# Migration as Submodular Optimization

**Paul Gölz** and **Ariel D. Procaccia**
Computer Science Department
Carnegie Mellon University

**Abstract**

Migration presents sweeping societal challenges that have recently attracted significant attention from the scientific community. One of the prominent approaches that have been suggested employs optimization and machine learning to match migrants to localities in a way that maximizes the expected number of migrants who find employment. However, it relies on a strong additivity assumption that, we argue, does not hold in practice, due to competition effects; we propose to enhance the data-driven approach by explicitly optimizing for these effects. Specifically, we cast our problem as the maximization of an approximately submodular function subject to matroid constraints, and prove that the worst-case guarantees given by the classic greedy algorithm extend to this setting. We then present three different models for competition effects, and show that they all give rise to submodular objectives. Finally, we demonstrate via simulations that our approach leads to significant gains across the board.

## 1 Introduction

Migration is one of the greatest societal challenges facing humanity in the 21st Century. In 2015 there were 244 million international migrants in the world, suggesting a larger increase in the rate of international migration than was previously anticipated (McAuliffe and Ruhs, 2018). A key reason behind this increase is the widely understood fact that migration often has significant benefits to the migrants and their families, as well as the host country and the country of origin. For example, migrants working in the United States earn wages that are higher by a median factor of 4.11 than those they would have earned in their home countries (Clemens, Montenegro, and Prichett, 2008); and the money they send back to their countries of origin is a reliable source of foreign currency. But the increase in the rate of migration is also driven by globalization, social disparities, and, in no small part, by a vast number of refugees — including, most recently, millions who have fled war in Syria, persecution in Myanmar, and economic calamity in Venezuela.

These events have fueled a surge of attempts to put migration, and, especially, refugee resettlement, on scientific footing. Alvin Roth, a Nobel laureate in economics, nicely summarizes the problem in a 2015 opinion piece (Roth, 2015):

> "Refugee relocation is what economists call a matching problem, in the sense that different refugees will thrive differently in different countries. Determining who should go where, and not just how many go to each country, should be a major goal of relocation policy."

This observation underlies work in market design, which draws on the extensive literature on matching problems such as school choice. It focuses on refugee resettlement mechanisms that elicit refugees' preferences over localities, and output a matching that is desirable with respect to these preferences (Moraga and Rapoport, 2014; Jones and Teytelboym, 2018; Delacrétaz, Kominers, and Teytelboym, 2016).

By contrast, a recent paper by Bansak et al. (2018), published in *Science*, takes a markedly different, data-driven approach to refugee resettlement, which employs machine learning and optimization. Their goal is to maximize the expected number of refugees who find employment.[1] Using historical data from the United States and Switzerland, they train predictors that estimate the probability $p_{i\ell}$ that a given refugee $i$ would find employment in a given locality $\ell$. The optimal solution is then an assignment of refugees to localities that maximizes the sum of probabilities, subject to capacity constraints. Assuming their predictions of employment probabilities are accurate, Bansak et al. demonstrate that their approach leads to a 40%–70% increase in the number of employed refugees, compared to the actual outcomes in the United States and Switzerland.

On a high level, we subscribe to the data-driven approach, and believe that the assumptions made by Bansak et al. (2018) are quite reasonable in the context of refugee resettlement — most notably, the implicit assumption that the probability $p_{i\ell}$ can be estimated based only on information about the refugee $i$ and the locality $\ell$, and does not depend on where other refugees are assigned. But as this approach gains traction, we envision it being deployed on a larger scale, and ultimately informing international migration policy more broadly, especially in the context of labor migration.

The key observation behind our work is that, at that scale, *competition effects* would invalidate the foregoing assumption. Indeed, the larger the number of, say, engineers, who settle in a specific area, the less likely it is that any particular

[1] To be precise, they consider families and maximize the expected number of families such that at least one member is employed, but this does not make a significant technical difference, so we focus on individuals for ease of exposition.

engineer would find a job. The immigration of approximately 331,000 Jews from the former Soviet Union to Israel in 1990 and 1991 serves as a case in point. A disproportionate number of these highly-educated immigrants were engineers and doctors, leading to a saturation of the local job markets: "a state with an oversupply of doctors could not possibly double the number of its doctors in several years" (Smooha, 2008). Our goal in this paper, therefore, is to enhance the data-driven approach to migration by explicitly modeling competition effects, and directly optimizing for them.

### 1.1 Our Approach and Results

On a technical level, our objective function $f$ receives a set of migrant-locality pairs as input, and returns the predicted number of employed migrants under the corresponding assignment. Crucially, we assume that $f$ is (monotone) *submodular*: individual elements provide diminishing marginal returns. Formally, if $S$ and $T$ are two subsets of migrant-locality pairs such that $S \subseteq T$, and $(i, \ell)$ is a migrant-locality pair such that $(i, \ell) \notin T$, then

$$f(S \cup \{(i, \ell)\}) - f(S) \geq f(T \cup \{(i, \ell)\}) - f(T).$$

This captures the idea that the larger the number of migrants that compete with $i$ for a job at locality $\ell$, the less likely it is that $i$ herself would find employment — especially if it is a skilled job — and the smaller the marginal contribution of $(i, \ell)$ to the objective function (overall number of migrants who find employment).

We can therefore cast our optimization problem as the maximization of a monotone submodular function subject to *matching constraints*, which represent caps on the number of migrants each locality can absorb.[2] In fact, we allow the objective function to be *approximately* submodular since most ways of estimating employment for a matching will introduce noise that invalidates submodularity.

The matching constraints in question can be naturally described as the intersection of two *matroids*. In Section 3, we show that a simple greedy algorithm — which is known to perform well when given access to an exactly submodular function — gives an approximation guarantee of $(P + 1 + \frac{4\epsilon}{1-\epsilon} k)^{-1}$ when maximizing a submodular function $f$ subject to $P$ many matroids if we only have access to an $\epsilon$-approximately submodular function approximating $f$ and if $k$ is the size of the largest set in the intersection of the matroids. We expect this result to be of independent use. In our setting, it gives us a guarantee — for $P = 2$ — on the performance of our greedy matching with respect to the optimal matching.

The submodular objective function can potentially be learned, or optimized directly, from historical data (Balcan and Harvey, 2018; Balkanski, Rubinstein, and Singer, 2016) without any further structural assumptions. As an alternative, in Section 4, we propose three models of how migrants find employment, all of which induce submodular objective functions. The purpose of these models is twofold: First, they represent different ways in which competition effects might arise, thereby helping to understand them. Second, in practice they may prove to be more accurate as they are defined by relatively few parameters, and are easier to train.

In Section 5, we compare the employment generated by the greedy heuristic on the submodular model to the baseline approach of assuming additivity. We find that the benefits of accounting for submodularity almost always outweigh the loss associated with using an inexact optimization technique. Across our three models and a variety of settings, the greedy approach frequently increases employment by 10% and more, suggesting that substantial gains can be made in practice.

### 1.2 Related Work

Our work is most closely related to that of Ahmed, Dickerson, and Fuge (2017). Motivated by *diversity* in problems such as movie recommendation and assignment of papers to reviewers, they formulate a weighted bipartite $b$-matching problem with a specific submodular objective, which is similar to our problem under a particular instantiation of the retroactive-correction model. They show that this problem can be formulated as a quadratic program (which may be intractable at a large scale). They also use the greedy algorithm, but only note in passing that it would give worst-case guarantees in a degenerate special case that essentially coincides with having no constraints. By contrast, our worst-case guarantees of Section 3 hold for any *approximately* submodular objective function *under capacity constraints*, or even under an arbitrary number of matroid constraints. Another key difference is that Ahmed, Dickerson, and Fuge focus on one specific objective function, whereas our results of Section 4 explore several different models, which are tailored to the migration domain. Perhaps the most striking difference is more fundamental: For Ahmed, Dickerson, and Fuge, diversity is an objective that is orthogonal to (additive) efficiency. By contrast, we consider diversity as an inherent part of efficiency since matchings lacking in diversity will suffer from competition effects.

A bit further afield, there is a large body of work on submodular optimization, and its applications in AI. The papers of Fisher, Nemhauser, and Wolsey (1978) and Horel and Singer (2016) are especially relevant — we discuss their results in Section 3. Applications of submodular optimization include influence in social networks (Kempe, Kleinberg, and Tardos, 2003), sensor placement (Krause et al., 2008), and human computation (Shahaf and Horvitz, 2010), just to name a few.

## 2 Preliminaries

Let $N$ be a set of agents or migrants, and $L$ be a set of localities. Each locality $\ell$ has a capacity of $q_\ell \in \mathbb{N}$, limiting the number of migrants it is willing to accept. A matching is a set $S \subseteq N \times L$ of migrant-locality pairs, such that every agent is matched to at most one locality and every locality $\ell$ to at most $q_\ell$ migrants. Our general aim is to find matchings that maximize a given function $f : 2^{N \times L} \to \mathbb{R}_{\geq 0}$, which assigns a utility to every matching.

