Manipulation-Robust Selection of Citizens' Assemblies

Bailey Flanigan,[1] **Jennifer Liang,**[2] **Ariel D. Procaccia,**[2] **Sven Wang** [3]

[1] Carnegie Mellon University, [2] Harvard University, [3] Massachusetts Institute of Technology

Abstract

Among the recent work on designing algorithms for selecting citizens' assembly participants, one key property of these algorithms has not yet been studied: their *manipulability*. Strategic manipulation is a concern because these algorithms must satisfy representation constraints according to volunteers' *self-reported* features; misreporting these features could thereby increase a volunteer's chance of being selected, decrease someone else's chance, and/or increase the expected number of seats given to their group. Strikingly, we show that *Leximin* — an algorithm that is widely used for its fairness — is highly manipulable in this way. We then introduce a new class of selection algorithms that use ℓ_p norms as objective functions. We show that the manipulability of the ℓ_p-based algorithm decreases in $O(1/n^{1-1/p})$ as the number of volunteers n grows, approaching the optimal rate of $O(1/n)$ as $p \to \infty$. These theoretical results are confirmed via experiments in eight real-world datasets.

1 Introduction

In a *citizens' assembly*, a panel of randomly-chosen constituents convenes to make a policy recommendation on a political issue. Although citizens' assembly participants are not career politicians, their recommendations are informed by an extensive process of learning from experts and deliberating with one another. As such, citizens' assemblies are appealing because they combine the goals of engaging everyday citizens in democratic decision-making, while also facilitating informed decisions. Citizens' Assemblies are now being used to make increasingly high-profile decisions around the world (Participedia 2023); for example, France recently ran a national-level assembly on the topic of assisted dying, and its outcome is slated to affect policy on palliative care (Bürgerrat 2023).

Because the participants of a citizens' assembly represent their entire underlying constituency, the process by which they are selected is crucial to whether the policy recommendation they produce is perceived as trustworthy. The importance of this selection process has motivated a growing body of research on *selection algorithms* (Ebadian and Micha 2023; Ebadian et al. 2022; Flanigan et al. 2020, 2021; Flanigan, Kehne, and Procaccia 2021), which solve the following task: from among a *pool* of volunteers, randomly sample a *panel* that is (at least approximately) *descriptively representative* of the underlying population. This means that if the population is 48% women, the panel should be approximately 48% women. Because exact representation of all identities cannot be achieved with a finite-size panel, practitioners' main goal is to achieve representation with respect to a handful of key features, such as gender, age, geographic location, education level, and opinion on the issue at hand.

The main algorithmic challenge in selecting descriptively representative participants is *self-selection bias*: different demographic groups agree to participate at vastly different rates, so the pool of volunteers from which the panel is sampled is demographically skewed compared to the underlying population. Consequently, simple sampling techniques do not produce the desired descriptive representation.

Existing work has circumvented the challenge of achieving representation to a large degree. The first selection algorithms, developed by practitioners, were heuristics that searched for representative panels, injecting randomness wherever possible. More recent work has contributed algorithms that not only find representative panels, but do so in a way that achieves other desiderata simultaneously. For example, Flanigan et al. (2021) presents a framework of algorithms that are *maximally fair* to individual pool members: that is, they make pool members' probabilities of being selected as equal as possible, subject to representation constraints. One algorithm within this framework, called *Leximin* (Flanigan et al. 2021), is now widely used in practice.

Beyond the desiderata of representation and maximal fairness, follow-up work has contributed methods for additionally achieving *transparency* (Flanigan, Kehne, and Procaccia 2021). However, at the current frontier of research on selection algorithms, a key desideratum remains yet untouched: their *manipulability*.

In this paper, we initiate the study of selection algorithms' vulnerability to perhaps the most salient type of potential manipulation: *volunteers misreporting their features*. With Example 1.1, we now illustrate in detail why the selection process, as it commonly works in practice, can permit — and strongly incentivize — such manipulation.

Example 1.1. We want to select a panel of 10 people to convene on climate policy. We care about descriptive representation of one feature only: people's level of concern about climate change. This feature has two possible values: those who are *less concerned* (20% of the population) and *more*

concerned (80% of the population). Thus, we will reserve 2 and 8 panel seats for these respective groups.

STAGE 1: RECRUITING THE POOL OF VOLUNTEERS. We send out invitations to 1000 uniformly sampled households in our constituency. In response, 100 people volunteer to participate, but they are strongly self-selected: only 4 are truly *less concerned*, and 96 of them are truly *more concerned*.[1] In preparation for selection, we ask all 100 volunteers to report which group they belong to. Among these volunteers, suppose there is one strategic agent i who is truly *more concerned*, but is willing to misreport their group membership if it increases their chance of being on the panel.

STAGE 2: PANEL SELECTION. Given this pool of volunteers and their self-reported group memberships, a selection algorithm is then used to choose a panel. We assume nothing about this algorithm except that it treats people in the same group uniformly, and it produces a panel with 2 seats for *less concerned* people and 8 seats for *more concerned* people.

It is not hard to see that, in this example, i benefits significantly from misreporting their group membership. If i truthfully reports they are *more concerned*, they will join a group of 96 people for whom the panel has 8 seats, and thus will be chosen with probability $8/96 \approx 8\%$. If i reports that they are *less concerned*, they will join a group of 5 people for whom the panel has 2 seats, and will be chosen with probability $2/5 = 40\%$. By misreporting that they are *less concerned*, i can increase their selection probability by almost 32%. Moreover, with probability 40%, i will be given a panel seat reserved for *less concerned* people, thereby giving the group of *more concerned* people an extra panel seat.

Example 1.1 illustrates why such manipulation is of practical concern: the nature of self-selection bias in this example would be fairly easy for constituents to anticipate — surely, people who care less about climate change will be less likely to volunteer — making the optimal manipulation public knowledge.[2] Moreover, we cannot always prevent manipulation through verification; here, people's opinions would be impossible to check. As citizens' assemblies are used for increasingly higher-profile decisions, the political power associated with participating — and thus the incentive to manipulate — will only increase. Example 1.1 also shows a fundamental impossibility: when there is self-selection bias, achieving descriptive representation *necessitates* giving different probabilities to different groups, thereby permitting manipulability. In other words, *no* selection algorithm can achieve representation while eliminating manipulation incentives. This motivates our research question:

Research question: What aspects of the selection process can we adjust in practice to *limit* agents' incentives to misreport their features?

Approach. We focus on two main aspects of the selection process that can be changed in practice: *the size of the pool of volunteers* n, and *the choice of selection algorithm*. The intuition for why increasing n could help is simple: as the pool grows, there are more volunteers per available panel seat. For the correct choice of selection algorithm, this could permit the decrease of *all* volunteers' selection probabilities, thereby diluting the potential gains of manipulation.

Among selection algorithms, we consider only algorithms that achieve maximal fairness, because per Example 1.1, manipulation incentives arise from *inequality* in selection probabilities (thus, the goal of equalizing selection probabilities is aligned with limiting manipulation). Specifically, we introduce and study *rounding-based* selection algorithms — a class of maximally fair algorithms that generalizes an algorithm of Flanigan et al. (2020). As discussed further in Section 2, rounding-based algorithms closely reflect those used in practice, but enforce a slightly relaxed notion of representation.

Each rounding-based algorithm optimizes a different *fairness objective*: a function measuring *how fairly* the chance to participate is spread over volunteers. We study several such functions: *Leximin,* the objective most commonly used in real-world panel selection (Flanigan et al. 2021); *Nash Welfare*, which has known fairness and transparency properties and is available online for practical use (Flanigan, Kehne, and Procaccia 2021); and all ℓ_p norms, which we newly introduce to the citizens' assembly setting.

Results and Contributions. (1) Manipulation model. Our first contribution is to formally model three realistic manipulation incentives in the assembly selection context: increasing one's own probability of selection, changing someone else's, and — as we saw in Example 1.1 — misappropriating seats from other groups. **(2) Impossibilities for existing algorithms.** We then show that, somewhat alarmingly, the state-of-the-art objectives *Leximin* and *Nash Welfare* are *arbitrarily manipulable* on multiple of these counts. Even as n grows large, they permit agents to gain *probability 1* by misreporting, and they allow coalitions to misappropriate a constant fraction of the panel seats. These lower bounds give a key insight: fairness objectives are manipulable when they permit some agents to receive very high selection probabilities. **(3) An optimal selection algorithm.** Motivated by this finding, we study ℓ_p norms, which heavily penalize high probabilities due to their strong convexity. We show that even when agents can costlessly misreport any vector of features, the manipulability of the ℓ_p-norm declines in n at a rate $n^{-(1-1/p)}$, a rate which holds for all three notions of manipulability. We further show that *any selection algorithm* must suffer manipulability at least $\Omega(1/n)$; as $p \to \infty$, our upper bound approaches this lower bound, implying that the ℓ_∞ norm — the objective that minimizes the maximum selection probability — achieves optimal convergence. As a bonus, our analysis handles coalitions of size up to $\Theta(n)$. **(4) Empirical results.** We complement these theoretical results with experiments in eight real-world panel selection datasets. Our empirical results closely track our theory, showing that *Leximin* and *Nash Welfare* suffer high manipulability even as n grows, while the manipulability of the ℓ_2 and ℓ_∞ norms declines quickly.

[1] These numbers are based on a real-world panel selection task (instance *sf-e* in our empirical analysis).

[2] More generally, there are clear patterns across real-world instances of which groups tend to be most underrepresented among volunteers (e.g., those with less education).

2 Model

2.1 Foundations of selection algorithms

Selection algorithms solve the *panel selection task*, where the goal is to select a panel of k agents from the pool of n agents. This panel must be descriptively representative with respect to a predefined set of *features* F, where each feature $f \in F$ takes on a *value* $v \in V_f$. For example, the feature $f = \textit{gender}$ might have values $V_{\textit{gender}} = \{\textit{non-binary, female, male}\}$. Descriptive representation is generally required to hold for *all* feature-value pairs (f, v) for $f \in F, v \in V_f$; we summarize all such pairs as $FV := \bigcup_{f \in F} V_f$. A representative panel includes a $p_{(f,v)}$ agents with value v for feature f, where $p_{(f,v)}$ the fraction of the underlying population with value v for feature f. Let these *population rates* be $p := (p_{(f,v)} | (f, v) \in FV)$.

An *instance* of the panel selection task is then defined by a set of population rates p; a desired panel size k; and the *pool* N, which is defined by all n agents' *true* values of each feature. To define these values, we let each feature operate as a function $f : [n] \to V_f$, so $f(i)$ is i's value for feature f. These values are summarized in i's *feature vector* $w(i) = (f(i) | f \in F)$. The *pool* of volunteers $N := (w(i) | i \in [n])$ is then an n-tuple containing all agents' feature vectors. We let $\mathcal{W} := \prod_{f \in F} V_f$ be the collection of all possible feature-vectors (i.e., all possible *intersections* of feature-value pairs). A generic feature vector is $w \in \mathcal{W}$. We will often reason only about *fractional composition* of a pool N, called $\boldsymbol{\nu}(N)$. This vector is indexed by feature-vector, with w-th entry $\nu_w(N) := |\{i \in [n] : w(i) = w\}|/|N|$ representing the fraction of the pool with vector w.

In practice, organizers must rely on agents to *report* their feature vectors. Agent i's *reported* feature vector is denoted $\tilde{w}(i) \in \mathcal{W}$; in general, we will use tilde $\tilde{\cdot}$ throughout the paper to distinguish reported values from true values. The *reported* pool is then denoted as $\tilde{N} = (\tilde{w}(i) | i \in [n])$. In an instance p, k, N, a *selection algorithm* $\mathcal{A}$ actually receives as input $p, k, \tilde{N}$, and must map it to a panel $K \subseteq \tilde{N}$.

In the next subsection, we will formally define three motives with which an agent might misreport their feature vector. All these motives revolve around controlling a particular resource: *selection probability*. Agent i's selection probability is $\mathbb{P}[i \in K]$, the probability i is chosen for the panel. We define $\pi_i^{\mathcal{A}}(p, k, \tilde{N})$ to be the selection probability given to agent i by algorithm $\mathcal{A}$ on input $p, k, \tilde{N}$. Accordingly, the vector of agents' selection probabilities is $\pi^{\mathcal{A}}(p, k, \tilde{N})$. Since p and k are taken to be known by the algorithm, we simply write $\pi^{\mathcal{A}}(\tilde{N})$. A generic vector of selection probabilities is π. Note that there are k available seats for n people, so the average selection probability over agents must be k/n.

