual values and social choice. New York: Wiley 24.

Balinski, M., Laraki, R., 2011. Election by majority judgment: experimental evidence. In: In Situ and Laboratory Experiments on Electoral Law Reform. Springer, pp. 13–54.

Balinski, M. L., Laraki, R., 2010. Majority judgment: measuring, ranking, and electing. MIT Press.

Balinski, M. L., Laraki, R., 2014. Judge: Don't vote! Operations Research 62 (3), 483–511.

Ball, M., Barnhart, C., Hansen, M., Kang, L., Liu, Y., Swaroop, P., Vaze, V., Yan, C., 2014. Distributed mechanisms for determining NAS-wide service level expectations. Tech. rep., NEXTOR II.

Ball, M., Barnhart, C., Hansen, M., Kang, L., Liu, Y., Vaze, V., Yan, C., 2017. Service level expectation setting for air traffic flow management: Practical challenges and benefits assessment, twelfth USA/Europe Air Traffic Management Research and Development Seminar (ATM2017).

Bonami, P., Biegler, L. T., Conn, A. R., Cornuéjols, G., Grossmann, I. E., Laird, C. D., Lee, J., Lodi, A., Margot, F., Sawaya, N., et al., 2008. An algorithmic framework for convex mixed integer nonlinear programs. Discrete Optimization 5 (2), 186–204.

Conen, W., Sandholm, T., 2001. Preference elicitation in combinatorial auctions. In: Proceedings of the 3rd ACM conference on Electronic Commerce. ACM, pp. 256–259.

Dunning, I., Huchette, J., Lubin, M., 2015. JuMP: A modeling language for mathematical optimization. SIAM Review.

Evans, A., Vaze, V., Barnhart, C., 2014. Airline-driven performance-based air traffic management: Game theoretic models and multicriteria evaluation. Transportation Science,Articles in Advance.

Gurobi Optimization Inc., 2018. Gurobi Optimizer Reference Manual.
    URL http://www.gurobi.com/documentation/7.5/

Hochbaum, D. S., Levin, A., 2006. Methodologies and algorithms for group-rankings decision. Management Science 52 (9), 1394–1408.

Kemeny, J. G., Snell, L., 1962. Preference ranking: an axiomatic approach. Mathematical models in the social sciences, 9–23.

Kuosmanen, T., 2008. Representation theorem for convex nonparametric least squares. The Econometrics Journal 11 (2), 308–325.

Lim, E., Glynn, P. W., 2012. Consistency of multidimensional convex regression. Operations Research 60 (1), 196–208.

Mazumder, R., Choudhury, A., Iyengar, G., Sen, B., 2015. A computational framework for multivariate convex regression and its variants. arXiv preprint arXiv:1509.08165.

Meyer, R., Johnson, E. J., 1995. Empirical generalizations in the modeling of consumer choice. Marketing Science 14 (3_supplement), G180–G189.

Saaty, T. L., 1977. A scaling method for priorities in hierarchical structures. Journal of mathematical psychology 15 (3), 234–281.

Swaroop, P., Ball, M., 2012. Consensus building mechanism for setting service expectations in air traffic management. Transportation Research Record: Journal of the Transportation Research Board 2325, 87–96.

Wächter, A., Biegler, L. T., 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical programming 106 (1), 25–57.

Xia, L., 2011. Computational voting theory: game-theoretic and combinatorial aspects. Ph.D. thesis, Duke University.

Appendix A. Grade Function Estimation

In this appendix, we suppress the voter index i for simplicity. Suppose $\mathbf{m}^1, \mathbf{m}^2, \cdots, \mathbf{m}^k$ are k candidates that we ask the voters to grade, and $\mathbf{y} = \{y^1, y^2, \cdots, y^k\}$ are the corresponding grades submitted by a specific voter.

We utilize a nonparametric regression method called "convex regression" (concave in our setting) to estimate voters' grade functions. The idea of this estimation method is to find an estimator $\hat{g}(\cdot)$ which minimizes of the sum of squares: $\sum_{i=1}^{k}(\hat{g}(\mathbf{m}^i) - y^i)^2$ over functions $\hat{g}$ that are concave and component-wise non-decreasing in $\mathbf{m}$. Let $\mathbf{f} = \{f_1 = \hat{g}(\mathbf{m^1}), f_2 = \hat{g}(\mathbf{m^2}), \cdots, f_k = \hat{g}(\mathbf{m^k})\}$, respectively, be the grades estimated through this best-fit function for candidates $\mathbf{m}^1, \mathbf{m}^2, \cdots, \mathbf{m}^k$ respectively. The formulation for solving the convex regression is presented below (Lim and Glynn, 2012; Mazumder et al., 2015):