In this paper, we will assume that $f$ is either submodular or approximately submodular. To be *submodular*, $f$ must satisfy

[2] Capacity constraints can be transformed into matching constraints on the complete bipartite graph with migrants on one side, and localities on the other, where the number of copies of each locality is equal to its capacity.

for all $S \subseteq T \subseteq N \times L$ and $x \notin T$ that $f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$. To prove submodularity, it is sufficient to show the above for $T$ being of the shape $S \cup \{y\}$ with $y \notin S$. A function $f$ is *supermodular* iff $(-f)$ is submodular, i.e., iff $f(S \cup \{x\}) - f(S) \leq f(T \cup \{x\}) - f(T)$ for appropriate $S, T$ and $x$. We will assume all submodular functions to be monotone and normalized, i.e., $f(\emptyset) = 0$.

A function $f$ is *$\epsilon$-approximately submodular* for some $\epsilon > 0$ if and only if there is a submodular function $\hat{f}$ such that $(1 - \epsilon)\,\hat{f}(S) \leq f(S) \leq (1 + \epsilon)\,\hat{f}(S)$. Note that $f$ itself need not be monotone.

A (finite) *matroid* is a pair $(G, \mathcal{F})$ of a finite ground set $G$ and a set $\mathcal{F} \subseteq 2^G$ of *independent sets* such that (i) $\emptyset \in \mathcal{F}$, (ii) if $S \in \mathcal{F}$, then every subset $T$ of $S$ is also in $\mathcal{F}$, and (iii) if $S, T \in \mathcal{F}$ such that $|S| = |T| + 1$, there is some $x \in S \setminus T$ such that $T \cup \{x\} \in \mathcal{F}$. For any set $S \subseteq G$, let its *rank* $r(S)$ denote the size of the largest independent subset of $S$, and let its *span* $sp(S)$ be the set of all elements $x \in G$ such that $r(S) = r(S \cup \{x\})$. An important class of matroids are the *partition matroids* that partition the ground set $G$ into disjoint subsets $G_1, \ldots, G_k$ and designate $S \subseteq G$ as independent iff $|S \cap G_i|$ is at most some cap $g_i$ for all $i$.

## 3 Algorithmic Results

Both the constraint of matching each agent to at most one locality and the constraint induced by locality quotas easily translate into partition matroids over $N \times L$.[3] In the first matroid, we restrict a matching to select at most one edge out of the set of edges incident to each agent $i$; in the second matroid, to at most $q_\ell$ edges out of the set of edges incident to each locality $\ell$. Then, the matchings are exactly the sets that are independent in both matroids. Therefore, maximizing employment among matchings is an instance of maximizing a submodular function subject to multiple matroid constraints.

To optimize for employment, we must choose a way of predicting it under a given matching. For instance, we might use techniques from machine learning to generalize past observations about employment success. Alternatively, we might adopt a hand-crafted model — such as the ones presented in Section 4 — and set only its parameters based on data. The latter approach ensures that our predictor behaves reasonably on all inputs and requires less data for fitting.

However we choose to predict employment, the corresponding employment function will have a fairly complicated shape. As a result, we face the obstacle that optimizing the function exactly would be impractical from a computational viewpoint. Fortunately, Fisher, Nemhauser, and Wolsey (1978) found that a simple greedy algorithm gives an approximation guarantee of $\frac{1}{P+1}$ when optimizing a monotone, submodular function $z$ subject to $P$ matroid constraints.[4] Furthermore, in practice, the same greedy algorithm has been found to perform much better than this theoretical guarantee, often giving results that are close to optimal (Kempe, Kleinberg, and Tardos, 2003).

Call the matroids $\mathcal{M}_p = (N, \mathcal{F}_p)$ for $1 \leq p \leq P$, let $sp^p$ denote the span with respect to $\mathcal{M}_p$, and let the intersection of the matroids be $\mathcal{F} \coloneqq \bigcap_{p=1}^{P} \mathcal{F}_p$. To refer to the marginal contribution of an element $j$ with respect to a set $S$, we set $\rho_j(S) \coloneqq z(S \cup \{j\}) - z(S)$.

The greedy algorithm initializes $S^0 \leftarrow \emptyset$, $N^0 \leftarrow N$ and $t \leftarrow 1$. Then, it proceeds through the following steps, wherein $t$ denotes the number of the current iteration:

**Step 0.** If $N^{t-1} = \emptyset$, stop with $S^{t-1}$ as the greedy solution.

**Step 1.** Select $i(t) \in N^{t-1}$ for which $z(S^{t-1} \cup \{i(t)\})$ is maximal, with ties settled arbitrarily.

**Step 2a.** If $S^{t-1} \cup \{i(t)\} \notin \mathcal{F}$, set $N^{t-1} \leftarrow N^{t-1} \setminus \{i(t)\}$ and return to step 0.

**Step 2b.** If $S^{t-1} \cup \{i(t)\} \in \mathcal{F}$, set $\rho_{t-1} \leftarrow \rho_{i(t)}(S^{t-1})$, $S^t \leftarrow S^{t-1} \cup \{i(t)\}$, and $N^t \leftarrow N^{t-1} \setminus \{i(t)\}$.

**Step 3.** Set $t \leftarrow t + 1$ and continue from step 0.

Although the greedy algorithm is an effective way of maximizing submodular functions, we are unlikely to have direct access to the submodular function we are interested in, even for the value queries required for the greedy algorithm. If we adopt a machine-learning approach, our predictor might get reasonably close to the real employment function but is unlikely to be exactly monotone and submodular. The same problem presents itself even for the hand-crafted models: While we prove that each model induces a perfectly submodular function, this function is defined as the expectation over a random process. For scenarios of nontrivial size, we can only estimate this function through repeated sampling, and the introduced noise may invalidate the monotonicity and submodularity of the function. In both cases, we can only expect our functions to be approximately submodular.

Fortunately, the greedy algorithm still performs well if $z$ is only approximately submodular. We adapt the classic result by Fisher, Nemhauser, and Wolsey (1978) to show the following approximation guarantee.

**Theorem 3.1.** *Let $z : 2^N \to \mathbb{R}$ be $\epsilon$-approximately submodular and let $\hat{z}$ be the underlying (monotone, normalized) submodular function, i.e., let*

$$(1 - \epsilon)\,\hat{z}(S) \leq z(S) \leq (1 + \epsilon)\,\hat{z}(S)$$

*for all $S \subseteq N$. Let $\mathcal{F}$ be the intersection of $P$ matroids, and let $k$ denote the size of the largest $S \in \mathcal{F}$. Then the greedy algorithm selects a set $S \in \mathcal{F}$ such that*

$$\left(P + 1 + \frac{4\,\epsilon}{1 - \epsilon}\,k\right)\,\hat{z}(S) \geq \max_{S' \in \mathcal{F}} \hat{z}(S').$$

To the best of our knowledge, we are the first to give approximation bounds for optimizing approximately submodular functions subject to multiple matroid constraints. For a single matroid, Horel and Singer (2016) give a slightly sharper bound of $(2 + \frac{2\,\epsilon}{1-\epsilon}\,k)\,\hat{z}(S) \geq \max_{S'} \hat{z}(S')$, but their proof makes use of strong properties that only hold if $P = 1$. By contrast, in order to capture our matching constraints we

[3] We can even deal with migrant-locality incompatibilities, by removing incompatible pairs from the ground set.

[4] Lee et al. (2010) report a polynomial-time algorithm that achieves an approximation ratio of $\frac{1}{P+\epsilon}$ for $P \geq 2$ partition matroids, where $\epsilon$ can be made arbitrarily small. However, the degree of the polynomial grows very fast as $\epsilon$ becomes smaller, making their algorithm impractical for our purposes.

need to take the intersection of two matroids, that is, for our application $P = 2$, and the corresponding worst-case approximation guarantee is roughly $1/3$ — assuming $\epsilon$ is sufficiently small. This is a realistic assumption, especially in our handcrafted models, as standard concentration inequalities imply that sampling leads to exponentially fast convergence to the true function value.

Turning to the proof, let $U^t$ be the set of elements considered in the first $t+1$ iterations except for the element added to $S$ in iteration $t+1$, i.e., $U^t := N \setminus N^t$. We make use of the following known results.

**Lemma 3.2** (Fisher, Nemhauser, and Wolsey 1978)**.** *For all $t$, $U^t \subseteq \bigcup_{p=1}^{P} sp^p(S^t)$.*

**Lemma 3.3** (Fisher, Nemhauser, and Wolsey 1978)**.** *Let $\rho_i, \sigma_i \geq 0$ be given for $i = 0, \ldots, K-1$ such that $\sum_{i=0}^{t-1} \sigma_i \leq t$ for $t = 1, \ldots, K$, and the sequence $(\rho_i)_i$ decreases monotonically. Then,*

$$\sum_{i=0}^{K-1} \rho_i \, \sigma_i \leq \sum_{i=0}^{K-1} \rho_i.$$

We are now ready for the theorem's proof.