2.2 Manipulation of selection algorithms

In the game we study, we permit all agents to costlessly misreport any feature vector in $\mathcal{W}$. We assume that agents report their feature vector $\tilde{w}(i)$ with knowledge of the entire instance p, k, N, plus full access to the selection algorithm.[3] While the assumption that agents exactly know the true pool N is slightly adversarial, our study of simple manipulation heuristics in Section 5 will shed light the potential for manipulation using less detailed information about the pool.

We do not commit to a specific utility function for agents, because they might manipulate with a variety of different goals. Instead, we define the three measures of manipulability below, each corresponding to a different motive: the *internal* manipulability $\text{MANIP}_{\text{int}}$ captures how much a coalition can increase the selection probability of its members; the *external* manipulability $\text{MANIP}_{\text{ext}}$ captures how much a coalition can harm a non-member; and the *composition* manipulability $\text{MANIP}_{\text{comp}}$ captures how many seats (in expectation) a coalition can misappropriate from any feature-value group. We denote a coalition as C, and we let N_{-C} denote the pool with the feature vectors of $i \in C$ removed. In instance p, k, N, the manipulability of $\mathcal{A}$ by any coalition of size c is defined, per notion, as follows, where $\divideontimes := \max_{C \subseteq [n], |C| = c} \max_{\tilde{\boldsymbol{w}} \in \mathcal{W}^{|C|}}$ is shorthand for taking the worst possible coalition of size c and worst possible strategic reports of its members.

$$\text{MANIP}_{\text{int}}(N, \mathcal{A}, c) := \divideontimes \max_{i \in C} \ \pi_i^{\mathcal{A}}(N_{-C} \cup \tilde{\boldsymbol{w}}) - \pi_i^{\mathcal{A}}(N),$$

$$\text{MANIP}_{\text{ext}}(N, \mathcal{A}, c) := \divideontimes \max_{i \notin C} \ \pi_i^{\mathcal{A}}(N_{-C} \cup \tilde{\boldsymbol{w}}) - \pi_i^{\mathcal{A}}(N),$$

$$\text{MANIP}_{\text{comp}}(N, \mathcal{A}, c) :=$$

$$\divideontimes \max_{(f,v) \in FV} \sum_{i : f(i) = v} \pi_i^{\mathcal{A}}(N_{-C} \cup \tilde{\boldsymbol{w}}) - \sum_{i : f(i) = v} \pi_i^{\mathcal{A}}(N).$$

2.3 Rounding-based selection algorithms

We study the manipulability of a class of selection algorithms which we call *rounding-based* selection algorithms. Each rounding-based algorithm is specified by a convex function $g : [0, 1]^n \to \mathbb{R}$; we will refer to the algorithm defined by function g simply as g. Algorithm g proceeds in two steps: Step 1 computes selection probabilities that minimize g, subject to some constraints; then, Step 2 dependently rounds these probabilities to produce a final panel. Since selection probability is the resource sought by manipulating agents — and the selection probabilities are fully determined in Step 1 — only the Step 1 will be of interest in this paper.

Step 1. Find g-optimal selection probabilities. Given instance p, k, N, in this step the algorithm optimizes g over the polytope $\mathcal{R}(N)$, defined such that $\pi \in \mathcal{R}(N) \iff \pi$ satisfies the following constraints:

$$\sum_{i \in N : f(i) = v} \pi_i = k p_{(f,v)} \ \text{ for all } (f, v) \in FV \tag{C1}$$

$$\sum_{i \in N} \pi_i = k \tag{C2}$$

$$\pi \in [0, 1]^n \tag{C3}$$

[3] It is realistic to assume agents know p and k, and can access the selection algorithm: p is found in census data, and for transparency, k might be public and the selection algorithm would be open-sourced. Assuming agents know N is somewhat adversarial, because in practice, the agents report their features simultaneously; however, this assumption reflects the concern that, by comparing census data and the compositions of past pools, agents could infer who tends to participate, and thus the likely composition of N.

(C1) requires *ex-ante* representation for all feature-value pairs; (C2) requires that the panel is the correct size in expectation (required for sub-routine 2), and (C3) requires π to contain valid probabilities. Formally, in step 1 the algorithm g solves the following convex program:

$$\min_{\pi} \; g(\pi) \quad \text{s.t. } \pi \in \mathcal{R}(N) \qquad \text{(OPT-PROB)}$$

Note that without loss of generality, we can assume that the solution of this convex program assigns the same probability to all agents with the same feature vector, since as any feasible solution can be transformed into such a solution, per the definition of $\mathcal{R}(N)$. We will consider only such solutions throughout the paper.

Step 2: Randomized-rounding. This step intakes the selection probabilities found in the previous step, called π^g, and samples a panel K of size k using the discrepancy-based rounding procedure of Flanigan et al. (2020). For our purposes, the key property of this rounding procedure is that it preserves the selection probabilities π^g; we defer the details of this procedure to Appendix A.1.

Specific choices of g. We instantiate the randomized-rounding algorithms above with several convex functions g — all which, when minimized, tend to make selection probabilities more equal. We analyze two choices of g that serve as benchmarks: *Nash Welfare*, and *Leximin*. Nash Welfare is the geometric mean of selection probabilities:

$$\textsf{nash}(\pi) := -\prod_{i \in [n]} \pi_i.$$

Leximin is not itself strictly a function, but a refinement of the objective *Maximin*, which maximizes the minimum selection probability given to any agent:

$$\textsf{maximin}(\pi) := -\min_{i \in [n]} \pi_i.$$

The *Leximin*-optimal solution is computed iteratively: optimize maximin, fix the minimum entry of that solution as a lower bound on any entry of π, then maximize the second-lowest entry; repeat until all entries are fixed.

Finally, we study all ℓ_p norms for $p > 1$, which measure the distance between π and the vector of exactly equal selection probabilities $(k/n, k/n, \ldots, k/n)$:

$$\ell_p(\pi) := \|\pi - (k/n, \ldots, k/n)\|_p^p.$$

Connections to existing algorithms. With rounding-based algorithms defined, we can now compare them to existing selection algorithms. The most closely-related algorithm is that of Flanigan et al. (2020). Their algorithm computes selection probabilities within $\mathcal{R}$ as in our in Step 1, and then rounds them via the same procedure as in our Step 2. The main difference is that their algorithm manually sets selection probabilities to specific values in Step 1 in a way that ends up satisfying the constraints, while algorithm g within our class sets them by optimizing the function g.

Slightly further afield are the most widely-implemented maximally fair algorithms, as introduced by Flanigan et al. (2021). These algorithms differ from ours only in that they enforce representation slightly differently: instead of *ex ante* representation, they require the satisfaction of hard upper and lower demographic quotas *ex post* (e.g., quotas might require that a panel of 10 people contains between 4 and 6 women). As we show in Proposition A.2, our algorithms are formally equivalent to a continuous relaxation of these quota-based algorithms where agents are *divisible*. Moreover, our rounding-based algorithms do, in fact, achieve a relaxed version of these ex-post quotas: they are guaranteed to produce a panel containing within $\pm|F|$ of $kp_{(f,v)}$ agents with each value v of each feature f (Lemma 9, Flanigan et al. (2020)). This panel is found via a rounding scheme based on a discrepancy theorem due to Beck and Fiala (1981).

3 Leximin and Nash are Highly Manipulable

We begin by analyzing the two objectives most closely tied to practice. Strikingly, Theorem 3.1 shows that both *leximin* and *nash* are extremely manipulable: using either algorithm, an individual agent can gain selection probability 1 by misreporting, and a coalition can *deterministically* misappropriate (approaching) *half* of all panel seats for their own group. The proof of this theorem is found in Appendix B.1; we give a proof sketch below.

Theorem 3.1. *For an arbitrarily large n and for all $c \in [1, k/2)$, there exists an instance p, k, N, $|N| = n$ such that*

$$\begin{aligned} \text{MANIP}_{int}(N, \textsf{leximin}, 1) &= 1 \textit{ and} \\ \text{MANIP}_{int}(N, \textsf{nash}, 1) &= 1; \textit{ moreover,} \\ \text{MANIP}_{comp}(N, \textsf{leximin}, c) &= c \textit{ and} \\ \text{MANIP}_{comp}(N, \textsf{nash}, c) &= c. \end{aligned}$$

Proof sketch. Fix a $c \in [1, k/2)$. All claims are proven by a single instance p, k, N with features f_1, f_2 that take on binary values $\{0, 1\}$ (so the possible feature vectors are $00, 01, 10, 11$). In this instance, we let the population rates of all feature-values be balanced: $p_{f_1,0} = p_{f_1,1} = p_{f_2,0} = p_{f_2,1} = 1/2$. We construct N with the following fractional composition, where ν^* should be thought of as a quantity shrinking in c: $\nu_{00}(N) = \nu_{11}(N) = \nu^*$, $\nu_{10}(N) = 1 - 2\nu^*$, and $\nu_{01}(N) = 0$. We let this pool have some size $|N| = n \geq k^2$, such that its fractional composition can be realized.

First, observe that in this instance, all agents with vector 10 must receive zero selection probability *due to the constraints*: giving them any probability would induce a constraint-violating imbalance in the probability given to agents with $f_1 = 0$ versus $f_2 = 0$, which cannot be rebalanced because the complementary vector 01 does not exist in N. This suggests a manipulation strategy: an agent with 10 could misreport 01, thereby permitting greater fairness by allowing agents with 10 to receive some probability.

Let i with $w(i) = 10$, and define $\tilde{N} := N_{-i} \cup \{01\}$ as the pool resulting from i using the proposed strategy. In instance $p, k, \tilde{N}$, agents with 10 can receive probability; the catch is that, for every unit of probability given to such an agent, a unit must also be given to i, meaning that i must receive $|N|/2$ times the probability of any agent with 10. The key observation is that both *leximin* and *nash* prioritize ensuring the *minimum* probability is not too small, with little consideration for what happens to the highest probability. For this reason, both algorithms give i selection probability

1 in the instance $p, k, \tilde{N}$. i has gained probability 1 by misreporting, implying the bounds on $\text{MANIP}_{\text{int}}(N, \textit{leximin}, 1)$ and $\text{MANIP}_{\text{int}}(N, \textit{nash}, 1)$. This argument extends to an entire coalition of $c < k/2$ such agents, implying the bounds on $\text{MANIP}_{\text{comp}}(N, \textit{leximin}, c)$ and $\text{MANIP}_{\text{comp}}(N, \textit{nash}, c)$. □

Takeaway: strongly convex objectives. The key takeaway from this proof is that objectives that do not penalize high selection probabilities can be highly manipulable. A natural class of objectives that *do* penalize high probabilities are *strongly convex* objectives — we formalize this intuition in Proposition B.1. This insight suggests that in future study of selection algorithms, it may be desirable to focus on such objectives. This finding also motivates our focus on ℓ_p norms — a natural class of strongly-convex objectives.

4 ℓ_p-Norms Approach Optimal Manipulability as $p \to \infty$

We now present upper-bounds on all three measures of manipulability for all rounding-based algorithms ℓ_p with $p > 1$. These upper bounds will hold for any instance whose pool satisfies Assumption 4.1, which conceptually requires that the pool has a minimal level of feature vector richness.

Assumption 4.1 (Pool richness)**.** N contains some set of feature-vectors $\mathcal{W}^* \subseteq \mathcal{W}$ such that

1. there is a constant $\kappa^* > 0$ such that $\nu_w(N) \geq \kappa^* + k/n$ for all $w \in \mathcal{W}^*$, and
2. $\mathcal{R}(N)$ contains a solution π^* such that $\pi_i = 0$ for all $i : w(i) \notin \mathcal{W}^*$.

This assumption is likely to hold in practice; in fact, due to how the pool is sampled, *every* feature-vector group's presence in the pool should grow approximately linearly in n. We expand on this in Appendix C.1. Also, note that the pool used to prove Theorem 3.1 satisfies Assumption 4.1 (with $\mathcal{W}^* = \{00, 11\}$), thus demonstrating a genuine gap between the manipulability of all ℓ_p norms and *leximin*, *nash*.