$$\min_{\mathbf{f}, \xi} \quad \|\mathbf{f} - \mathbf{y}\|_2^2 \tag{A.1}$$

$$\text{s.t.} \quad f_j \le f_i + \xi_i^T(\mathbf{m}^j - \mathbf{m}^i), \ \forall i, j \in \{1, \cdots, k\}, i \ne j \tag{A.2}$$

$$\|\xi_i\|_2^2 \le C, \ \forall i \in \{1, \cdots, k\} \tag{A.3}$$

$$\xi_i \ge \mathbf{0}, \ \forall i \in \{1, \cdots, k\} \tag{A.4}$$

Note that $\xi_1, \cdots, \xi_k$ are the supergradients of the best estimated convex function $\hat{g}(\cdot)$ at points $\mathbf{m}^1, \cdots, \mathbf{m}^k$ respectively. Constraints (A.2) represent the concavity requirement regarding the supergradient of a concave function. Constraints (A.3) represent the Lipschitz constraint to bound the $\mathcal{L}_2$ norm of the supergradients ξ from above. These constraints are imposed to prevent overfitting and to ensure certain convergence rate properties (Mazumder et al., 2015). We refer interested readers to Mazumder et al. (2015) for details. In our implementation, we chose $C = 10$ which is an upper bound that we empirically found for all the grade functions generated using the method described in Appendix D. Finally, constraints (A.4) ensure that the best-fit function is component-wise non-decreasing. After obtaining the optimal solution $\mathbf{f}^*, \xi^*$ of the quadratic program (A.1)-(A.4), the estimated grade function $\hat{g}(\cdot)$ is specified as follows:

$$\hat{g}(\mathbf{m}) = \min_{i \in \{1, \cdots, k\}} \left\{ y_i + \xi_i^T \left(\mathbf{m} - \mathbf{m}^i \right) \right\}$$
$$= \min_{i \in \{1, \cdots, k\}} \left\{ \xi_i^T \mathbf{m} + \left(y_i - \xi_i^T \mathbf{m}^i \right) \right\}$$

Appendix B. Procedure for Random Generation of Candidates on the Efficient Frontier

We first generate a vector $\mathbf{c} \in [0, 1]^n$ where each component c_i is uniformly distributed between 0 and 1. We then solve the following linear program and denote the optimal solutions of this linear program as $\mathbf{m}^*$.

$$\max \quad \sum_{i=1}^{n} m_i \tag{B.1}$$

$$\text{s.t.} \quad c_j m_i = c_i m_j, \ \forall i, j \in \{1, \cdots, n\} \tag{B.2}$$

$$\mathbf{m} \in P \tag{B.3}$$

The linear program (B.1) - (B.3) essentially finds a candidate $\mathbf{m}^*$ by moving along the direction $(c_1, c_2, \cdots, c_n)$ as far as possible while staying within the feasible candidate space P. However, the resultant candidate might not be on the efficient frontier. The next linear program

further projects $\mathbf{m}^*$ onto the efficient frontier:

$$\max \quad \sum_{i=1}^{n} m_i \tag{B.4}$$

$$\text{s.t.} \quad m_i \geq m_i^*, \ \forall i \in \{1, \cdots, n\} \tag{B.5}$$

$$\mathbf{m} \in P \tag{B.6}$$

One can easily check that the optimal solution of this second linear program (B.4) - (B.6) is on the efficient frontier of P.

Appendix C. Proof of Corollary 1

To prove that constraints (5) are redundant, we are going to prove that for any optimal solution $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ to the formulation described by (2)-(4) and (6)-(10), which violates constraints (5), we can always modify the values of $\mathbf{y}$ and transform it into another optimal solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ which satisfies constraints (5). That is, for any such $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$, we can always find an optimal solution to the formulation described by (2)-(10) by only changing the $\mathbf{y}$ values. For any such $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ that doesn't satisfy constraints (5), there is some voter i_0 with $y_{i_0}^* = 1$ and $x_{i_0}^* = 0$. Let $j_0 = argmin_{i \in N : x_i^* = 1} v_i^*$. We construct a new solution with $\mathbf{y}^{**}$ such that $y_{j_0}^{**} = 1$ and $y_j^{**} = 0, \forall j \in N \setminus \{j_0\}$. We claim that $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ is also an optimal solution to formulation (2)-(4) and (6)-(10) and satisfies $y_{j_0}^{**} = 1$, $x_{j_0}^* = 1$. To see that, we only need to prove that $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ is feasible because they have the same objective function values z^*. We can verify that constraints (3),(4),(6),(7), and (10) are satisfied. Constraints (8) are satisfied because the way we select j_0 ensures that $v_{j_0}^* \leq v_i^*, \forall i \in N : x_i^* = 1$. Constraints (9) are also satisfied because of the fact $z^* \leq v_{i_0}^* \leq v_{j_0}^*$. The first inequality holds because $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ satisfies constraints (9) and $y_{i_0}^* = 1$. The second inequality holds because $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ satisfies constraints (8) and $x_{j_0}^* = 1$, $y_{i_0}^* = 1$. We can repeat this process for all such i_0 with $y_{i_0}^* = 1$ and $x_{i_0}^* = 0$ one by one, to arrive at a solution that satisfies constraints (5). This proves that constraints (5) are redundant.