*Proof of Theorem 3.1.* Let $\hat{T}$ be a maximizing subset for $\hat{z}$ and let $S$ be the set returned by the greedy mechanism when run on $z$. Let $K = |S|$ and define $R_{t-1} := \hat{T} \cap (U^t \setminus U^{t-1})$, $r_{t-1} = |R_{t-1}|$, for $t = 1, \ldots, K$. Without loss of generality, all elements of the ground set can actually appear in a set $S \in \mathcal{F}$, thus $U^0 = \emptyset$.

Let $\hat{\rho}_j(S)$ be the marginal contribution of $j$ with respect to $S$ and the function $\hat{z}$. If $\rho_{t-1}$ was set in the algorithm as $\rho_{i(t)}(S^{t-1})$, let $\hat{\rho}_{t-1}$ denote $\hat{\rho}_{i(t)}(S^{t-1})$.

By submodularity of $\hat{z}$, it holds that

$$\hat{z}(\hat{T}) \leq \hat{z}(S) + \sum_{j \in \hat{T}} \hat{\rho}_j(S). \tag{1}$$

We can bound the second term as

$$\begin{aligned}
&\sum_{j \in \hat{T}} \hat{\rho}_j(S) \\
&= \sum_{t=1}^{K} \sum_{j \in R_{t-1}} \hat{\rho}_j(S) \\
&\leq \sum_{t=1}^{K} \sum_{j \in R_{t-1}} \hat{\rho}_j(S^{t-1}) && \text{(submodularity, } S^{t-1} \subseteq S\text{)} \\
&\leq \sum_{t=1}^{K} \sum_{j \in R_{t-1}} \frac{z(S^{t-1} \cup \{j\})}{1-\epsilon} - \sum_{t=1}^{K} \sum_{j \in R_{t-1}} \hat{z}(S^{t-1}).
\end{aligned}$$

The greedy algorithm chose $i(t)$ as the element $j \in N^{t-1} = N \setminus U^{t-1}$ with maximum $z(S^{t-1} \cup \{j\})$, thus

$$\begin{aligned}
&\leq \frac{1}{1-\epsilon} \sum_{t=1}^{K} \sum_{j \in R_{t-1}} z(S^{t-1} \cup \{i(t)\}) - \sum_{t=1}^{K} \sum_{j \in R_{t-1}} \hat{z}(S^{t-1}) \\
&\leq \frac{1+\epsilon}{1-\epsilon} \sum_{t=1}^{K} r_{t-1} \, \hat{z}(S^t) - \sum_{t=1}^{K} r_{t-1} \, \hat{z}(S^{t-1}) \\
&= \sum_{t=1}^{K} r_{t-1} \, \hat{\rho}_{t-1} + \frac{2\,\epsilon}{1-\epsilon} \sum_{t=1}^{K} r_{t-1} \, \hat{z}(S^t).
\end{aligned} \tag{2}$$

We claim that for all $t = 1, \ldots, K$, $\sum_{i=1}^{t} \frac{r_{i-1}}{P} \leq t$. Indeed,

$$\sum_{i=1}^{t} r_{i-1} = |\hat{T} \cap U^t|.$$

Since, by Lemma 3.2, $U^t \subseteq \bigcup_{p=1}^{P} sp^p(S^t)$, it follows that

$$\sum_{i=1}^{t} r_{i-1} \leq \sum_{p=1}^{P} |\hat{T} \cap sp^p(S^t)|.$$

Because $\hat{T}$ is independent in the matroid $(N, \mathcal{F}_p)$, and because the rank of $sp^p(S^t)$ is $t$, $|\hat{T} \cap sp^p(S^t)| \leq t$. Thus, $\sum_{i=1}^{t} r_{i-1} \leq P\,t$, and the claim follows.

From the sequence $(\hat{\rho}_{t-1})_t$, form a new sequence $(\bar{\rho}_{t-1})_t$, where

$$\bar{\rho}_{t-1} := \min_{1 \leq t' \leq t} \hat{\rho}_{t'-1}.$$

Clearly, this sequence decreases monotonically and is nonnegative. Fix some $t$ and let $t' = \operatorname{argmin}_{1 \leq t' \leq t} \hat{\rho}_{t'-1}$. Then, we can relate $\hat{\rho}_{t-1}$ and $\bar{\rho}_{t-1}$ as follows:

$$\begin{aligned}
\hat{\rho}_{t-1} &= \hat{\rho}_{i(t)}(S^{t-1}) \\
&\leq \hat{\rho}_{i(t)}(S^{t'-1}) && \text{(submodularity of } \hat{z}, t' \leq t\text{)} \\
&= \hat{z}(S^{t'-1} \cup \{i(t)\}) - \hat{z}(S^{t'-1}) \\
&\leq \frac{z(S^{t'-1} \cup \{i(t)\})}{1-\epsilon} - \hat{z}(S^{t'-1}) \\
&\leq \frac{z(S^{t'-1} \cup \{i(t')\})}{1-\epsilon} - \hat{z}(S^{t'-1}) \\
&&& \text{(greedy chose } i(t') \text{ over } i(t) \text{ in iteration } t'\text{)} \\
&\leq \frac{1+\epsilon}{1-\epsilon} \, \hat{z}(S^{t'-1} \cup \{i(t')\}) - \hat{z}(S^{t'-1}) \\
&= \hat{\rho}_{t'-1} + \frac{2\,\epsilon}{1-\epsilon} \, \hat{z}(S^{t'}) \\
&\leq \hat{\rho}_{t'-1} + \frac{2\,\epsilon}{1-\epsilon} \, \hat{z}(S^{t}) && \text{(monotonicity)} \\
&= \bar{\rho}_{t-1} + \frac{2\,\epsilon}{1-\epsilon} \, \hat{z}(S^{t}) && \text{(def. of } \bar{\rho}_{t-1}\text{, choice of } t'\text{)}.
\end{aligned}$$

By Lemma 3.3, we know that

$$\sum_{t=1}^{K} r_{t-1} \, \bar{\rho}_{t-1} = P \sum_{t=1}^{K} \frac{r_{t-1}}{P} \, \bar{\rho}_{t-1} \leq P \sum_{t=1}^{K} \bar{\rho}_{t-1}. \tag{3}$$

This allows us to continue Eq. (2):

$$\begin{aligned}
&\sum_{t=1}^{K} r_{t-1} \, \hat{\rho}_{t-1} + \frac{2\,\epsilon}{1-\epsilon} \sum_{t=1}^{K} r_{t-1} \, \hat{z}(S^t) \\
&\leq \sum_{t=1}^{K} r_{t-1} \left( \bar{\rho}_{t-1} + \frac{2\,\epsilon}{1-\epsilon} \, \hat{z}(S^t) \right) + \frac{2\,\epsilon}{1-\epsilon} \sum_{t=1}^{K} r_{t-1} \, \hat{z}(S^t) \\
&\leq P \sum_{t=1}^{K} \bar{\rho}_{t-1} + \frac{4\,\epsilon}{1-\epsilon} \sum_{t=1}^{K} r_{t-1} \, \hat{z}(S^t) && \text{(Eq. 3)}
\end{aligned}$$

$$\leq P \sum_{t=1}^{K} \hat{\rho}_{t-1} + \frac{4\,\epsilon}{1-\epsilon} \sum_{t=1}^{K} r_{t-1}\, \hat{z}(S^t) \qquad \text{(def. of } \bar{\rho}_{t-1})$$

$$\leq P\, \hat{z}(S) + \frac{4\,\epsilon}{1-\epsilon} |\hat{T}|\, \hat{z}(S)$$

$$\leq P\, \hat{z}(S) + \frac{4\,\epsilon}{1-\epsilon} k\, \hat{z}(S).$$

Combining this with Eq. (1) yields

$$\hat{z}(\hat{T}) \leq \left( P + 1 + \frac{4\,\epsilon}{1-\epsilon} k \right) \hat{z}(S). \qquad \square$$

# 4 Submodular Objectives

We propose three models for the submodular effects of competition between migrants. Each of them makes different assumptions about how migrants find their jobs and, thus, about how they compete with each other. In each model we are interested in the *expected employment function* that it induces, i.e., the function that takes as input agent-locality pairs, and outputs the expected number of agents that find employment (in each case this function depends on parameters of the model).

## 4.1 The Retroactive-Correction Model

The easiest of our models generalizes the additive model used by Bansak et al. (2018), by retroactively correcting employment success for competition. We assume that the agents $N$ can be partitioned into disjoint professions $\pi$, and that only agents of the same profession compete for jobs. Each agent $i$ has a probability $p_{i\ell}$ of finding employment in locality $\ell$.