Theorem 4.2. *Let $p > 1$, and let N be any pool of size n satisfying Assumption 4.1 with $\mathcal{W}^*, \kappa^*, \pi^*$. Let $\kappa \in (0, \kappa^*)$; then, for any coalition size $c \leq \kappa n$, we have that*

$$\text{MANIP}_{\textit{int}}(N, \ell_p, c) \in O\left(k/n^{1-1/p}\right),$$
$$\text{MANIP}_{\textit{ext}}(N, \ell_p, c) \in O\left(k/n^{1-1/p}\right), \textit{ and}$$
$$\text{MANIP}_{\textit{comp}}(N, \ell_p, c) \in O\left(ck/n^{1-1/p}\right).$$

Proof. Fix a pool N with $\mathcal{W}^*, \kappa^*, \pi^*$, as in the theorem statement. Fix any coalition $C \subseteq N$ of size $c \leq \kappa n$. Let $\tilde{N} := N_{-C} \cup \{\tilde{w}(i) | i \in C\}$ be the manipulated pool. For convenience, we will again work with feature-vector-indexed objects. We will again use $\nu_w(N)$ as the frequency of w in N. We also define $t_w(\pi) : \sum_{i:w(i)=w} \pi_i$ as the total probability π gives to agents with vector w. Let the vector of these totals be $t(\pi) = (t_w(\pi) | w \in \mathcal{W})$. We can now reformulate the constraints defining $\mathcal{R}(N)$ in terms of the variable t: let $\mathcal{T}(N) \subseteq \mathbb{R}^{|\mathcal{W}|}$ such that $t(\pi) \in \mathcal{T}(N)$ iff

$$\sum_{w:w_f=v} t_w(\pi) = k p_{(f,v)} \text{ for all } (f, v) \in FV \qquad \text{(C1')}$$
$$\sum_w t_w(\pi) = k \qquad \text{(C2')}$$
$$\frac{t_w(\pi)}{n\nu_w(N)} \in [0, 1] \text{ for all } w \in \mathcal{W} \qquad \text{(C3')}$$

Let $\pi^* \in \mathcal{R}(N)$ be the feasible solution assumed to exist by Assumption 4.1. Then, construct the vector $\tilde{\pi}$ as follows:

$$\tilde{\pi}_i = t_{w(i)}(\pi^*) \,/\, n\nu_{w(i)}(\tilde{N}) \quad \text{for all } i \in N.$$

What this definition effectively does is maintains the *total* probability assigned to each vector group from π^* to $\tilde{\pi}$, spreading it over the potentially changing number of voters with that vector from pool N to $\tilde{N}$. Formalizing this:

Claim 1: For all $w \in \mathcal{W}$, $t_w(\pi^*) = t_w(\tilde{\pi})$. *Proof:*

$$t_w(\tilde{\pi}) = \sum_{i:w(i)=w} \tilde{\pi}_i = \sum_{i:w(i)=w} \frac{t_w(\pi^*)}{n\nu_w(\tilde{N})} = t_w(\pi^*).$$

Claim 2: $\tilde{\pi} \in \mathcal{R}(N)$. *Proof:* We prove this by equivalently showing that $t(\tilde{\pi}) \in \mathcal{T}(\tilde{N})$. The satisfaction of constraints C1' and C2' follow from Claim 1. Moreover, by definition $\frac{t_w(\tilde{\pi})}{n\nu_w(\tilde{N})} \geq 0$ for all w. Then, to show C3' it just remains to show that $\frac{t_w(\tilde{\pi})}{n\nu_w(\tilde{N})} \leq 1$ for all w:

$$\frac{t_w(\tilde{\pi})}{n\nu_w(\tilde{N})} = \frac{t_w(\pi^*)}{n\nu_w(\tilde{N})} \leq \frac{t_w(\pi^*)}{n(\nu_w(N) - \kappa)}$$
$$\leq \frac{t_w(\pi^*)}{n(\kappa^* + k/n - \kappa)} \leq \frac{k}{k + n(\kappa^* - \kappa)} \leq 1.$$

Now, we will show that the vectors of probabilities π^*, $\tilde{\pi}$ have maximum entry on the order $1/n$:

Claim 3: $\|\pi^*\|_\infty \leq {}^{k}/_{\kappa^* n}$ and $\|\tilde{\pi}\|_\infty \leq {}^{k}/_{(\kappa^* - \kappa)n}$. *Proof:* For all i with $w(i) \notin \mathcal{W}^*$, $\pi^*_i = \tilde{\pi}_i = 0$ by definition. For i with $w(i) \in \mathcal{W}^*$, we have that

$$\pi^*_i = \frac{t_w(\pi^*)}{n\nu_w(N)} \leq \frac{k}{n\kappa^*} \text{ and } \tilde{\pi}_i = \frac{t_w(\pi^*)}{n\nu_w(\tilde{N})} \leq \frac{k}{n(\kappa^* - \kappa)}.$$

Now, we relate the infinity-norms of any feasible solution and the ℓ_p-optimal solution of OPT-PROB:

Claim 4: For any $\pi \in \mathcal{R}(N)$, $\|\pi^{\ell_p}(N)\|_\infty \leq n^{1/p}\|\pi\|_\infty$. *Proof:* By the optimality of $\pi^{\ell_p}(N)$, we have that $\ell_p(\pi^{\ell_p}(N)) \leq \ell_p(\pi(N))$. Then, using properties of norms, and the triangle inequality (twice), we obtain that

$$\|\pi^{\ell_p}(N)\|_\infty \leq \ell_p(\pi^{\ell_p}(N)) + \|k/n1\|_p \leq \ell_p(\pi) + \|k/n1\|_p$$
$$\leq \|\pi\|_p + 2\|k/n1\|_p \leq n^{1/p}\|\pi\|_\infty + 2kn^{-\frac{p-1}{p}}.$$

Using that $\pi^* \in \mathcal{R}(N)$, $\tilde{\pi} \in \mathcal{R}(\tilde{N})$, Claims 3 and 4 together imply that $\|\pi^{\ell_p}(N)\|_\infty \leq k/(\kappa^* n^{1-1/p}) + 2k/n^{1-1/p}$ and likewise, $\|\pi^{\ell_p}(\tilde{N})\|_\infty \leq k/((\kappa^* - \kappa)n^{1-1/p}) + 2k/n^{1-1/p}$. Using that the entries of all π are nonnegative, it follows that

$$\|\pi^{\ell_p}(\tilde{N}) - \pi^{\ell_p}(N)\|_\infty \leq \left(\frac{1}{\kappa^* - \kappa} + 2\right) \frac{k}{n^{1-1/p}}. \quad (1)$$

We've now shown an upper bound on how many any i's probability changes between pool N and pool $\tilde{N}$. This immediately implies the upper bounds on $\text{MANIP}_{\text{int}}(N, \ell_p, c)$ and $\text{MANIP}_{\text{ext}}(N, \ell_p, c)$. Our upper bound on $\|\pi^{\ell_p}(\tilde{N})\|_\infty$ further implies that post-defection, the members of the coalition can have at most $O(ck/n^{1-1/p})$ total selection probability, giving our upper bound on $\text{MANIP}_{\text{comp}}(N, \ell_p, c)$. □

We now show a lower bound that applies to *any* rounding-based algorithm. It shows that up to constants, the manipulability of ℓ_∞ decreases at the *optimal* rate in n.

Theorem 4.3. *There is some $\eta > 0$ such that there exist pools N of arbitrarily large size n which, for any coalition size $c \leq 5n/64$ and all objectives g, satisfy*

$$\text{MANIP}_{int}(N, g, c) \geq \eta\, k/n, \quad \text{MANIP}_{ext}(N, g, c) \geq \eta\, k/n,$$
$$\text{MANIP}_{comp}(N, g, c) \geq \eta\, ck/n.$$

The same pools also satisfy Assumption 4.1.

The proof is in Appendix C.2 and relies on an example exactly like Example 1.1: there is one binary feature, where v_1 is severely underrepresented in the pool. The bounds arise from agents with v_0 misreporting v_1.

5 Manipulability of Real-World Instances

Now we compare the manipulability of *leximin*, *nash*, ℓ_2 and ℓ_∞ in eight real-world panel selection instances. Instance details are provided in Appendix D.1. We present here two representative instances, called *sf(a)* and *hd*, and defer the rest to Appendix D. The datasets were obtained from groups of assembly organizers based in the UK and US, respectively. Each real-world instance consists of p, k, N. To study how manipulability changes as we increase the pool size, we simply copy the pool, leaving p and k fixed. In each instance, we copy the pool until $n \geq 100k$, as practitioners often specify their target pool size in multiples of k.

We will test our selection algorithms against an *individual* manipulator — that is, we measure how much selection probability any agent can gain by misreporting their feature vector. The most powerful individual manipulator could gain $\text{MANIP}_{\text{int}}(N, \mathcal{A}, 1)$ probability against $\mathcal{A}$ — the quantity to which our theoretical bounds apply. Given the computational difficulty of calculating the optimal manipulation (each agent has $|\mathcal{W}| \in \Omega(2^{|F|})$ possible strategies), we test our algorithms against three practically-motivated heuristic strategies: *OPT-1, MU,* and *HP*, defined below. The results are summarized in Figure 1.

OPT-1: *Optimal misreport of one feature*. An agent playing strategy *OPT-1* reports the feature vector that benefits them most, *subject to misreporting their value for at most one feature*. This strategy, in practice, might correspond to a practical setting in which only a few features cannot be validated. When comparing across algorithms, we think of OPT-1 as a proxy for the optimal individual manipulation. As column 1 of Figure 1 shows, the manipulability of ℓ_2 and ℓ_∞ against OPT-1 declines quickly in n, while *leximin* and *nash* remain arbitrarily susceptible to manipulation. The fact that *leximin* and *nash* are so manipulable *even when agents are willing to misreport only one feature* was not implied by our lower bounds, and shows the findings in our theoretical lower bounds are of practical relevance.

MU: *Most underrepresented*. Let $\eta_{(f,v)}(N) := |\{i | f(i) = v\}|/|N|$ be the fraction of agents with value v for feature f. An agent playing strategy *MU* reports the vector containing the most underrepresented value of each feature f — that is, $\tilde{w}_f := \arg\max_{v \in V_f} p_{(f,v)}/\eta_{(f,v)}(N)$. Again, *leximin* and *nash* are arbitrarily manipulable against *MU*, even for large n. The vulnerability of *leximin* and *nash* here is of especially high practical concern, because the *MU* manipulation strategy is perhaps the most likely to be used in practice by less sophisticated manipulators: it is intuitive and requires only ordinal information about (the only $O(|F|)$ many) feature-value frequencies and no access to the algorithm (in contrast, *OPT-1* and *HP* require algorithm access *and* information about the pool's vector-level composition).

HP: *Highest-Probability*. Another reasonable heuristic a manipulator i might use would be to report the vector $\tilde{w}$ that receives the highest selection probability in the true pool; we call this heuristic *HP*. That this strategy's efficacy declines in n intuitively makes sense: misreporting a vector that is already in the pool means joining a vector group whose size is growing linearly in n (at least in these experiments, where we are duplicating N). This intuition alludes to the insight that the most problematic misreports for suboptimal algorithms are those of vectors that do not already exist in the pool — an intuition supported by both the proof of our lower bound in Theorem 3.1, and the fact that the most underrepresented vector (targeted by the much more effective strategy *MU*) is not in the original pool of any instance we study.

5.1 Extension: manipulability and *selection bias*

While n is much easier to change in practice than the level of self-selection bias (SSB), the SSB could be decreased by a more targeted recruitment process, motivating our study of this would impact the manipulability. We introduce a measure of SSB in an instance, which roughly captures how severely the algorithm must skew selection probabilities to satisfy the constraints:

$$\Delta_{p,k,N} := \max_{(f,v) \in FV} \frac{p_{(f,v)}}{\eta_{(f,v)}(N)} - \min_{(f,v) \in FV} \frac{p_{(f,v)}}{\eta_{(f,v)}(N)}$$

Figure 2(a) shows that this measure of SSB is highly predictive of manipulability: across instances, the manipulation gain of OPT-1 (scaled by k/n, for standardization) against ℓ_∞ corresponds closely with instances' $\Delta_{p,k,N}$ values, as listed in the figure legend. Proceeding with this measure, we evaluate the impact of decreasing it in two ways. First, in Figure 2(b), we decrease the SSB smoothly by *interpolating* between the original pool N and the "nearest" (by Euclidean distance) pool N' with $\Delta_{p,k,N'} = 0$. Second, in Figure 2(c), we decrease the SSB by successively dropping features from the instance in decreasing order of their *feature-level* SSB, defined as $\Delta_{p,k,N}$ restricted to the values of a given feature. Using either approach, in *sf(a)*, the manipulability of all algorithms except *leximin* against *OPT-1* drops quickly, while *leximin* remains manipulable until extremely low levels of SSB are reached. We defer the details of these methods, plus results for the remaining instances, to Appendix D.5.