To prove that equality constraints (7) can be relaxed to less-than-or-equal-to constraints, we follow the same proof technique as the one used in the first part of this proof. So we are going to show that for any optimal solution $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ of MJ-OPT which violates constraints (7), we can always modify the values of $\mathbf{v}$ and/or $\mathbf{y}$ to transform it into another optimal solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^{**}, \mathbf{m}^*, z^*\}$ of MJ-OPT which satisfies constraints (7). We start with an optimal solution of MJ-OPT which doesn't satisfy constraints (7), i.e., there exists some $i \in N$ such that $v_i^* < g_i(\mathbf{m}^*)$. We next divide our discussion into two different cases:

1. If $y_i^* = 0$: Replace v_i^* with $v_i^{**} = g_i(\mathbf{m}^*)$. The new solution is still feasible and thus optimal for MJ-OPT. This is because constraints (8) are redundant at index i, and thus they have no restriction on increasing the value of v_i. Note that constraints (9) can never be violated by increasing the value of any v_i^*.

2. If $y_i^* = 1$: Let $j_0 = argmin_{j \in N \setminus \{i\} : x_j^* = 1} v_j^*$. Then replace v_i^* with $v_i^{**} = g_i(\mathbf{m}^*)$ and replace $\mathbf{y}^*$ with $\mathbf{y}^{**}$ such that $y_{j_0}^{**} = 1$ and $y_j^{**} = 0, \forall j \in N \setminus \{j_0\}$. Now, we only need to prove that this new solution is still feasible, and thus optimal, for MJ-OPT. We first claim that $v_{j_0}^* = v_i^*$. To see that, suppose $v_{j_0}^* > v_i^*$ instead (note that $v_i^* \leq v_{j_0}^*$ because of constraints (8) and knowing that $y_i^* = 1$, and $x_{j_0}^* = 1$). Replacing v_i^* with $v_i' = \min\{g_i(\mathbf{m}^*), v_{j_0}^*\}$, and adjusting the $\mathbf{y}^*$ variables accordingly, the resultant solution is still feasible but with a higher objective value equal to $z^* + (v_i' - v_i^*)$ because $v_i' = \min\{g_i(\mathbf{m}^*), v_{j_0}^*\} > v_i^*$. This reaches a contradiction with the assumption that $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ is an optimal solution. So we have proved that $v_{j_0}^* = v_i^*$. Now consider the new solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^{**}, \mathbf{m}^*, z^*\}$ constructed. We can easily verify that it satisfies

constraints (3), (4), (5), (6), and (10). Constraints (8) are satisfied because $v_{j_0}^{**} = v_{j_0}^* = v_i^* \le v_j^*, \forall j \in N : x_j^* = 1$. Constraints (9) are also satisfied because $z^* = v_i^* = v_{j_0}^* = v_{j_0}^{**}$, where the first equality holds because of the optimality of $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$.

In summary, for any optimal solution $\{\mathbf{x}^*, \mathbf{y}^*, \mathbf{v}^*, \mathbf{m}^*, z^*\}$ of MJ-OPT, we can transform it to another optimal solution $\{\mathbf{x}^*, \mathbf{y}^{**}, \mathbf{v}^{**}, \mathbf{m}^*, z^*\}$ for MJ-OPT, which satisfies constraints (5) and (7). This finishes the proof of equivalence between MJ-OPT and formulation described by (2)-(10). $\square$

Appendix D. Specifications and Generation of True Grade Functions

In this appendix, we present the specifications and the process of generation of the underlying true grade functions that we use in the computational experiments in Section 4. We will suppress the voter index i in this appendix for simplicity. Without loss of generality, the individual components of the candidate vectors, m_j, $\forall j \in \{1, \cdots, n\}$ are normalized to have allowable values between 0 and 1. Recall that n here is the dimension of the feasible space. In the capital budgeting example (Section 4), $n = 10$.