In the additive model, one can imagine each migrant's job search as a coin with a bias of this probability; the employment is the number of successful coin flips, and, therefore, the expected employment is just the sum of probabilities $p_{i\ell}$ over all matched pairs of agents $i$ and localities $\ell$. In this model, however, the coin flip only simulates an agent's attempt to qualify for being hired, not her chance of actually landing a job, since the latter is influenced by other agents. An agent might qualify by means of language acquisition, by transferring degrees, through further professional training, or by choosing a good way to present herself to prospective employers. The actual employment at a locality $\ell$ and in a profession $\pi$ is obtained by applying a concave and monotone *correction function* $C_{\ell\pi} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ to the number of qualifying agents. Total employment is computed by summing up employment over all localities and professions; the submodular function to optimize for is the expected employment generated by this process. While there is no direct competition between agents of different professions, a matching algorithm cannot simply treat them in isolation since different professions have to share the locality caps.

An easy example for a correction function might be $x \mapsto \min(x, c)$ for some constant $c$. This models the scenario where there are $c$ jobs in that locality and profession, and where the job market manages to bring all qualifying agents into work, up to the hard cap of $c$. Other functions might slow down their growth before hitting the cap, simulating inefficiencies in the job market.

**Observation 4.1.** *The expected employment function induced by the retroactive-correction model is submodular.*

To see this, it is enough to show submodularity for a single locality $\ell$ and profession $\pi$ since a sum of submodular functions is submodular. Because all agent-locality pairs have $\ell$ as the second component, we can identify them with their agents. For a set $S$ of agents and two other agents $x$ and $y$, we need to show that the marginal contribution of $x$ is at least as high with respect to $S$ as it is with respect to $S \cup \{y\}$. Fix the coin flips for all agents. If $x$ failed, her marginal contribution is zero in both cases. If $y$ failed, the marginal contribution of $x$ is the same whether $y$ is present or not. If both succeed, the marginal contribution must be at least as high with respect to the smaller set than with respect to the larger one by the concavity of $C_{\ell\pi}$. Since employment in the case without fixed coin flips can be seen as a convex combination of employment in the cases with fixed coin tosses, submodularity is preserved.

## 4.2 The Interview Model

In the second model, we again partition agents by profession. In contrast to the previous model, a migrant's employment success is not determined by a single hurdle of qualification but through a sequence of applications to individual jobs. Each locality $\ell$ has a certain number $k_{\ell\pi}$ of jobs for profession $\pi$, and each agent $i \in N$ has a probability $p_{i\ell}$ of being accepted at a particular job of their profession at locality $\ell$.

Employment is calculated individually for each locality $\ell$ and profession $\pi$. Initially, there are $k_{\ell\pi}$ many jobs available. Then, we iterate over agents of profession $\pi$ matched with locality $\ell$ in an order selected uniformly at random. For agent $i$, we throw $p_{i\ell}$-biased coins until we either hit success or until we have failed as many times as the number of available jobs. Each of these coin tosses represents the agent applying for one of the remaining jobs, where each application succeeds with probability $p_{i\ell}$. If one of the tosses succeeds, this agent is counted as employed, and the process continues with one less job available. Else, the agent is not employed, and the number of available jobs remains the same for the next agent. The utility of a matching is the sum over localities and professions of the expected number of employed agents in that locality and profession.

**Theorem 4.2.** *The expected employment function induced by the interview model is submodular.*

In contrast to our other models, it is quite nontrivial to establish submodularity in the interview model. The proof of Theorem 4.2 is relegated to Appendix A.

## 4.3 The Coordination Model

Our final model goes beyond the strict separation between professions, and assumes that employment for all agents in the same locality is coordinated. Similarly to the previous model, each locality $\ell$ has a number $k_\ell$ of jobs, and each agent $i$ has a certain probability $p_{ij}$ of being compatible with a specific job $j$. These probabilities might be induced by a strict partition into professions, but can be more fluid with people being more or less qualified for jobs closer to or further from their areas of expertise. Whereas, in the interview

model, a migrant directly takes a job as soon as she is found eligible, we now determine compatibility between all migrants and jobs in $\ell$, and we coordinate the assignment such that employment is maximized.

To calculate employment at a locality $\ell$, we flip a coin with bias $p_{ij}$ for each agent $i$ matched to $\ell$ and each job $j$ at $\ell$. By drawing an edge between each pair $(i, j)$ whose coin flip succeeded, we obtain a bipartite graph. Employment at this locality is the size of a maximum matching, which corresponds to the highest number of agents that can be given work. Again, the total utility of a matching is the sum of expected employment over localities.

**Theorem 4.3.** *The expected employment function induced by the coordination model is submodular.*

The relatively simple proof of the theorem builds up on a result by Rabanca (2016) and appears in Appendix B.

## 5 Simulations

The approximation results of Section 3 provide a way of respecting competition effects when matching migrants to localities. But when is it worth adopting such an approach? After all, besides the potential benefits, embracing submodularity also entails an increase in modeling complexity, and it forces us to abandon the efficient optimization tools that are applicable to additive optimization.

If one chooses not to deal with these drawbacks, one can always just ignore submodularity: For each agent and locality, one may estimate the probability of her finding employment at this place in the absence of all other agents. One can then optimize the additive function induced by the aforementioned probabilities, say, by formulating the problem as an integer linear program, and hope that the result would do well under the true objective function. The approach of Bansak et al. (2018) can be understood as doing exactly this. Our goal is to show that — when competition effects are indeed present, and the objective function is submodular — accounting for submodularity is almost always the preferable choice and can lead to significantly better outcomes than the additive approach described above. To this end, we empirically evaluate the two approaches on all three models from Section 4.

Our simulation code is written in Python; we use Gurobi for additive optimization and IGraph for computing maximum bipartite matchings. All code is open source and available for reproduction at `https://github.com/pgoelz/migration`. We provide additional simulation results in Appendix C, where we also link to IPython notebooks with the exact setup of all of our experiments.

For increased performance, we reuse estimations of expected employment. When a model is queried with a matching that puts the same agents of profession $\pi$ — or just the same agents in case of the coordination model — in the same locality $\ell$ as a previous matching, the previous estimation of expected employment is used. Since all these estimations are obtained by averaging many random simulations, this should not significantly influence the experiments. Furthermore, since the additive algorithm is not affected, and since all final utility values are computed without memoization, this only disadvantages the greedy algorithm, strengthening our findings.

Due to the high number of parameters in all of our models, we adopt a standard setting that is applicable across all of them. Let $N$ be a set of 100 agents, split equally between two professions, 1 and 2. While we vary the number of localities, we distribute 100 jobs — 50 per profession — randomly over the localities, ensuring that each locality has at least one job. Furthermore, we set the capacity of each locality to exactly its number of jobs. Finally, we average 1 000 simulations to estimate expected employment whenever our models are queried with a matching.

We quickly sketch how this setting translates into the idiosyncrasies of our individual models: In the retroactive-correction model, for each agent $i$, we uniformly select a probability between 0 and 1, which is used as $p_{i\ell}$ for all localities $\ell$. We choose to keep these probabilities equal between localities to make sure that our samples will contain agents that have significantly different overall chances of being employed, an aspect that we would expect to see in any real dataset. As the correction function for a locality $\ell$ and profession $\pi$, we choose $x \mapsto \min(x, c)$, where $c$ is the number of available jobs in $\ell$ and $\pi$. Note that, while the cap ensures that the number of agents in a locality is at most the total number of jobs, the number of agents of a certain profession can exceed the number of jobs in that profession, which lets the cap $c$ kick in. Likewise, in the interview model, we choose $p_{i\ell}$ uniformly at random and equal over all localities. Finally, we induce the compatibility probabilities in the coordination model by the professions, i.e., each agent has a single compatibility probability chosen uniformly at random for all jobs of her profession and is incompatible with all other jobs.

As shown in Fig. 1, the greedy algorithm outperforms additive optimization in nearly all cases, frequently increasing employment by 10% and more. Given the wide range of scenarios that are randomly generated, it is striking that the greedy heuristic virtually never leads to worse employment than additive maximization. This trend persists over all settings that we simulated. While the advantage of the greedy algorithm does not seem as pronounced for small numbers of localities in the retroactive-correction model, it leads to strong improvements over all locality numbers in the other two models.

In the following, we set the number of localities to 10 and vary the number of agents $n$ — still equally split between professions — instead. Each locality has a capacity of $n/10$ and has $n/10$ jobs, where the $n/2$ jobs of each profession are randomly distributed within these constraints. Figure 2 shows that the greedy algorithm leads to impressive improvements across all simulated numbers of agents, again especially pronounced in the interview and coordination models. While the gains become smaller for high numbers of agents in the correction model, the reason is innocuous: Looking at the absolute utilities instead of the ratio, we see that in these scenarios both additive and greedy get very close to an employment of half of the agents, which is the expected number of qualifying agents and thus optimal (see Appendix C.2).