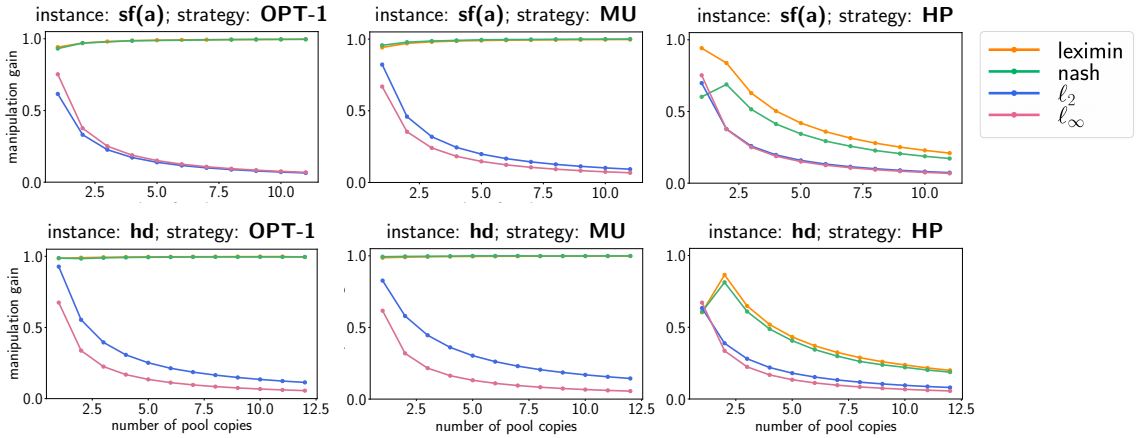


Figure 1: Rounding-based algorithms *leximin*, *nash*, ℓ_2, and ℓ_∞ versus each manipulation strategy in instances *sf(a)* and *hd*.

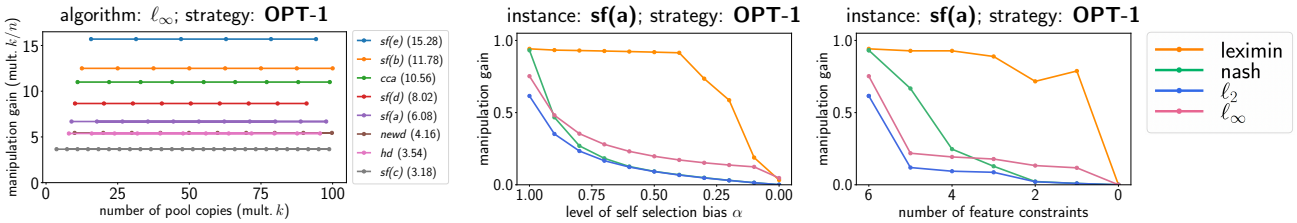


Figure 2: The impact of self-selection bias on the manipulability of *leximin*, *nash*, ℓ_2 and ℓ_∞ by an agent playing *OPT-1* strategy.

6 Discussion

Our work illuminates a tradeoff between two goals: ensuring that no one gets too *little* selection probability (as pursued in the related work (Flanigan et al. 2021)), and ensuring that no one gets too *much* probability (which we show is important for limiting manipulation incentives). *leximin* and *nash* prioritize the first goal but, as we show, perform poorly on the second. In contrast, we show that ℓ_p norms can be optimal in regards to the second goal, but they perform poorly on the first: we find that both ℓ_2 and ℓ_∞ give at least one agent zero probability in all eight instances we study (see Appendix D.7). This begs the question: *is there an objective that both prevents high probabilities (thereby limiting manipulability) as well as low probabilities?* An objective with *optimal* dependency on n for both desiderata at once would give all agents $\Theta(1/n)$ probability.[4]

Another first-order technical extension of this work would be to repeat this analysis within *quota-based* algorithms, as they implement the notion of representation most commonly used (Flanigan et al. 2021). Because the separation between *leximin*, *nash* versus ℓ_p norms is due to fundamental properties of these objectives, we expect them to exhibit roughly similar behavior in quota-based algorithms. However, the combinatorial structure of quotas may make quota-based algorithms much *more* manipulable in the worst case.

[4] $\Theta(1/n)$ is the optimal rate at which manipulability can decline (Theorem 4.3); because any algorithm must divide k probability over n people, the minimum probability can be at most $\Theta(1/n)$.

Even without this extension to quota-based algorithms, our work raises some practical insights. First, it suggests that in general, algorithms permitting high selection probabilities come with risks of manipulability — a property that can be tested in any selection algorithm, maximally fair or not. If one *does* maximize a carefully chosen fairness objective, our work reveals practicable strategies for limiting manipulation incentives: decreasing the SSB (even simply by dropping features that one expects to be highly self-selected), or recruiting a larger pool. Based on our empirical results, even doubling the pool sizes currently used in practice would substantially decrease manipulability.

Beyond the application of assembly selection, our problem is conceptually reminiscent of *strategic classification*, in which agents may misreport their features to increase their probability of receiving a desirable prediction from a machine-learned classifier (Hardt et al. 2016; Dong et al. 2018; Chen, Liu, and Podimata 2020; Ahmadi et al. 2021). Within the strategic classification framework, we can view a selection algorithm as a *constrained* classifier: one which classifies agents as either on or off the panel with some probability based on their features, while satisfying demographic representation constraints on who receives a positive classification. While some existing work is tangentially related (Liu, Garg, and Borgs 2022), to our knowledge this precise problem has not been studied in the strategic classification literature. Our notions of manipulability, and our technical results on the stability of our convex program, may be of interest for this domain.

References

Ahmadi, S.; Beyhaghi, H.; Blum, A.; and Naggita, K. 2021. The strategic perceptron. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, 6–25.

Bansal, N. 2019. On a generalization of iterated and randomized rounding. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, 1125–1135.

Beck, J.; and Fiala, T. 1981. "Integer-making" theorems. *Discrete Applied Mathematics*, 3(1): 1–8.

Bürgerrat. 2023. French citizens' assembly supports assisted dying. https://www.buergerrat.de/en/news/french-citizens-assembly-supports-assisted-dying/.

Chen, Y.; Liu, Y.; and Podimata, C. 2020. Learning strategy-aware linear classifiers. *Advances in Neural Information Processing Systems*, 33: 15265–15276.

Dong, J.; Roth, A.; Schutzman, Z.; Waggoner, B.; and Wu, Z. S. 2018. Strategic classification from revealed preferences. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, 55–70.

Ebadian, S.; Kehne, G.; Micha, E.; Procaccia, A. D.; and Shah, N. 2022. Is Sortition Both Representative and Fair? *Advances in Neural Information Processing Systems*, 35.

Ebadian, S.; and Micha, E. 2023. Boosting Sortition via Proportional Representation. Manuscript.

Flanigan, B.; Gölz, P.; Gupta, A.; Hennig, B.; and Procaccia, A. D. 2021. Fair algorithms for selecting citizens' assemblies. *Nature*, 596(7873): 548–552.

Flanigan, B.; Gölz, P.; Gupta, A.; and Procaccia, A. D. 2020. Neutralizing self-selection bias in sampling for sortition. *Advances in Neural Information Processing Systems*, 33: 6528–6539.

Flanigan, B.; Kehne, G.; and Procaccia, A. D. 2021. Fair sortition made transparent. *Advances in Neural Information Processing Systems*, 34: 25720–25731.

Hardt, M.; Megiddo, N.; Papadimitriou, C.; and Wootters, M. 2016. Strategic classification. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, 111–122.

Liu, L. T.; Garg, N.; and Borgs, C. 2022. Strategic ranking. In *International Conference on Artificial Intelligence and Statistics*, 2489–2518.

Participedia. 2023. https://participedia.net/search?selectedCategory=case&recruitment_method=random,stratified.

A Supplemental materials from Section 2

A.1 Details of *randomized rounding* step of rounding-based algorithms

Inputs: The randomized rounding task takes two inputs:

- a vector of marginal probabilities, $\pi \in [0,1]^n$ such that $\sum_{i\in[n]} \pi_i = k$, and
- an $|FV| + 1 \times n$ matrix H, which can be seen as the binary matrix defining the adding up constraints in OPT-PROB. That is, each column of H corresponds to an agent, and each row (except the last) corresponds to a different feature-value $(f, v) \in FV$. The ith column has 1s in rows corresponding to feature-values possessed by i, and 0s elsewhere. The last row corresponds to the adding up constraints, and so contains a 1 in every column.

Task: The goal is to round the entries of π into a vector $\tilde{\pi} \in \{0,1\}^n$ such that the three criteria below are satisfied. This rounding procedure will be randomized, so $\tilde{\pi}$ is a random variable. Conceptually, $\tilde{\pi}$ will encode the selected panel K, where $\tilde{\pi}_i = 1 \iff i \in K$.

- The adding up constraint is deterministically preserved:

$$\sum_{i\in[n]} \tilde{\pi}_i = k \text{ with probability 1}$$

- The representation constraints satisfied by the original selection probabilities are deterministically satisfied within a relaxation of $|F|$:

$$\sum_{i:f(i)=v} \tilde{\pi}_i \in \sum_{i:f(i)=v} \pi_i \pm |F| \text{ for all } (f,v) \in FV \text{ with probability 1}$$

- The selection probabilities in π are preserved:

$$\mathbf{E}[\tilde{\pi}_i] = \pi_i \text{ for all } i \in [n]$$

Algorithm (RANDOMIZED-ROUND): This rounding algorithm is exactly the algorithm used to prove Lemma 3 in Flanigan et al. (2020). We outline their key arguments here, rephrased in our notation.

Lemma A.1 (Lemma 9 in (Flanigan et al. 2020))**.** *Let $(\pi_i)_{i\in[n]}$ be any collection of variables in $[0,1]$ such that $\sum_{i\in[n]} \pi_i = k$. Then, we can efficiently compute a deterministic 0/1 rounding $(\tilde{\pi}_i)_{i\in[n]}$ such that $\sum_{i\in[n]} \tilde{\pi}_i = k$ and such that, for each feature-value pair (f, v),*

$$\sum_{i:f(i)=v} \tilde{\pi}_i \in \sum_{i:f(i)=v} \pi_i \pm |F|.$$

The proof of this lemma is based on discrepancy theorem by Beck and Fiala (1981), and crucially relies on the fact the underlying matrix H is relatively sparse: that is, each agent i has only $|F| + 1$ 1s in their column of the H matrix.

The above lemma implies a *deterministic* rounding procedure satisfying only criteria 1 and 2 above. To transform this deterministic rounding procedure into a randomized rounding procedure satisfying criteria 3, as do Flanigan et al., we can apply Theorem 1.2 from (Bansal 2019), which does exactly the needed transformation. We outsource the (relatively straightforward) details of applying this theorem in our setting to Lemma 3 in Flanigan et al. (2020).

A.2 Relationship between rounding-based and quota-based algorithms

Conceptually, OPT-PROB is equivalent to the *relaxation* of quota-based algorithms in which all agents are treated as divisible (i.e., a panel can contain fractional agents). We formalize this here now, defining all relevant programs and relaxations, and then proving the equivalence in Proposition A.2.