The grade function $g(\mathbf{m})$ for a specific voter is composed of component-wise value functions $v_j(m_j)$ of each component m_j. We define the overall value function $V(\mathbf{m})$, which combines the component-wise value functions as a multiplicative-multilinear function of $v_j(m_j)$'s, and allows modeling complementarities and substitutions among the different components. This functional form is based on well-accepted theories developed by economists and marketing researchers in the fields of choice modeling and multi-attribute valuation (Meyer and Johnson, 1995):

$$V(\mathbf{m}) = \sum_{j=1}^n r_j v_j(m_j) + \sum_{1 \le j < k \le n} r_{jk} v_j(m_j) v_k(m_k), \tag{D.1}$$

Coefficients r_j here are non-negative. For pairwise interaction coefficients r_{jk}, if $r_{jk} > 0$, then it means that components j and k are complements; on the other hand, if $r_{jk} < 0$, then it means that components j and k are substitutes. Finally, the normalization step converts the overall value into a grade, using a simple linear scaling based on the maximum value $V^{\max} = \max_{\mathbf{m} \in P} V(\mathbf{m})$. Thus the grade function for the voter evaluated at candidate $\mathbf{m}$ is specified as:

$$g(\mathbf{m}) = \frac{V(\mathbf{m})}{V^{\max}} \tag{D.2}$$

A quadratic form without an intercept is specified for each component-wise value function, $v_j(m_j) = a_j m_j^2 + b_j m_j$, which was also used by Evans et al. (2014) in the collaborative air traffic flow management context. We require each $v_j(m_j)$ to be concave and non-decreasing; and without loss of generality, we also require $v_j(m_j) \in [0,1], \forall m_j \in [0,1]$. Therefore, the values of a_j and b_j need to be constrained such that $-1 \le a_j \le 0$ and $-2a_j \le b_j \le 1 - a_j$. Moreover, to ensure global concavity of the overall grade function $g(\mathbf{m})$, further constraints need to be imposed on the coefficients. We first expand the grade function $g(\mathbf{m})$ according to (D.1) and (D.2) as follows, where $K_j^a, K_j^b, K_{jk}^{aa}, K_{jk}^{ab}, K_{jk}^{ba}, K_{jk}^{bb}$ are some constants that can be derived from a_j, b_j, r_j, r_{jk}.

$$g(\mathbf{m}) = \sum_{j=1}^n K_j^b m_j + \sum_{j=1}^n K_j^a m_j^2 + \sum_{1 \le j < k \le n} \left(K_{jk}^{bb} m_j m_k + K_{jk}^{ab} m_j^2 m_k + K_{jk}^{ba} m_j m_k^2 + K_{jk}^{aa} m_j^2 m_k^2 \right),$$

where

$$K_j^a = \frac{r_j a_j}{V^{\max}}, K_j^b = \frac{r_j b_j}{V^{\max}}, K_{jk}^{aa} = \frac{r_{jk} a_j a_k}{V^{\max}}, K_{jk}^{ab} = \frac{r_{jk} a_j b_k}{V^{\max}}, K_{jk}^{ba} = \frac{r_{jk} b_j a_k}{V^{\max}}, K_{jk}^{bb} = \frac{r_{jk} b_j b_k}{V^{\max}}$$
(D.3)

Note that the component-wise non-decreasing condition on $g(\mathbf{m})$ is equivalent to $\nabla g(m) \geq \mathbf{0}$. Also, the Hessian matrix of a function being negative semi-definite in a given region is a necessary and sufficient condition for the concavity of the function within that region. Let the Hessian matrix of the grade function be:

$$\mathbf{H_g} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{12} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{1n} & g_{2n} & \cdots & g_{nn} \end{bmatrix}$$

where g_{jk} is the second-order partial derivative of $g(\mathbf{m})$ with respect to m_j and m_k. The first-order partial derivatives, g_j, $j \in (1, 2, \cdots, n)$ are:

$$g_j = \frac{\partial g(\mathbf{m})}{\partial m_j} = K_j^b + 2K_j^a m_j + \sum_{k:k<j} \left(K_{kj}^{bb} m_k + K_{kj}^{ab} m_k^2 + 2K_{kj}^{ba} m_k m_j + 2K_{kj}^{aa} m_k^2 m_j \right)$$
$$+ \sum_{k:j<k} \left(K_{jk}^{bb} m_k + 2K_{jk}^{ab} m_k m_j + K_{jk}^{ba} m_k^2 + 2K_{jk}^{aa} m_k^2 m_j \right)$$