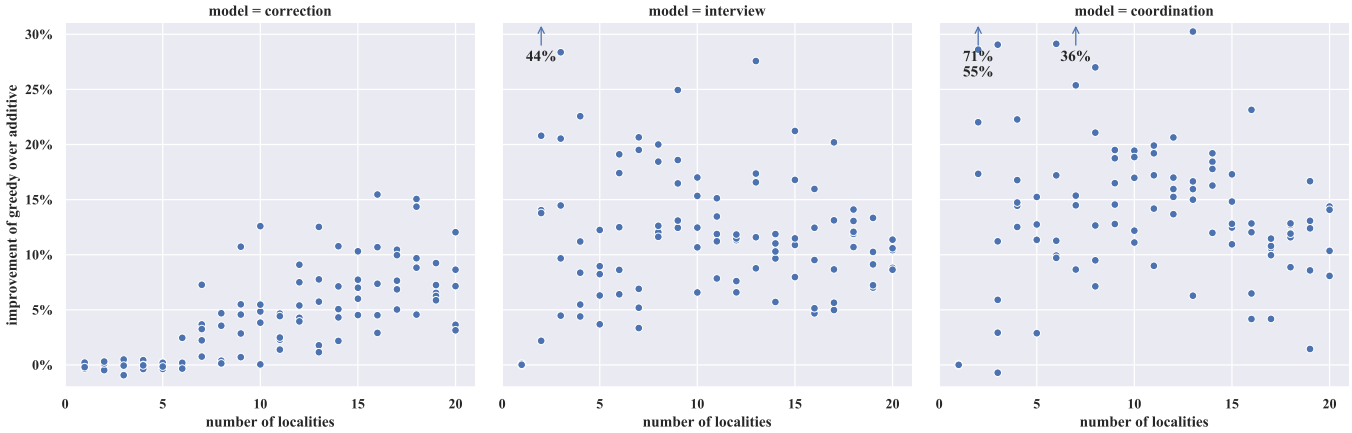



Figure 1: Increase in utility by using greedy instead of additive by number of localities. Each dot represents a new instantiation of the standard setting for the corresponding model, to which we applied both algorithms. Outliers are indicated by arrows.

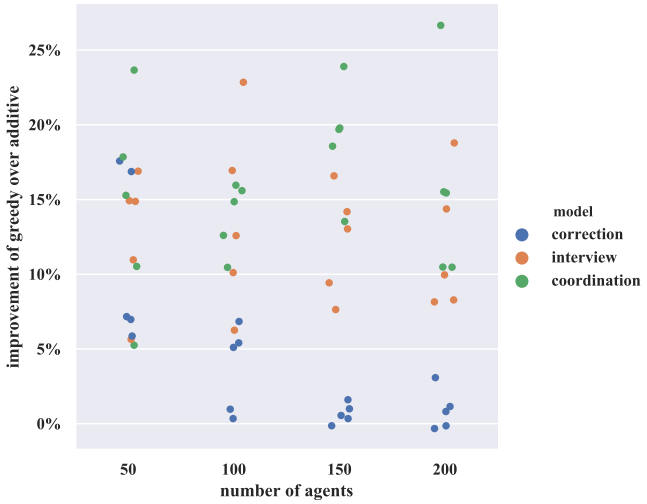



Figure 2: Improvements for different numbers of agents split across 10 localities.

In Appendices C.3 to C.5, we vary further parameters, namely the number of professions, the level of specialization of localities, and the availability of jobs with respect to fixed capacities. In general, we find that the improvements from the greedy optimization persist across these parameter settings.

Note that the results of the greedy algorithm provide a *lower bound* on what can be achieved when submodularity is taken into account. Other heuristics with access to the submodular function might be able to further improve employment while retaining the low query complexity of the greedy algorithm.

## 6 Discussion

The simulations in the previous section revealed that employment in all of our models can be improved by accounting for submodularity. Since this effect appeared consistently between three significantly different models of how migrants find employment, we conjecture that similar gains can be obtained in practice. The next logical step is to measure competition effects on real migration data.

Alas, it seems to be very difficult to obtain datasets with the necessary amount of detail. For instance, the *Database on Immigrants in OECD Countries* (DIOC)[5] contains data on a vast number of migrants in different countries. However, while File D contains information on the level of educational attainment and on the migrants' current occupations, there is no information about the field of education or pre-migration occupation. As a result, it is unclear whether an unemployed migrant was looking for employment at all and, if so, which job market she participated in. An additional problem is that DIOC does not track migrants who return to their home countries as a result of unemployment.

By contrast, longitudinal studies such as the German *Socio-Economic Panel*[6] contain extensive information about the educational background of individuals. What is more, they follow individual migrants over an extended period of time and therefore track their success on the labor market in perfect detail. Unfortunately, the longitudinal studies that we saw observe too few individuals spread over too many localities. To directly observe competition, we need data on a large fraction of the migrants competing for jobs in a locality, from multiple localities.

Even the paper by Bansak et al. (2018), for which the authors had access to sensitive migration data, conspicuously does not use any features relating to the refugees' field of occupation. Indeed, despite the predictive power of such features for employment success in a locality, at least one major refugee-resettlement agency in the US does not track this information at all. We have reason to believe that the same is true for other resettlement agencies as well. A clear policy recommendation, therefore, is to compile the records in question and to make them available for research. Hopefully, data on present migration will help to better direct flows of

[5] https://www.oecd.org/els/mig/dioc.htm

[6] https://www.diw.de/en/diw_02.c.222517.en/data.html

migration in the future — benefitting both the migrants and the societies they join.

## Acknowledgements

This work was partially supported by the National Science Foundation under grants IIS-1350598, IIS-1714140, CCF-1525932, and CCF-1733556; by the Office of Naval Research under grants N00014-16-1-3075 and N00014-17-1-2428; and by a Sloan Research Fellowship and a Guggenheim Fellowship. We would like to thank Lucas Leopold for his helpful pointers on navigating migration datasets.

## References

Ahmed, F.; Dickerson, J. P.; and Fuge, M. 2017. Diverse weighted biparite $b$-matching. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 35–41.

Balcan, M.-F., and Harvey, N. J. 2018. Submodular functions: Learnability, structure, and optimization. *SIAM Journal on Computing* 47(3):703–754.

Balkanski, E.; Rubinstein, A.; and Singer, Y. 2016. The power of optimization from samples. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS)*, 4017–4025.

Bansak, K.; Ferwerda, J.; Hainmueller, J.; Dillon, A.; Hangartner, D.; Lawrence, D.; and Weinstein, J. 2018. Improving refugee integration through data-driven algorithmic assignment. *Science* 359(6373):325–329.

Clemens, M. A.; Montenegro, C. E.; and Prichett, L. 2008. The place premium: Wage differences for identical workers across the US border. Policy research working paper no. 4671, World Bank.

Delacrétaz, D.; Kominers, S. D.; and Teytelboym, A. 2016. Refugee resettlement. Manuscript.

Fisher, M. L.; Nemhauser, G. L.; and Wolsey, L. A. 1978. An analysis of approximations for maximizing submodular set functions – II. *Mathematical Programming Study* 8:73–87.

Horel, T., and Singer, Y. 2016. Maximization of approximately submodular functions. *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS)* 3045–3053.

Jones, W., and Teytelboym, A. 2018. The local refugee match: Aligning refugees' preferences with the capacities and priorities of localities. *Journal of Refugee Studies* 31(2):152–178.

Kempe, D.; Kleinberg, J. M.; and Tardos, E. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining (KDD)*, 137–146.

Krause, A.; Leskovec, J.; Guestrin, C.; VanBriesen, J.; and Faloutsos, C. 2008. Efficient sensor placement optimization for securing large water distribution networks. *Journal of Water Resources Planning and Management* 134(6):516–526.

Lee, J.; Mirrokni, V. S.; Nagarajan, V.; and Sviridenko, M. 2010. Maximizing nonmonotone submodular functions under matroid and knapsack constraints. *SIAM Journal on Discrete Mathematics* 23(4):2053–2078.

McAuliffe, M., and Ruhs, M., eds. 2018. *World Migration Report*. IOM and United Nations.

Moraga, J. F.-H., and Rapoport, J. 2014. Tradable immigration quotas. *Journal of Public Economics* 115:94–108.

Rabanca, G. O. 2016. Maximum weight matching and submodular functions. Theoretical Computer Science Stack Exchange. URL: `https://cstheory.stackexchange.com/q/36209` (version: 2016-07-19).

Roth, A. E. 2015. Migrants aren't widgets. Politico Opinion.

Shahaf, D., and Horvitz, E. 2010. Generalized task markets for human and machine computation. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, 986–993.