Definition of quota-based algorithms: OPT-QUOTA. To begin, we first formally specify the optimization solved by a quota-based algorithm, as used in practice and studied in Flanigan et al. (2021). At a high level, these algorithms differ from Equation (OPT-PROB) by requiring representation in a different way: they impose upper and lower *quotas* on all (f, v), which impose some tolerance of error around each $p_{(f,v)} \cdot k$ that must satisfied deterministically by the chosen panel K. Formally, for all $(f, v) \in FV$, a *lower quota* is $\ell_{(f,v)}$, an upper quota is $u_{(f,v)}$, and the chosen panel K is sampled from the *panel distribution* ρ resulting from the optimization program below. We let $\mathcal{K}$ be the collection of all *feasible panels*, that is, all subsets of $[n]$ satisfying the following two constraints:

$$\mathcal{K} := \Big\{ K : |K| = k \quad \wedge \sum_{i:f(i)=v} \mathbf{1}(i \in K) \in \big[\ell_{(f,v)}, u_{(f,v)}\big] \text{ for all } (f,v) \in FV \Big\}.$$

Implicitly, the panel distribution ρ implies selection probabilities π: π_i is equal to the probability of choosing any panel containing i, as defined by p. We encode this constraint in the optimization problem below:

$$\min_{\pi\in[0,1]^n,\ p\in[0,1]^{|\mathcal{K}|}} g(\pi) \quad \text{s.t.} \quad \sum_{K\in\mathcal{K}} \rho_K = 1 \quad \wedge \quad \pi_i = \sum_{K\in\mathcal{K}} \rho_K \cdot \mathbf{1}(i \in K) \ \text{ for all } i \in [n] \tag{OPT-QUOTA}$$

***Continuous relaxation* of quota-based algorithms: OPT-QUOTA-CONTINUOUS.**
Now, we define a version of OPT-QUOTA in which individuals are treated as divisible. Then, a panel $X \in [0,1]^n$ is a vector of length n, whose i-th entry x_i specifies the fraction of agent i included in panel X. Then, the set of feasible panels is the following, uncountable infinite set:

$$\mathcal{X} := \Big\{X : \sum_{i\in[n]} x_i = k \quad \wedge \sum_{i:f(i)=v} x_i \in \left[\ell_{(f,v)}, u_{(f,v)}\right] \ \text{ for all } (f,v) \in FV\Big\}.$$

Then, for variables π and *panel density function* ρ, we optimize

$$\min_{\pi\in[0,1]^n,\ \rho} g(\pi) \quad \text{s.t.} \quad \int_{X\in\mathcal{X}} \rho_X dX = 1 \quad \wedge \quad \pi_i = \int_{X\in\mathcal{X}} \rho_X\, x_i dX \ \text{ for all } i \in [n] \tag{OPT-QUOTA-CONTINUOUS}$$

Generalized version of rounding-based algorithms: OPT-PROB-RANGE.
Here, we define a slightly generalized version of OPT-PROB, in which the representation targets are replaced with ranges (where we should think of this range encompassing the exact representation target in OPT-PROB, so $kp_{(f,v)} \in [\ell_{(f,v)}, u_{(f,v)}]$).

$$\min_{\pi\in[0,1]^n} g(\pi) \quad \text{s.t.} \quad \sum_{i:f(i)=v} \pi_i \in [\ell_{(f,v)}, u_{(f,v)}] \ \text{ for all } (f,v) \in FV \quad \wedge \quad \sum_{i\in[n]} \pi_i = k \tag{OPT-PROB-RANGE}$$

Formal equivalence of OPT-PROB-RANGE and OPT-QUOTA-CONTINUOUS. Conceptually, what this shows is that for any quotas $\ell_{(f,v)}, u_{(f,v)}$ imposed in quota-based algorithms, maximizing our fairness objective while treating people as *divisible* is equivalent — from the perspective of selection probabilities — to solving our rounding-based optimization problem with the same representation target ranges. To realize exactly OPT-PROB, one could run a quota-based algorithm with divisible agents and $\ell_{(f,v)} = u_{(f,v)} = kp_{(f,v)}$.

Proposition A.2. *π is feasible in* OPT-PROB-RANGE $\iff$ *there exists a panel density function ρ over $\mathcal{X}$ which realizes π in* OPT-QUOTA-CONTINUOUS.

Proof. (Forward direction): *π is feasible in* OPT-PROB $\implies$ *there exists a panel density function ρ over $\mathcal{X}$ which realizes π:*
Fix a feasible π. Define a panel X^* such that $x^*_i = \pi_i$ for all $i \in [n]$. By the fact that π satisfies the constraints in OPT-PROB-RANGE, it follows immediately that $X^* \in \mathcal{X}$. Place all the mass in ρ on X^*, so $\rho_{X^*} = 1$ and $\rho_X = 0$ for all $X \in \mathcal{X} \setminus \{X^*\}$. Then, by definition, π is realized by this ρ, because for all $i \in [n]$,

$$\pi_i = \int_{X\in\mathcal{X}} \rho_X x_i dX = x_i.$$

(Reverse direction): *ρ is a valid density function over $\mathcal{X}$ and implies π $\implies$ π is feasible in* OPT-PROB-RANGE.
Fix a valid ρ and let it imply π. Then, we can confirm that π satisfies the constraints of OPT-PROB:

$$\sum_{i\in[n]} \pi_i = \sum_{i\in[n]} \int_{X\in\mathcal{X}} \rho_X x_i dX = \int_{X\in\mathcal{X}} \rho_X \sum_{i\in[n]} x_i dX = \int_{X\in\mathcal{X}} \rho_X k dX = k.$$

For any $(f,v) \in FV$, let $\sum_{i:f(i)=v} x_i = r_{(f,v)}$.

$$\sum_{i:f(i)=v} \pi_i = \sum_{i:f(i)=v} \int_{X\in\mathcal{X}} \rho_X x_i dX = \int_{X\in\mathcal{X}} \rho_X \sum_{i:f(i)=v} x_i dX = \int_{X\in\mathcal{X}} \rho_X r_{(f,v)} dX$$
$$\in \left[\ell_{(f,v)}, u_{(f,v)}\right].$$

□

B Supplemental materials from Section 3

B.1 Proof of Theorem 3.1

Proof. This result is most naturally proven using feature vector-indexed analogs of our standard agent-indexed objects, so we define them now: we use $\nu_w(N) := |\{i : i \in [n], w(i) = w\}|/|N|$ to denote the fraction of the pool N containing feature vector w, with $\boldsymbol{\nu}(N) = (\nu_w | w \in \mathcal{W})$. Noting that all reasonable objectives (including those considered here) will give all agents with the same vector the same selection probability, we will use $q_w(N)$ to denote the selection probability given to each individual agent with vector w, i.e., for all $i \in N : w(i) = w$, $q_w(N) = \pi_i$. We summarize these selection probabilities in the vector $\boldsymbol{q}(N)$.

Reformulation of optimization problem. We reformulate our optimization problem in terms of the variables q_w here, for both objectives. First our feasible set of values of $q_w | w \in \mathcal{W}$, call it $\mathcal{Q}$, is defined by the following constraints, analogs of those defining $\mathcal{R}$: $\boldsymbol{q} \in \mathcal{Q} \iff \boldsymbol{q}$ satisfies

$$\sum_{w : w_f = v} \nu_w(N) q_w(N) = p_{(f,v)} \cdot k/n \text{ for all } (f,v) \in FV \quad \wedge \quad \sum_w \nu_w(N) q_w(N) = k/n \quad \wedge \quad \boldsymbol{q}(N) \in [0,1]^{|\mathcal{W}|}. \tag{2}$$

Now, to defining our full optimization problems: first, recalling that *leximin* is just a refinement of maximin,

$$\text{maximin}(p,k,N): \qquad \max_{q \in [0,1]^{\mathcal{W}_N}} \min_{w \in \mathcal{W}_N} q_w(N) \quad \text{s.t.} \quad \boldsymbol{q} \in \mathcal{Q}.$$

For *nash*, we equivalently analyze the log of the geometric mean, whose optimizer is the same as that of the geometric mean:

$$\textit{nash}(p,k,N): \qquad \min_{q \in [0,1]^{\mathcal{W}_N}} \sum_{w \in W_N} -\nu_w(N) \log(q_w(N)) \quad \text{s.t.} \quad \boldsymbol{q} \in \mathcal{Q}.$$

Instance. Fix a $\delta \in [1, k/2)$. All four claims will be proven via the same class of instances (parameterized by δ), which has two features $F = \{f_1, f_2\}$ with binary values in $\{0,1\}$, and as such, $\mathcal{W}$ contains the feature vectors $00, 01, 10, 11$. Now, to define this instance p, k, N: let the population rates be $p_{f_1,0} = p_{f_2,0} = 1/2$. Let $k \geq 2$. Fix a pool N of size $n \geq k^2$ where n is a multiple of both $\delta/(2k)$ and $1 - \delta/k$, with the following composition: $\nu_{00} = \nu_{11} = \nu^* = \delta/(2k)$, $\nu_{10} = 1 - 2\nu^*$, and $\nu_{01} = 0$.

Optimal selection probabilities in instance (with true pool). Now, we characterize the *leximin* and *nash*-optimal selection probabilities in this instance with the true pool. They are simple, because they are essentially determined by the constraints: notice that for any n, the constraints require $q_{10}(N) = 0$, because any probability mass added to $\nu_{(10)}$ will induce imbalance in the amount of probability given to $f_1 = 0$ versus $f_2 = 0$, a violation of the constraints that $\rho_{f_1=0} = \rho_{f_2=0} = 1/2$ that cannot be counteracted because the complementary vector (01) does not exist in the pool. Also, because vectors 00 and 11 are completely symmetric in the instance, they must receive identical selection probabilities. Thus, $q_{00}(N) = q_{11}(N)$; by the adding up constraint, we have that $\nu^* q_{00}(N) + \nu^* q_{11}(N) = k/n$, implying that $q_{00}(N) = q_{11}(N) = 1/(2\nu^*) \cdot k/n$. To recap, for any n,

$$q_{00}^{\textit{leximin}}(N) = q_{00}^{\textit{nash}}(N) = q_{11}^{\textit{leximin}}(N) = q_{11}^{\textit{nash}}(N) = 1/(2\nu^*) \cdot k/n, \quad q_{10}^{\textit{leximin}}(N) = q_{10}^{\textit{nash}}(N) = 0. \tag{3}$$

Defining the manipulated pool. Now, define the following manipulating coalition C of size $c = k/2 - \delta$ such that $w(i) = 10$ for all $i \in C$ — that is, all agents in the coalition will have true vector 10. They will also all misreport the same vector 01, so $\tilde{w}(i) = 01$ for all $i \in C$. We define the resulting manipulated pool as $\tilde{N} := N_{-C} \cup (\tilde{w}(i) | i \in C)$. Then, in the corresponding $\tilde{\nu}$, we have that $\tilde{\nu}_{00} = \tilde{\nu}_{11} = 1/4$, $\tilde{\nu}_{10} = 1/2 - c/n$, and $\tilde{\nu}_{01} = c/n$.

Optimal selection probabilities in the manipulated pool. Before analyzing any specific objective, we reduce the constraints to be in terms of a single selection probability q_{01}. Beginning with the raw constraints (where all probabilities q_v here are implicitly $q_v(\tilde{N})$, the probabilities in the manipulated pool):

$$\begin{aligned} \nu^* \, q_{00} + c/n \, q_{01} &= 1/2 \cdot k/n \\ \nu^* \, q_{00} + (1 - 2\nu^* - c/n) \, q_{10} &= 1/2 \cdot k/n \\ \nu^* \, q_{00} + c/n \, q_{01} + (1 - 2\nu^* - c/n) \, q_{10} + \nu^* \, q_{11} &= k/n \end{aligned}$$

This system of 3 linear equations and 4 unknowns simplifies to the following expressions, where all the selection probabilities are in terms of q_{01}:

$$q_{00} = q_{11} = \frac{1/2 \cdot k/n - c/n q_{01}}{\nu^*} \qquad \text{and} \qquad q_{10} = \frac{1/2 \cdot k/n - (1/2 \cdot k/n - c/n q_{01})}{1 - 2\nu^* - c/n} = \frac{c/n \cdot q_{01}}{1 - 2\nu^* - c/n}.$$

Handling box constraints. Above, we expressed all agents' selection probabilities in terms of q_{01}. Now, we will show that for all $q_{01} \in [0,1]$, all agents' selection probabilities fall between $[0,1]$ for the parameter settings above. First, this is trivially true for q_{01}. For $q_{00} = q_{11}$, we have that

$$q_{00} = q_{11} = \frac{1/2 \cdot k/n - c/n q_{01}}{\nu^*} = \frac{k - 2(k/2-\delta)q_{01}}{2n \cdot \delta/(2k)} = \frac{k(k-(k-2\delta)q_{01})}{n\delta}$$

Bounding this above and below for all $q_{01} \in [0,1]$:

$$0 \leq \frac{2k}{n} = \frac{k(k-(k-2\delta))}{n\delta} \leq \frac{k(k-(k-2\delta)q_{01})}{n\delta} \leq \frac{k^2}{n\delta} \leq 1.$$

Finally, for q_{10},

$$\frac{c/n}{1-2\nu^*-c/n} \cdot q_{01} = \frac{k/2-\delta}{n(1-2\delta/(2k)) - (k/2-\delta)} = \frac{k/2-\delta}{n(1-\delta/k) - (k/2-\delta)}$$

Bounding this above and below for all $q_{01} \in [0,1]$ (and assuming $k \geq 2$, as is always the case in real panels):

$$0 \leq \frac{k/2-\delta}{n(1-\delta/k)} \leq \frac{k/2-\delta}{n(1-\delta/k)-(k/2-\delta)} \leq \frac{k/2}{n(1-1/2)-k/2} = \frac{k}{n-k} \leq \frac{k}{k^2-k} = \frac{1}{k-1} \leq 1.$$

Now, we've shown that in this instance, the constraints $q_{00} \in [0,1], q_{11} \in [0,1]$, and $q_{10} \in [0,1]$ in Equation (2) will never bind. This means that we have reduced the problem to a single-variable problem of the following form:

$$\min_{q_{01}} g(q_{01}) \quad \text{such that } q_{01} \in [0,1].$$

We now compute the optimizer of this program below for both $g = \textit{leximin}$ and $g = \textit{nash}$, showing that in either case, the optimizer sets $q_{01} = 1$.