The second-order partial derivatives ($\forall j = 1, \cdots, n; 1 \leq j < k \leq n$) are:

$$g_{jj} = \frac{\partial g_j}{\partial m_j} = 2 \left(K_j^a + \sum_{k:k<j} \left(K_{kj}^{ba} m_k + K_{kj}^{aa} m_k^2 \right) + \sum_{k:j<k} \left(K_{jk}^{ab} m_k + K_{jk}^{aa} m_k^2 \right) \right)$$
$$g_{jk} = \frac{\partial g_j}{\partial m_k} = K_{jk}^{bb} + 2K_{jk}^{ab} m_j + 2K_{jk}^{ba} m_k + 4K_{jk}^{aa} m_k m_j$$

We then have,

$$\mathbf{m^T H_g m} = \begin{bmatrix} m_1 & m_2 & \cdots & m_n \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{12} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{1n} & g_{2n} & \cdots & g_{nn} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix}$$
$$= \sum_{j=1}^{n} m_j^2 g_{jj} + 2 \left(\sum_{1 \leq j < k \leq n} m_j m_k g_{jk} \right)$$

If $\forall \mathbf{m} \in P$ such that $\mathbf{m} \neq \mathbf{0}$, $\mathbf{m^T H_g m} \leq 0$, i.e., if $\mathbf{H_g}$ is negative semi-definite over the entire feasible candidate space, then the grade function $g_i(\mathbf{m})$ is concave over P. Now we discuss in detail the process of generating parameters a_j, b_j, r_j, and r_{jk} to ensure that the overall grade function is concave and component-wise non-decreasing.

We summarize the overall algorithm for generating one set of voters' grade functions in Algorithm 2. As described before, the coefficients a_j and b_j in $v_j(m_j) = a_j m_j^2 + b_j m_j$ should satisfy $a_j \in [-1, 0]$ and $b_j \in [-2a_j, 1 - a_j]$. For sampling, we assume that all parameters, namely a_j, b_j, r_j, and r_{jk}, are uniformly distributed within their specified ranges. For the purposes of Algorithm 2, we assume that $r_j \sim \text{unif}(l_j, u_j)$, $\forall j \in \{1, \cdots, n\}$, with $l_j = 2$ and $u_j = 6, \forall j \in \{1, \cdots, n\}$. Similarly, we assume that $r_{jk} \sim \text{unif}(l_{jk}, u_{jk})$ $\forall j, k \in \{1, \cdots, n\}, j < k$

with $l_{jk} = -2$ and $u_{jk} = 2$, $\forall j, k \in \{1, \cdots, n\}, j < k$. Note that $r_j > |r_{jk}|$ to constrain the interaction effects to be smaller than the major effects. We check a discrete grid of points (denoted by $P' \subset P$), which are evenly spaced in P, for the component-wise non-decreasing condition and the concavity condition. Specifically, in the capital budgeting case study, we check at all points in $P' = \{0, 0.25, 0.5, 0.75, 1\}^{10} \cap P$.

Algorithm 2 Generating Voters' True Grade Functions
---

   repeat
      for $j = 1 : n$ **do**
         $a_j \sim \text{unif}(-1, 0)$
         $b_j \sim \text{unif}(-2a_j, 1 - a_j)$
         $r_j \sim \text{unif}(l_j, u_j)$
         for $k = (j + 1) : n$ **do**
            $r_{jk} \sim \text{unif}(l_{jk}, u_{jk})$
         end for
      end for
      concavity condition = true
      non-decreasing condition = true
      for all $\mathbf{m} \in P'$ **do**
         if $\nabla g(\mathbf{m}) \geq \mathbf{0}$ **then**
            do nothing
         else
            non-decreasing condition = false
            break the *for* loop
         end if
         if $\mathbf{m}^{\mathbf{T}} \mathbf{H_g} \mathbf{m} > 0$ **then**
            concavity condition = false
            break the *for* loop
         end if
      end for
   until concavity condition and non-decreasing condition are satisfied
   calculate $K_j^a, K_j^b, K_{jk}^{aa}, K_{jk}^{ab}, K_{jk}^{ba}, K_{jk}^{bb}$ according to (D.3)
   return $K_j^a, K_j^b, K_{jk}^{aa}, K_{jk}^{ab}, K_{jk}^{ba}, K_{jk}^{bb}$