Smooha, S. 2008. The mass immigrations to Israel: A comparison of the failure of the Mizrahi immigrants of the 1950s with the success of the Russian immigrants of the 1990s. *The Journal of Israeli History* 27(1):1–27.

# Appendix: Migration as Submodular Optimization

## A Proof of Theorem 4.2

Instead of directly proving the expected employment to be submodular, it is more natural to show that the expected number of positions that remain open is supermodular. Since a sum of supermodular functions is supermodular, it suffices to show this for a single locality $\ell$ and profession $\pi$. Since all agent-locality pairs will have the same locality, we can treat the set of pairs as a set of agents. Furthermore, we will show supermodularity for an arbitrary fixed order over all agents, which implies supermodularity for a random order.

For a set $S$ of agents of profession $\pi$, let $o(S)$ denote how many of the $k_{\ell\pi}$ jobs remain open in expectation if $S$ is the set of agents of profession $\pi$ matched with $\ell$, considered in the order fixed above. We need to show that for all sets $S$ of agents and distinct agents $x, y \notin S$,

$$o(S \cup \{x, y\}) - o(S \cup \{y\}) \geq o(S \cup \{x\}) - o(S). \tag{4}$$

How the process behaves at an agent $i$ is entirely determined by the number $k_i$ of jobs still available when it is her turn and the index $c_i$ of her first success in a sequence of coin flips. More specifically, it only matters whether $c_i \leq k_i$ (she gets a job) or $c_i > k_i$ (she fails to get any job). Without loss of generality, all $c_i$ are at most $k_{\ell\pi}$ since larger values of $c_i$ will lead to the same behavior of never getting a job. We show that Eq. (4) holds if the expectations are conditioned on fixed values of $c_i$ for all agents $i$ up to $x$ or $y$, whoever comes later in the order. We will refer to all agents up to the later of $x$ and $y$ as the *prefix*, and to all later agents as the *suffix*. Equation (4) follows from the previous proposition because the unconditional expectation is a convex combination of the expectations conditioned on these mutually exclusive and jointly exhaustive events. Fixing the coin flips of the prefix allows us to reason about how many jobs will still be available at the first agent of the suffix, and how this differs depending on whether $x$ and $y$ are present. Equation (4) will then follow from the fact that the expected number of remaining jobs is monotone and convex in the number of jobs available at the first agent of the suffix.

### A.1 Monotonicity and Convexity for a Fixed Set of Agents

Let $A$ be a set of agents $i$, each with their chance $p_{i\ell}$ of success when applying for a job. Set $\tilde{o}(n)$ for the number of positions remaining open after starting from $n$ available jobs and having all agents in $A$ apply in a fixed order.

**Lemma A.1.** *$\tilde{o}$ is monotone and is convex (i.e., $\tilde{o}(n+2) - \tilde{o}(n+1) \geq \tilde{o}(n+1) - \tilde{o}(n)$ for all $n \in \mathbb{N}$).*

*Proof.* One way of visualizing the random process with expectation $\tilde{o}(n)$ is the following: Have a grid with $n$ columns and one row for each agent in $A$, in their fixed order. The row corresponding to $i$ is filled with $p_{i\ell}$-biased coins. Proceed through the cells row by row, left to right. If the current row contains a flipped coin that was a success anywhere to the left of the current cell, remove the current coin from the grid and proceed. This corresponds to the case where agent $i$ already landed a job and does not apply for further jobs. Likewise, if the current column contains a successfully-flipped coin somewhere above the current cell, remove the current coin and proceed. This represents a job being already taken by a previous agent, which is why the current agent cannot apply there. Otherwise, flip the coin and proceed, simulating the current agent applying for the current job.

Clearly, the total number of successful coin flips after proceeding through all cells can be interpreted as the resulting employment number. For this reason, $\tilde{o}(n)$ equals the expected number of columns without a successful coin flip. Note that the probabilities of different end states of the grid do not change if we proceed through the grid in a different order, as long as all cells to the left of and above a cell have been evaluated before dealing with the current cell. In particular, we may proceed column by column, top to bottom.

In this interpretation, it is obvious that $\tilde{o}$ is monotone. Indeed, increasing its argument from $n$ to $n+1$ corresponds to adding a new column that will be evaluated after all others. Adding this column can increase the number of columns without successes but cannot decrease it.

Similarly, we can now show that $\tilde{o}$ is convex. By linearity of expectation, the two terms $\tilde{o}(n+2) - \tilde{o}(n+1)$ and $\tilde{o}(n+1) - \tilde{o}(n)$ describe the probability that column $n+2$ and $n+1$ stays without a success, respectively. Fix some coin flips for the first $n$ columns. To stay without success, column $n+1$ must fail the coin flips in all rows that do not have a success in the first $n$ columns yet. In the worst case, if column $n+1$ stays without success, column $n+2$ has to satisfy the same condition. But if column $n+1$ has a success, $n+2$ needs to fail on one less coin to stay without success. This shows that, conditional on the coin flips of the first $n$ columns, column $n+2$ is at least as probable to stay without success as column $n+1$. The unconditional probabilities are obtained via a convex combination of the conditional probabilities, so the inequality persists. □

### A.2 Description of the Process with Fixed Coin Flips

We now formalize the interviewing process for a fixed order and fixed coin flips in some prefix of agents. A *setting* consists of a sequence of integers $c_i$ for agents $i$ and a function $\tilde{f} : \mathbb{N} \to \mathbb{R}$. As described earlier, $c_i$ is the index of the first successful coin flip for an agent $i$ in the prefix, capped at a large enough constant. The function $\tilde{f}$ encapsulates all agents in the suffix, whose coin

flips are not fixed, by giving the expected number of left-over jobs as a function of the number of jobs that are available when entering the suffix. Such a setting can be complemented with a number $n$ of initially available jobs to form a *program*. Formally, settings $s$ and programs $p$ are defined by the following Backus-Naur forms:

$$s ::= [\tilde{f}] \mid m; s$$
$$p ::= s\ \{n\} \mid r$$

In the above, $n \in \mathbb{N}$ is used for the number of available jobs, $m \in \mathbb{N}_{>0}$ is used for the $c_i$, $r \in \mathbb{R}$ is the result of a function call, and $\tilde{f}$ is an externally defined function symbol $\tilde{f} : \mathbb{N} \to \mathbb{R}$ representing the suffix.

Programs can step according to the rules of our random process: If the prefix is empty, we just evaluate the function $\tilde{f}$. Otherwise, agent $i$ consumes a job iff $c_i$ is at most the number of available jobs. We capture this by a stepping relation $\mapsto$ between programs $p$, which is the smallest relation such that

- if $r = \tilde{f}(n)$, then $[\tilde{f}]\ \{n\} \mapsto r$,
- if $n \geq m$, then $m; s\ \{n\} \mapsto s\ \{n-1\}$, and
- if $n < m$, then $m; s\ \{n\} \mapsto s\ \{n\}$.

If we evaluate a program long enough, we reach a real number $r$, which is the expected number of remaining jobs after running the scenario on the number of available jobs. To be able to directly speak about the final evaluation result of a program, we define the evaluation relation $\Downarrow$ between programs and real numbers as the smallest one satisfying

- $r \Downarrow r$, and
- if $p \mapsto p'$ and $p' \Downarrow r$, then $p \Downarrow r$.

### A.3 Lemmas on Evaluation with Fixed Coin Flips

**Lemma A.2.** *If we have $m_1; \ldots; m_k; s\ \{n\} \mapsto^k p$ and $m_1; \ldots; m_k; s'\ \{n\} \mapsto^k p'$, then $p = s\ \{n'\}$ and $p' = s'\ \{n'\}$ for a single value $n' \in \mathbb{N}$.*

*Proof.* By induction on $k \in \mathbb{N}$. If $k = 0$, the claim follows immediately for $n' = n$. Else, let $k > 0$. Then, if $n \geq m_1$, we can determine the first step of the two terms as follows:

$$m_1; \ldots; m_k; s\ \{n\} \mapsto m_2; \ldots; m_k; s\ \{n-1\} \mapsto^{k-1} p$$
$$m_1; \ldots; m_k; s'\ \{n\} \mapsto m_2; \ldots; m_k; s'\ \{n-1\} \mapsto^{k-1} p'.$$

The claim about $p$ follows from the induction hypothesis.

Else, if $n < m_1$, we enter the analogous case of

$$m_1; \ldots; m_k; s\ \{n\} \mapsto m_2; \ldots; m_k; s\ \{n\} \mapsto^{k-1} p$$
$$m_1; \ldots; m_k; s'\ \{n\} \mapsto m_2; \ldots; m_k; s'\ \{n\} \mapsto^{k-1} p',$$

and the claim again follows from the induction hypothesis. □

**Lemma A.3.** *If we have $m_1; \ldots; m_k; s\ \{n\} \mapsto^k p$ and $m_1; \ldots; m_k; s'\ \{n+1\} \mapsto^k p'$, then $p = s\ \{n'\}$ and $p' = s'\ \{n''\}$ for $n'$ and $n''$ such that $n' \leq n'' \leq n' + 1$.*

*Proof.* By induction on $k \in \mathbb{N}$. If $k = 0$, the claim is trivial with $n' = n$ and $n'' = n + 1$. Else, we distinguish three cases, depending on whether $m_1 \leq n$, $m_1 = n + 1$, or $m_1 > n + 1$.