Analysis of *leximin*. *leximin* has only one degree of freedom q_{01}, so it will maximize the minimum selection probability, i.e., it will set q_{01} to maximize the following expression:

$$\min\left\{q_{01}, \frac{c/n \cdot q_{01}}{1-2\nu^*-c/n}, \frac{1/2 \cdot k/n - c/n q_{01}}{\nu^*}\right\} \tag{4}$$

We will show that the second term in this minimum is the smallest over the entire domain of q_{01}. First, comparing the second term to the first term in (4), we use that $c \leq k/2$ to show that

$$q_{01} \geq \frac{c/n \cdot q_{01}}{1-2\nu^*-c/n} \iff 1-2\nu^*-c/n \geq c/n \impliedby 1-2\nu^* \geq k/n.$$

Plugging in our parameters, we deduce that $1-2\nu^* = 1-2\delta/(2k) \geq 1-1/2 = 1/2 \geq k/n$, as needed.

Next, comparing the second term to the third term in (4), we deduce that

$$\begin{aligned}
\frac{1/2 \cdot k/n - c/n q_{01}}{\nu^*} \geq \frac{c/n \cdot q_{01}}{1-2\nu^*-c/n} &\iff (1/2 \cdot k/n - c/n \cdot q_{01})(1-2\nu^*-c/n) \geq c/n q_{01} \cdot \nu^* \\
&\iff k \geq \frac{2c \cdot q_{01}(\nu^* + 1 - 2\nu^* - c/n)}{1-2\nu^*-c/n} \\
&\iff k \geq \frac{2c \cdot q_{01}(1-\nu^*-c/n)}{1-2\nu^*-c/n} \qquad \text{(b)}
\end{aligned}$$

Observe that if (b) holds for $q_{01} = 1$, it holds for all $q_{01} \in [0,1]$. Thus, setting $q_{01} = 1$, we deduce the bound in reverse:

$$\begin{aligned}
k \geq \frac{2c(1-\nu^*-c/n)}{1-2\nu^*-c/n} &\iff k \geq \frac{2(k/2-\delta)(1-\delta/(2k)-(k/2-\delta)/n)}{1-2\delta/(2k)-(k/2-\delta)/n} \\
&\iff k(1-\delta/k-(k-2\delta)/(2n)) \geq (k-2\delta)(1-\delta/(2k)-(k-2\delta)/(2n)) \\
&\iff k-\delta-\frac{k(k-2\delta)}{2n} \geq k-\delta/2-\frac{k(k-2\delta)}{2n} - 2\delta + \delta^2/k + 2\delta\frac{k-2\delta}{2n} \\
&\iff -\delta \geq \delta/2 - 2\delta + \delta^2/k + \delta\frac{k-2\delta}{n} \\
&\iff \delta \leq -\delta/2 + 2\delta - \delta^2/k - \delta\frac{k-2\delta}{n}
\end{aligned}$$

Using that $k - 2\delta > 0$ and $n \geq 2k$,

$$\begin{aligned}
&\Longleftarrow \delta \leq -\delta/2 + 2\delta - \delta^2/k - \delta\frac{k-2\delta}{2k} \\
&\iff \delta \leq \delta - \delta^2/k + \delta^2/k \\
&\iff \delta \leq \delta.
\end{aligned}$$

Then, we have that (b) is true for all $q_{01} \in [0, 1]$.

We have shown that the second term of the minimum in (4) is the smallest term over the entire support $q_{01} \in [0, 1]$. Because this term is increasing in q_{01}, the ***leximin*** optimal solution will maximize this term by setting $q_{01} = 1$.

We conclude that in this instance, $q_{01}^{\textit{leximin}}(\tilde{N}) = 1$. That is, on the manipulated pool, ***leximin*** will give all agents in the manipulating coalition probability 1. Given that by Equation (3), $q_{10}^{\textit{leximin}}(N) = 0$ and $w(i) = 10$ for all $i \in C$, it follows that for any $i \in C$, $\pi_i^{\textit{leximin}}(\tilde{N}) - \pi_i^{\textit{leximin}}(N) = 1 - 0 = 1$.

Moreover, we've shown this for any size coalition $c \in [1, k/2)$. Setting $c = 1$ (corresponding setting to $\delta = k/2 - (k/2 - 1)$), this implies that $\text{MANIP}_{\text{int}}(N, \textit{leximin}, 1) = 1$. For generic δ, we conclude that $\text{MANIP}_{\text{comp}}(N, \textit{leximin}, k/2 - \delta) = k/2 - \delta$.

Analysis of *nash*. Repeating the same analysis for Nash, the function Nash maximizes in this instance is

$$\sum_w \nu_w \log(q_w) = 2\nu^* \log\left(\frac{1/2 \cdot k/n - c/nq_{01}}{\nu^*}\right) + (1 - 2\nu^* - c/n)\log\left(\frac{c/n}{1 - 2\nu^* - c/n} \cdot q_{01}\right) + c/n \log(q_{01})$$

This function is concave in q_{01}, so it has a unique maximizer that can be found by the first-order condition: Thus, taking the derivative with respect to q_{01} and setting it to zero, we get that this function is maximized when

$$2\nu^* \cdot \frac{\nu^*}{1/2 \cdot k/n - c/n \cdot q_{01}} \cdot \frac{-c}{n\nu^*} + (1 - 2\nu^* - c/n) \cdot \frac{1 - 2\nu^* - c/n}{c/n \cdot q_{01}} \cdot \frac{c/n}{1 - 2\nu^* - c/n} + \frac{c}{n \cdot q_{01}} = 0.$$

Dividing both sides by c/n and making cancellations,

$$\begin{aligned}
&\iff \frac{-2\nu^*}{1/2 \cdot k/n - c/n \cdot q_{01}} + \frac{(1 - 2\nu^* - c/n)/(c/n)}{q_{01}} + \frac{1}{q_{01}} = 0 \\
&\iff \frac{-2\nu^*}{1/2 \cdot k/n - c/n \cdot q_{01}} + \frac{1 - 2\nu^*}{c/n \cdot q_{01}} = 0 \\
&\iff q_{01} = \frac{k(1 - 2\nu^*)}{2c}
\end{aligned}$$

Plugging in our values for ν^*, c,

$$\begin{aligned}
&\iff q_{01} = \frac{k(1 - 2\delta/(2k))}{2(k/2 - \delta)} \\
&\iff q_{01} = \frac{k - \delta}{k - 2\delta} > 1
\end{aligned}$$

Of course, we have deduced that the unconstrained optimizer places $q_{01} > 0$. By the concavity of the objective, we know that the optimizer is then at $q_{01} = 1$, at the edge of the box constraint.

We conclude that in this instance, $q_{01}^{\textit{nash}}(\tilde{N}) = 1$ — that is, on the manipulated pool, ***nash*** will give all agents in the manipulating coalition probability 1. Given that by Equation (3), $q_{10}^{\textit{nash}}(N) = 0$ and $w(i) = 10$ for all $i \in C$, for all $i \in C$, $\pi_i^{\textit{nash}}(\tilde{N}) - \pi_i^{\textit{nash}}(N) = 1 - 0 = 1$.

Moreover, we've shown this for any size coalition $c \in [1, k/2)$. Setting $c = 1$ (corresponding setting to $\delta = k/2 - (k/2 - 1)$), this implies that $\text{MANIP}_{\text{int}}(N, \textit{nash}, 1) = 1$. For generic δ, we conclude that $\text{MANIP}_{\text{comp}}(N, \textit{nash}, k/2 - \delta) = k/2 - \delta$. □

B.2 Proof of Proposition B.1

Let $k/n\mathbf{1}$ be the n-length vector whose entries are all k/n.

Proposition B.1. *Let g be any strongly convex (with parameter m) that, when unconstrained, is minimized at at $k/n\mathbf{1}$, the point where all agents' selection probabilities are equalized. Let π be a set of marginal probabilities with $q = \max_i \pi_i$. Then,*

$$g(\pi) \geq m/2|q - k/n|^2.$$

Proof. Then, by the definition of strong convexity,

$$\begin{aligned} g(\pi) - g(k/n\mathbf{1}) + \nabla g(k/n\mathbf{1})^T(\pi - k/n\mathbf{1}) \geq m/2\|\pi - k/n\|^2 &= m/2\sum_i(\pi_i - k/n)^2 \\ &\geq m/2|q - k/n|^2 \end{aligned}$$

Noting that $\nabla g(k/n\mathbf{1})^T = 0$,

$$\begin{aligned} \iff g(\pi) - g(k/n\mathbf{1}) &\geq m/2|q - k/n|^2. \\ \implies g(\pi) &\geq m/2|q - k/n|^2. \end{aligned}$$

□

C Supplemental materials from Section 4

C.1 Practical justification of Assumption 4.1

What we need is a a set $\mathcal{W}^*$ of feature vectors within the pool such that each group $w \in \mathcal{W}^*$ grows *linearly in* n (up to the size of the total population) and that this set of vectors is sufficient to permit a feasible solution on their own. We cannot test this assumption directly in our data, since we only see one realized value of n. Thus, we base our discussion here on the statistical properties of the random pool recruitment process. Examining this process, we actually expect something stronger to be true, at least in expectation: *every vector group* to grow linearly in n, up to variance, which we will discuss at the end. This (expected) linear growth is due to how the pool is sampled: invitation recipients are uniformly selected from the population, so at least the *expected* pool composition, over the randomness of the invitation process, should be roughly constant in n (i.e., *all groups* grow linearly in n). We formalize this intuition below with a simple model of the pool formation process, which will also help us more precisely discuss the role of variance.

To model the pool formation process with minimal assumptions, let Y be the entire underlying population, Let $\mathcal{W}_Y$ be the set of all unique feature vectors in the population, γ_w be the fraction of the population with feature vector w, and q_i be the probability that each $i \in Y$ decides to participate conditional on being invited. Let $\bar{q}_w : \frac{1}{|\{i:w_i=w\}|}\sum_{i:w_i=w} q_i$ be the average rate of participation among population members with vector w. Then, in the process of sampling the pool N (with corresponding $\boldsymbol{\nu}(N)$), there are two stages of randomness: that of inviting recipients, and their decision of whether to participate. Regardless of the size of the pool N, $\mathbb{E}[\nu_w(N)] = \gamma_w \bar{q}_w$ for all $w \in \mathcal{W}_Y$ — that is, in expectation, *all vector groups* in the pool are growing linearly in n (and moreover, the randomness in this process consists of Bernoulli draws, so the pool composition should be concentrating around its expectation as n gets large). Variance in this process could be in the q_i values of agents with vector w relative to $\bar{q}_w$; variance in the sampling of who receives letters; and variance in the Bernoulli draws by which people decide whether to participate. Based on this process, variance will mainly be a problem for ensuring linear growth among very small groups, particularly when n is small.

The potential effects of variance in small groups, especially at practical sample sizes, is precisely the motivation for proving our results under Assumption 4.1 — a much weaker requirement than the assumption that *all* groups are growing in n. Under this assumption, we need only that *there some set of vectors yielding a feasible solution* growing linearly in n, rather than *all* vector groups in the pool. For our assumption to be violated, there would need to be *no such set of feature vectors*, corresponding to the unlikely case that *any possible set of feature vectors* supporting a feasible solution contains a group composing only a sliver of the population.

C.2 Proof of Theorem 4.3

Proof. Fix an instance with one binary feature with values 0 and 1; let $p_{f,v_1} = 1/2$; then we know that the total probability given to agents with vector 0 and 1 is $1/2k$. Now, suppose $\nu_0 = 7/8$ and $\nu_1 = 1/8$. The probabilities are then

$$\pi_0 = \frac{1/2k}{7n/8} = \frac{4}{7}k/n, \qquad \pi_1 = \frac{1/2k}{n/8} = 4k/n.$$

We will deal with the largest coalition size $c = 5n/64$ only; the argument for all smaller coalition sizes follows in the same way. We assume that this coalition defects from vector 0 to vector 1. Then, the resulting probability for vector 1 is

$$\tilde{\pi}_1 = \frac{1/2k}{n/8 + c} = \frac{4k}{n + 8c} = 4k/n - \frac{4k * 8c}{n(n+8c)} \geq 4k/n - \frac{32kc}{n^2}$$

Now, we characterize the three types of manipulability. Within the coalition, all members receive probability $\tilde{\pi}_1$ when before they received π_0, so

$$\textsc{manip}_{\text{int}}(N, \mathcal{A}, c) \geq \tilde{\pi}_1 - \pi_0 \geq 4k/n - \frac{32kc}{n^2} - \frac{4}{7}k/n = \frac{k}{n}\left(\frac{24}{7} - \frac{32c}{n}\right) \geq \frac{k}{n}(3 - 2.5) = 1/2 \cdot k/n.$$

By joining group 1, the coalition decreases the existing members' probabilities by making their group more numerous:

$$\begin{aligned}\textsc{manip}_{\text{ext}}(N, \mathcal{A}, c) \geq \pi_1 - \tilde{\pi}_1 = 4k/n - \left(4k/n - \frac{4k * 8c}{n(n+8c)}\right) = \frac{4k * 8c}{n(n+8c)} \geq \frac{32kc}{8n(n+c)} &= \frac{4k \cdot 5n/64}{n(n+5n/64)} \\ &= \frac{5/16 \cdot k}{n(1+5/64)} \\ &= 20/69 \cdot k/n.\end{aligned}$$

Then, because there are c true 0s impersonating 1s, the true seats given to 0 is, in expectation, $k/2$ (the number of seats that must be given to them in expectation, based on the perceived pool), plus however many seats 1-impersonators get in expectation:

$$\textsc{manip}_{\text{comp}}(N, \mathcal{A}, c) \geq (k/2 + c \cdot \tilde{\pi}_1) - k/2 = c \cdot \tilde{\pi}_1 = c\left(4k/n - \frac{4k * 8c}{n(n+8c)}\right) \geq 3ck/n.$$

Set $\eta = k \cdot 20/69$, and the proof is complete. □

D Supplemental materials from Section 5

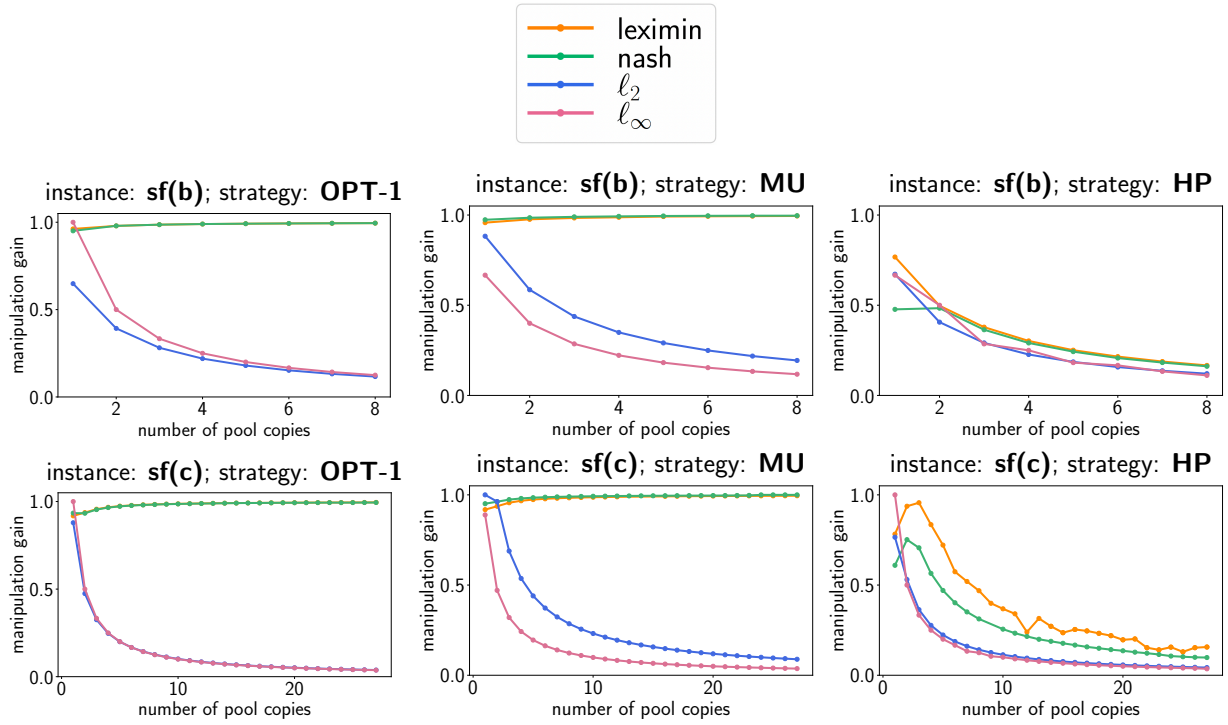
D.1 Panel selection instances

Instance	Organization	n	k	# unique vectors	# features	Δ
sf(a)	Sortition Foundation	312	35	182	6	6.08
sf(b)	Sortition Foundation	250	20	92	6	11.78
sf(c)	Sortition Foundation	161	44	92	7	3.18
sf(d)	Sortition Foundation	404	40	108	6	8.02
sf(e)	Sortition Foundation	1727	110	762	7	15.28
cca	Center for Blue Democracy	825	75	554	4	10.56
hd	Healthy Democracy	239	30	202	7	3.54
newd	New Democracy	398	40	173	6	4.16

Table 1: Overview of real-world instances. Δ is a measure of the self-selection bias in the instance, as defined as Section 5.1.

D.2 Additional instances for Figure 1

Below in Figure 3 we present plots for all 6 other instances, corresponding to those in Figure 1. An interesting aspect of these results results: For instances cca and sf(d), the strategy *MU* is harmful for nearly all agents in the pool under all three algorithms. This is promising for practitioners; although deviating to the feature vector with the most underrepresented feature values is a strategy that is most likely to be used in practice, cca and sf(d) serve as counterexamples where *leximin* and *nash* are not arbitrarily manipulable against *MU*.



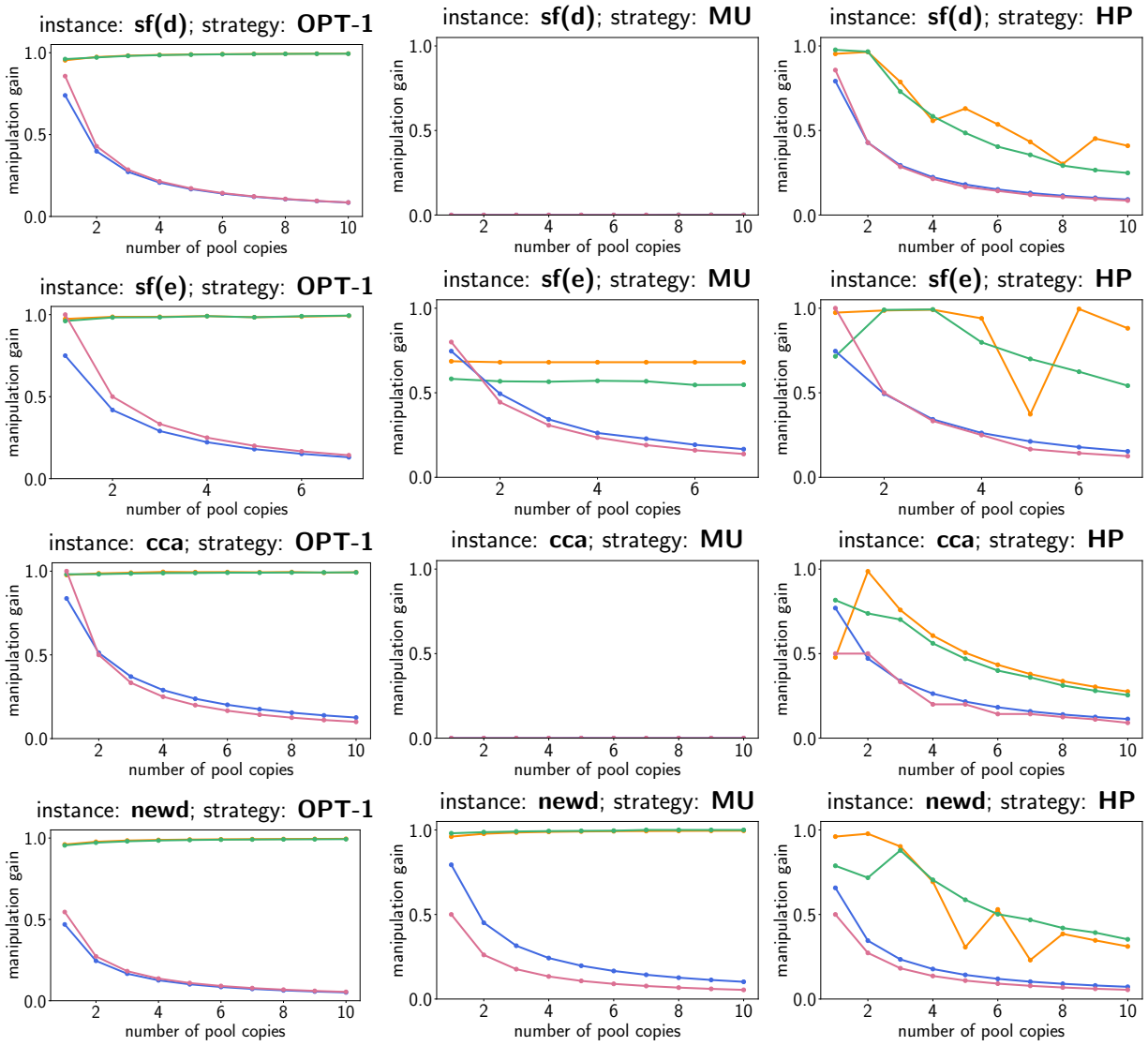


Figure 3: Figures for remaining instances from analysis in Figure 1

.

D.3 (Blank - placeholder to ensure body cross-refs are correct, will remove)

D.4 (Blank - placeholder to ensure body cross-refs are correct, will remove)

D.5 Self-selection bias experiments: methods

Here, we describe the details of the experiments used to produce plots Figure 2(b) and Figure 2(c), and correspondingly, those in Appendix D.6. For both SSB by *interpolating* and SSB by *feature dropping*, we define the precise sequence of instances we test, and then prove that over the sequences of instances induced by either approach, $\Delta_{p,k,N}$ is decreasing (Claim D.1 for interpolation, and Claim D.2.