If $m_1 \leq n$, we can uniquely determine the first step of both terms as

$$m_1; \ldots; m_k; s\ \{n\} \mapsto m_2; \ldots; m_k; s\ \{n-1\} \mapsto^{k-1} p$$
$$m_1; \ldots; m_k; s'\ \{n+1\} \mapsto m_2; \ldots; m_k; s'\ \{n\} \mapsto^{k-1} p',$$

and the claim follows from the induction hypothesis.

If $m_1 = n + 1$, we know that

$$m_1; \ldots; m_k; s\ \{n\} \mapsto m_2; \ldots; m_k; s\ \{n\} \mapsto^{k-1} p$$
$$m_1; \ldots; m_k; s'\ \{n+1\} \mapsto m_2; \ldots; m_k; s'\ \{n\} \mapsto^{k-1} p',$$

and the claim follows from Lemma A.2.

The case of $m_1 > n + 1$ is analogous to the first one. □

### A.4 Supermodularity

We are now ready to speak about the full scenario and about how it changes depending on the presence of two prefix elements $x$ and $y$. Throughout this section, fix a monotone, convex function $\tilde{f} : \mathbb{N} \to \mathbb{R}$. Additionally, let $1 \leq x < y$, let $m_1, \ldots, m_y \in \mathbb{N}_{>0}$, and let $n \in \mathbb{N}$. Fix four programs:

$$\begin{aligned}
p_{x,y} &\coloneqq m_1; \ldots; m_y; [\tilde{f}]\ \{n\} \\
p_{\bar{x},y} &\coloneqq m_1; \ldots; m_{x-1}; m_{x+1}; \ldots; m_y; [\tilde{f}]\ \{n\} \\
p_{x,\bar{y}} &\coloneqq m_1; \ldots; m_{y-1}; [\tilde{f}]\ \{n\} \\
p_{\bar{x},\bar{y}} &\coloneqq m_1; \ldots; m_{x-1}; m_{x+1}; \ldots; m_{y-1}; [\tilde{f}]\ \{n\}.
\end{aligned}$$

Additionally, let their evaluation results be

$$p_{x,y} \Downarrow r_{x,y} \qquad p_{\bar{x},y} \Downarrow r_{\bar{x},y} \qquad p_{x,\bar{y}} \Downarrow r_{x,\bar{y}} \qquad p_{\bar{x},\bar{y}} \Downarrow r_{\bar{x},\bar{y}}.$$

**Lemma A.4.** *It holds that $r_{x,y} - r_{\bar{x},y} \geq r_{x,\bar{y}} - r_{\bar{x},\bar{y}}$ and $r_{x,y} - r_{x,\bar{y}} \geq r_{\bar{x},y} - r_{\bar{x},\bar{y}}$.*

*Proof of Lemma A.4.* It is sufficient to prove that $r_{x,y} - r_{\bar{x},y} \geq r_{x,\bar{y}} - r_{\bar{x},\bar{y}}$ because we can obtain the inequality $r_{x,y} - r_{x,\bar{y}} \geq r_{\bar{x},y} - r_{\bar{x},\bar{y}}$ by subtracting $r_{x,\bar{y}}$ and adding $r_{\bar{x},y}$.

By Lemma A.2, the first $x - 1$ steps lead to settings of the same shape and a common integer $n'$. Thus, we may assume without loss of generality that $x = 1$.

If $n < m_1$, $p_{x,y} \mapsto p_{\bar{x},y}$ and $p_{x,\bar{y}} \mapsto p_{\bar{x},\bar{y}}$. Then, $r_{x,y} = r_{\bar{x},y}$ and $r_{x,\bar{y}} = r_{\bar{x},\bar{y}}$, which shows the claim.

Thus, assume that $m_1 \leq n$. Now,

$$\begin{aligned}
p_{x,y} &\mapsto m_{x+1}; \ldots; m_y; [\tilde{f}]\ \{n-1\} \\
p_{x,\bar{y}} &\mapsto m_{x+1}; \ldots; m_{y-1}; [\tilde{f}]\ \{n-1\}.
\end{aligned}$$

By Lemmas A.2 and A.3,

$$\begin{aligned}
p_{x,y} &\mapsto^{y-x} m_y; [\tilde{f}]\ \{n'\}, \\
p_{x,\bar{y}} &\mapsto^{y-x} [\tilde{f}]\ \{n'\}, \\
p_{\bar{x},y} &\mapsto^{y-x-1} m_y; [\tilde{f}]\ \{n''\}, \\
p_{\bar{x},\bar{y}} &\mapsto^{y-x-1} [\tilde{f}]\ \{n''\},
\end{aligned}$$

such that $n' \leq n'' \leq n' + 1$. If $n' = n''$, then as above $r_{x,y} = r_{\bar{x},y}$ and $r_{x,\bar{y}} = r_{\bar{x},\bar{y}}$, and we are done.

Assume, therefore, that $n'' = n' + 1$. If $m_y > n' + 1$, then $m_y; [\tilde{f}]\ \{n'\} \mapsto [\tilde{f}]\ \{n'\}$ and $m_y; [\tilde{f}]\ \{n''\} \mapsto [\tilde{f}]\ \{n''\}$. Then, $r_{x,y} = r_{x,\bar{y}}$ and $r_{\bar{x},y} = r_{\bar{x},\bar{y}}$, which shows the claim. Else, if $m_y = n' + 1$, then again $m_y; [\tilde{f}]\ \{n'\} \mapsto [\tilde{f}]\ \{n'\}$ but $m_y; [\tilde{f}]\ \{n''\} \mapsto [\tilde{f}]\ \{n'\}$. Thus $r_{x,y} = r_{\bar{x},y} = r_{x,\bar{y}} = f(n')$ and $r_{\bar{x},\bar{y}} = f(n' + 1)$. The latter term is larger by monotonicity of $\tilde{f}$ and we are done. Else, it must hold that $m_y \leq n'$, thus $m_y; [\tilde{f}]\ \{n'\} \mapsto [\tilde{f}]\ \{n' - 1\}$ and $m_y; [\tilde{f}]\ \{n''\} \mapsto [\tilde{f}]\ \{n'\}$. Now, $r_{x,y} = \tilde{f}(n' - 1)$, $r_{\bar{x},y} = r_{x,\bar{y}} = \tilde{f}(n')$ and $r_{\bar{x},\bar{y}} = \tilde{f}(n' + 1)$. The claim now follows from convexity. □

### A.5 Putting It All Together

*Proof of Theorem 4.2.* As outlined in the beginning of the proof, it suffices to show that Eq. (4) holds for a fixed order of agents, for fixed set of agents $S$ and other agents $x$ and $y$, and for fixed coin flip indices $c_i$ for all agents $i$ in the prefix.

By Lemma A.1, the expected number of left-over jobs is a monotone and convex function $\tilde{o}$ in the number of available jobs on entering the suffix. Due to independence between the coin flips, the expectation conditioned on the coin flips is the evaluation result of $c_{i_1}; c_{i_2}; \ldots; c_{i_k}; [\tilde{o}]\ \{k_{\ell\pi}\}$, where $i_1, \ldots, i_k$ are the prefix agents in their order. Potentially flipping the roles of $x$ and $y$, the inequality then follows from Lemma A.4. □

## B Proof of Theorem 4.3

The proof of the theorem follows rather easily from the following lemma.

**Lemma B.1** (Rabanca 2016)**.** *Given a bipartite graph $G = (A \cup B, E)$, let $f : 2^A \to \mathbb{N}$ be the function that maps $S \subseteq A$ to the size of the maximum matching in the induced graph $G[S \cup B]$. Then $f$ is submodular.*

*Proof of Theorem 4.3.* For each agent $i$ and job $j$, flip a coin with bias $p_{ij}$, and fix the coin flips hereinafter. As before, the employment function is a convex combination of functions corresponding to these fixed coin flips, so it is sufficient to show that each of these functions is submodular.

For each locality $\ell$, consider a bipartite graph $G_\ell = (N \cup J_\ell, E_\ell)$, where $J_\ell$ is the set of jobs in $\ell$, and $(i, j) \in E_\ell$ if and only if the (previously fixed) coin flip corresponding to this edge succeeded. Let $f_\ell : 2^N \to \mathbb{N}$ be the function that maps $N' \subseteq N$ to the size of the maximum matching in the induced graph $G_\ell[N' \cup J_\ell]$. By Lemma B.1, $f_\ell$ is submodular.