SSB by *interpolating* (corresponding to Figure 2(b)) Here, we studied how our selection algorithms performed against the OPT-1 strategy over a sequence of pools with decreasing SSB. The pools in this sequence are different convex combinations of two pools: N (the original pool) and pool N', defined as the solution of the convex program below, which finds the pool "closest" (by Euclidean distance) to N that has SSB $\Delta_{p,k,N} = 0$.

$$N' := \underset{N'':|N''|=n}{\arg\max} \|\boldsymbol{\nu}(N) - \boldsymbol{\nu}(N'')\|_2 \quad \text{s.t.} \quad \sum_{w:w_f=v} \nu_w(N'') = p_{(f,v)} \ \text{ for all } (f,v) \in FV.$$

Now, define the sequence of pools $N_0, N_1, \ldots, N_{10}$ in which N_ℓ is defined such that $|N_\ell| = n$ and

$$\nu_w(N_\ell) = (1 - \ell/10) \cdot \nu_w(N) + \ell/10 \cdot \nu_w(N') \qquad \text{for all } w \in \mathcal{W}.$$

In Figure 2(b), the α on the x axis is then the interpolation weight, ranging over $\alpha = (\ell/10)_{\ell \in [10]}$. More formally, across the x axis, we're testing the sequence of instances $p, k, N_0,\ p, k, N_1, \ldots, p, k, N_{10}$.

Claim D.1. *$\Delta_{p,k,N}$ is weakly decreasing over the sequence of instances $p, k, N_0, \ldots, p, k, N_{10}$.*

Proof. Partition the feature-values FV into three exhaustive subsets:

$$FV^{under}(N) := \left\{(f,v) : \frac{p_{(f,v)}}{\eta_{(f,v)}(N)} > 1\right\},\ FV^{over}(N) := \left\{(f,v) : \frac{p_{(f,v)}}{\eta_{(f,v)}(N)} < 1\right\},$$

$$\text{and } FV^{exact}(N) := \left\{(f,v) : \frac{p_{(f,v)}}{\eta_{(f,v)}(N)} = 1\right\}.$$

Observe that for all (f,v), by the constraints defining N', $\eta_{f,v}(N') = p_{(f,v)}$. Then we have that

$$\eta_{f,v}(N_\ell) = (1 - \ell/10)\,\eta_{(f,v)}(N) + \ell/10\eta_{(f,v)}(N') = (1 - \ell/10)\,\eta_{(f,v)}(N) + \ell/10 p_{(f,v)}$$

We can see from this expression that $(f,v) \in FV^{under}(N) \implies (f,v) \in FV^{under}(N_\ell)$ for all $\ell < 10$, and likewise for FV^{under}, FV^{exact}.

Now, let $\ell' > \ell$ for $\ell \in 0 \ldots 9$. We have that for all $(f,v) \in FV^{over}(N)$,

$$\frac{p_{f,v}}{\eta_{(f,v)}(N_\ell)} > \frac{p_{f,v}}{\eta_{(f,v)}(N_{\ell'})}$$

This is seen by the fact that for all $(f,v) \in FV$, the following quantity is decreasing And similarly, for all $(f,v) \in FV^{under}(N)$,

$$\frac{p_{f,v}}{\eta_{(f,v)}(N_\ell)} < \frac{p_{f,v}}{\eta_{(f,v)}(N_{\ell'})}.$$

Observing that if $FV^{under}(N)$ is non-empty (and thus $FV^{over}(N)$ is also non-empty) the feature values that yield the max and min terms in $\Delta_{p,k,N}$ must come from $FV^{under}(N)$ and $FV^{over}(N)$, respectively. Therefore, the difference between the max and the min must be decreasing, and $\Delta_{p,k,N_0}, \Delta_{p,k,N_1}, \ldots, \Delta_{p,k,N_{10}}$ is decreasing. □

SSB by *feature dropping*. In a fixed instance, we define the self-selection bias of a single feature according to $\Delta_{p,k,N}$ *restricted to the values of f*, or formally, as

$$\Delta^f_{p,k,N} := \max_{v \in V_f} p_{(f,v)} \,/\, \eta_{(f,v)}(N) - \min_{v \in V_f} p_{(f,v)} \,/\, \eta_{(f,v)}(N).$$

Now, let the features be ordered in decreasing order of their self-selection bias, so $\Delta^{f_1}_{p,k,N} \geq \Delta^{f_2}_{p,k,N} \geq \cdots \geq \Delta^{f_{|F|}}_{p,k,N}$. We will decrease the self-selection bias by successively drop features from the problem in this order.

When we "drop" a feature f out of the problem, we are formally dropping constraints $\sum_{i:f(i)=v} \pi_i = k p_{(f,v)}$ for all $v \in V_f$ from Equation (OPT-PROB) Accordingly, dropping features corresponds to changing the instance p, k, N by dropping entries of p. Formally, express $p = (p_{(f,v)})_{f \in F, v \in V_f}$. Now, we define a sequence of $p_1, \ldots, p_{|F|}$ where $p_\ell := (p_{(f,v)})_{f \in \{f_\ell \ldots f_{|F|}\}, v \in V_f}$. Then, across the x axis of Figure 2(c) (and all corresponding figures for other instances), we are testing how our selection algorithms perform against the OPT-1 strategy over the sequence instances $p_1, k, N,\ p_2, k, N,\ \ldots p_{|F|}, k, N$.

Claim D.2. *$\Delta_{p,k,N}$ is weakly decreasing over the sequence of instances $p_1, k, N, \ldots p_{|F|}, k, N$.*

Proof. Dropping constraints f, v out of FV can only decrease $\max_{(f,v) \in FV} \frac{p_{(f,v)}}{\eta_{(f,v)}(N)}$ and increase $\min_{(f,v) \in FV} \frac{p_{(f,v)}}{\eta_{(f,v)}(N)}$. □

D.6 Self-selection bias experiments: supplemental empirical results

Additional instances for Figure 2(b). Figure 4 shows tests decreasing self-selection bias by *interpolation* for all remaining instances.

Additional instances for Figure 2(c). Figure 5 shows tests decreasing self-selection bias by *feature dropping* for all remaining instances.

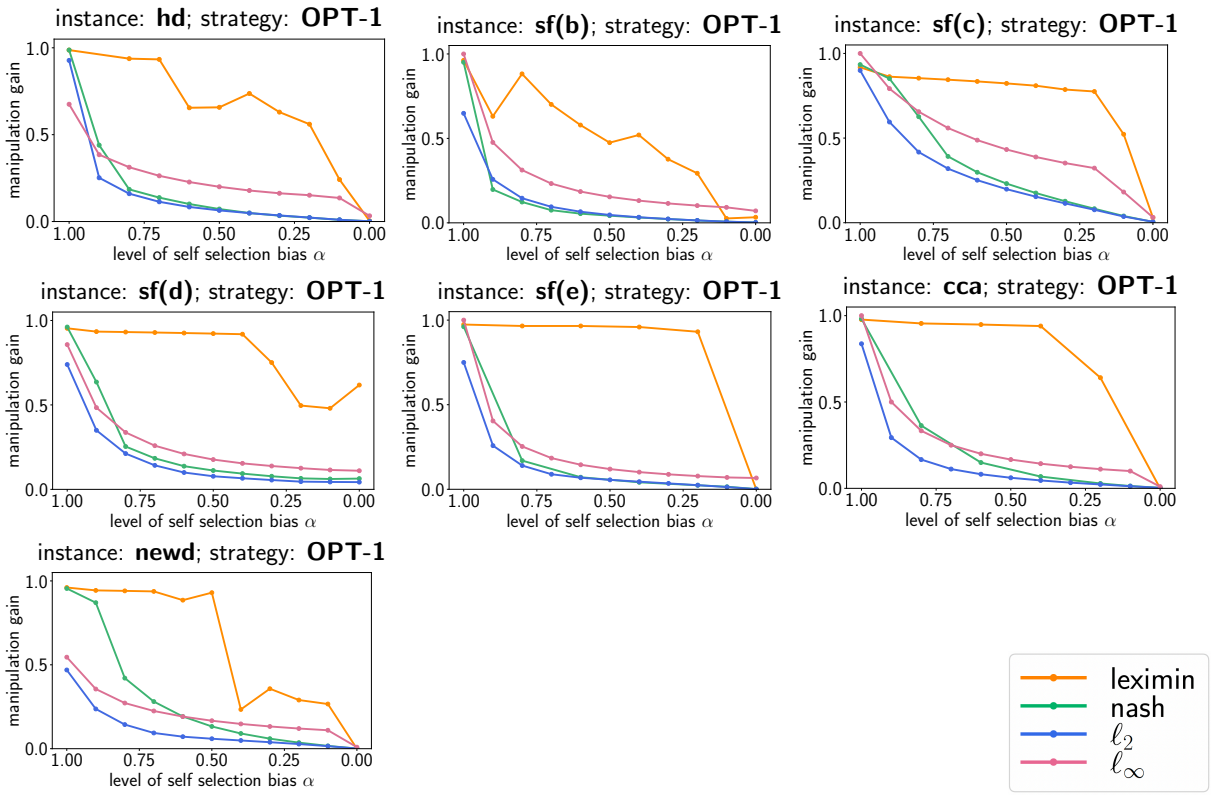


Figure 4: Figures for remaining instances from analysis in Figure 2(b)

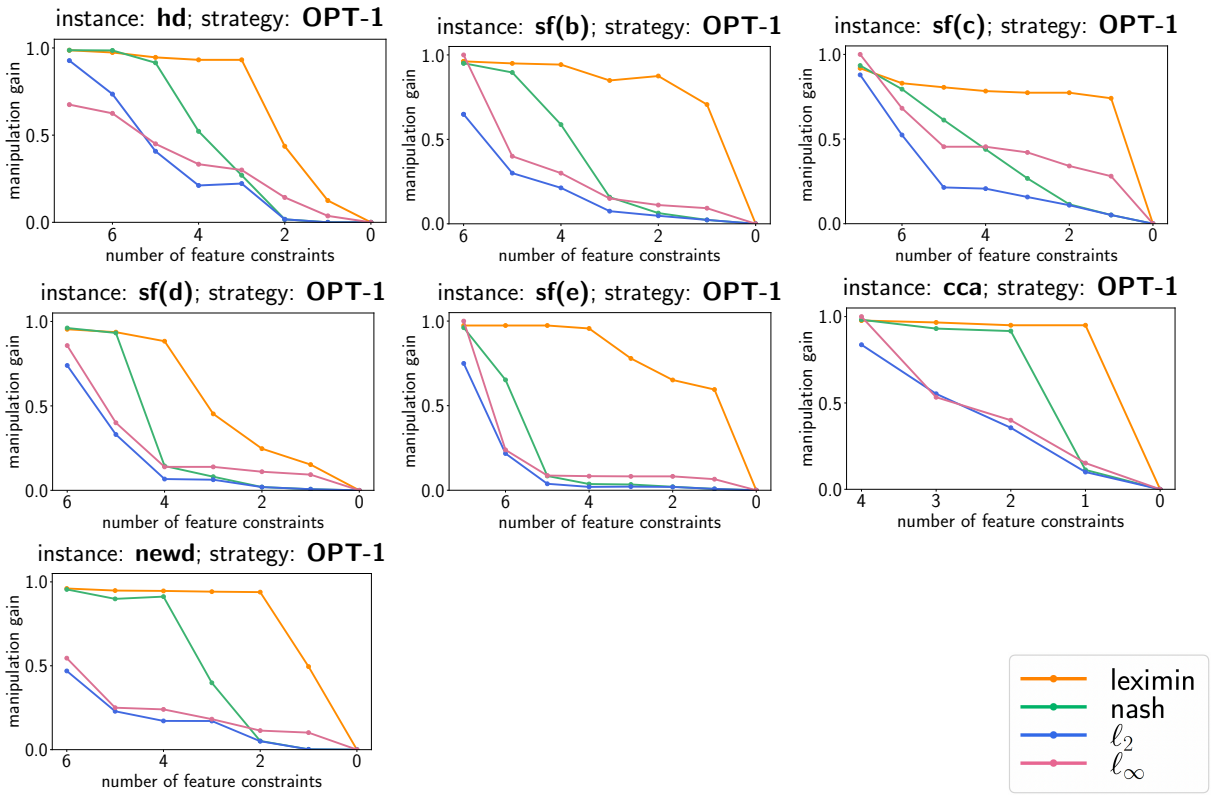


Figure 5: Figures for remaining instances from analysis in Figure 2(c)

D.7 Empirical minimum selection probabilities given by norms

Minimum probability	sf(a)	sf(b)	sf(c)	sf(d)	sf(e)	sf(hd)	sf(newd)	sf(cca)
ℓ_2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ℓ_∞	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 2: Minimum selection probability given to any agent by ℓ_2, ℓ_∞ across instances