Now denote the employment function (for fixed coin flips) by $g$. We can write $g = \sum_{\ell \in L} g_\ell$, where each $g_\ell$ is defined by

$$g_\ell(S) := f_\ell(\{i \in N : (i, \ell) \in S\})$$

for all subsets of agent-locality pairs $S \subseteq 2^{N \times L}$. The submodularity of $g_\ell$ follows directly from the submodularity of $f_\ell$, and therefore $g$ itself is submodular. □

# C Experiments

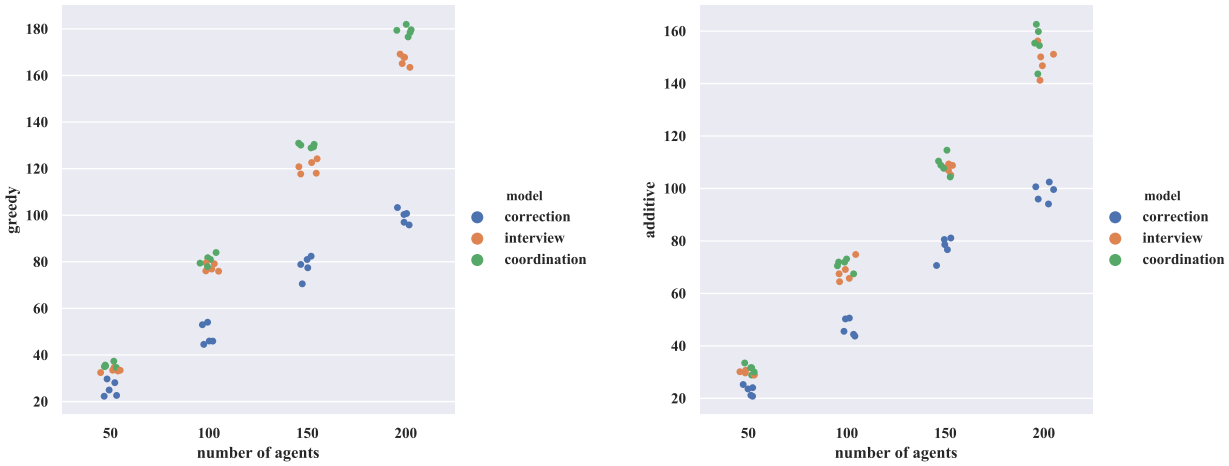
In the following sections, we provide more information about our simulations, namely the two experiments described in the body of the paper and two additional ones. Besides figures and interpretations, each section links to the IPython notebook of the corresponding experiment. These notebooks show the exact code and output of our simulation runs, and they can be downloaded for interactive experimentation.

## C.1 Number of Localities

The first experiment varies the number of localities. Its setting and the resulting diagram in Fig. 1 are described in detail in Section 5. The corresponding notebook can be found at `https://github.com/pgoelz/migration/blob/master/num_localities.ipynb`.

## C.2 Number of Agents

The second experiment varies the number $n$ of agents. This modification of the setting was also described in Section 5: There are 10 localities, each with an equal cap of $n/10$. This experiment produced Fig. 2, but we also generated the following plots of the absolute utilities achieved by both mechanisms:

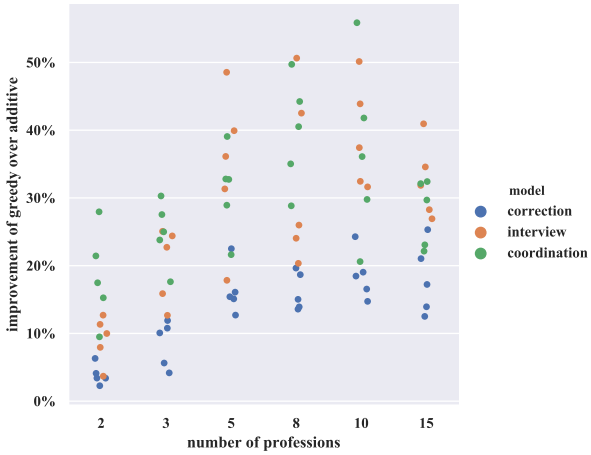



Please mind the different scales of the y-axes. As claimed before, the utility of the additive algorithm is very close to $n/2$ in the correction model, which is the expected value of the total number of people who can potentially be brought into work, since qualification probabilities are selected uniformly from $[0, 1]$. This explains why the percentage improvement of the greedy algorithm over the additive one measured in Fig. 2 is relatively low in these settings.

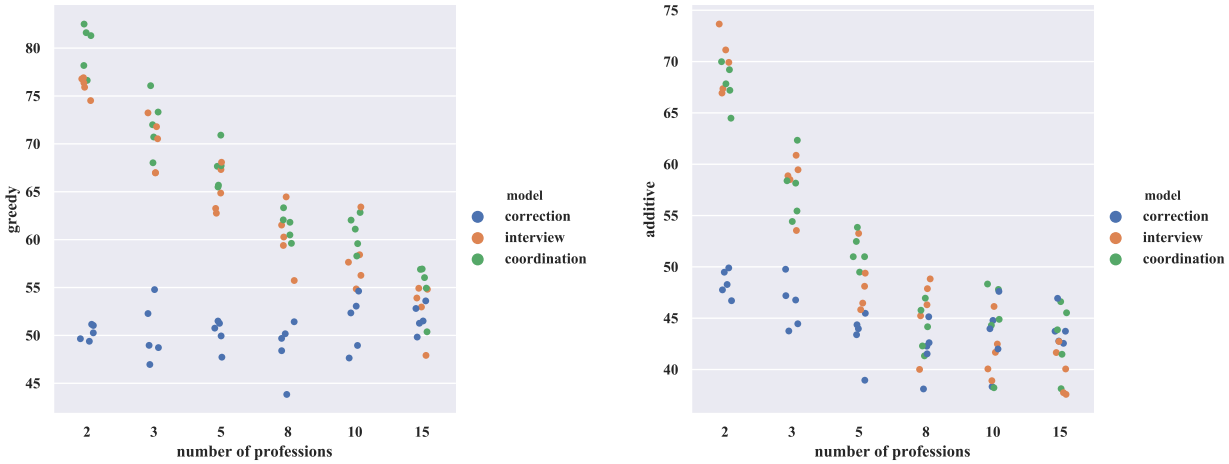
The notebook for the experiment is available at `https://github.com/pgoelz/migration/blob/master/num_agents.ipynb`.

## C.3 Number of Professions

Let there be 10 localities, each with a capacity of 10. The total number of agents is set back to 100. We vary the number of professions, each of which has at least one agent. All other agents are assigned to a uniformly chosen profession. There is a job of the right profession for every agent, and each locality randomly receives 10 of these jobs.



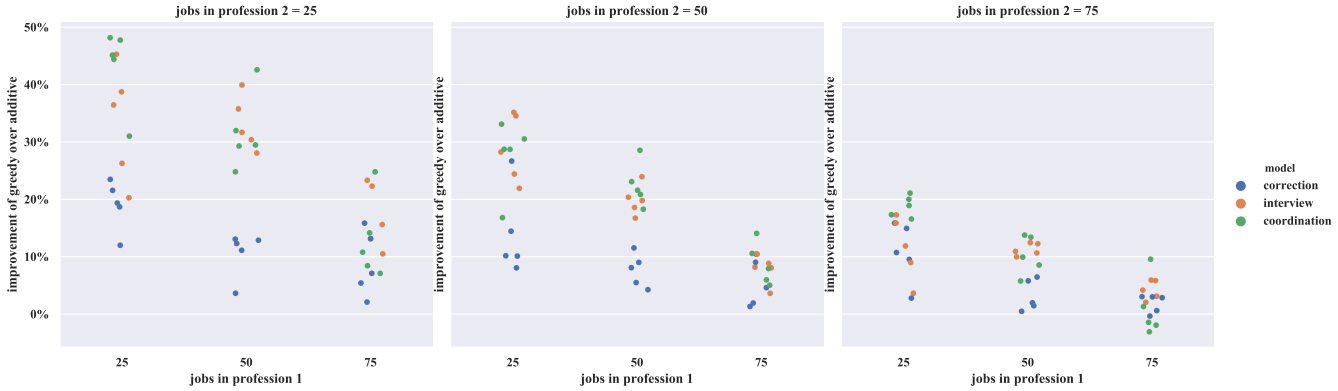

When we increase the number of professions, the gap between greedy and additive increases significantly. Our hypothesis is that, with many professions, a high fraction of matchings are bad just because the agents' professions do not line up with job availability, which disproportionately disadvantages the additive algorithm. This can be observed in the following plot of absolute employment, where the performance of the additive algorithm stagnates on a low level, nearly independently of the model and at numbers of professions at which the greedy algorithm still achieves much higher employment.



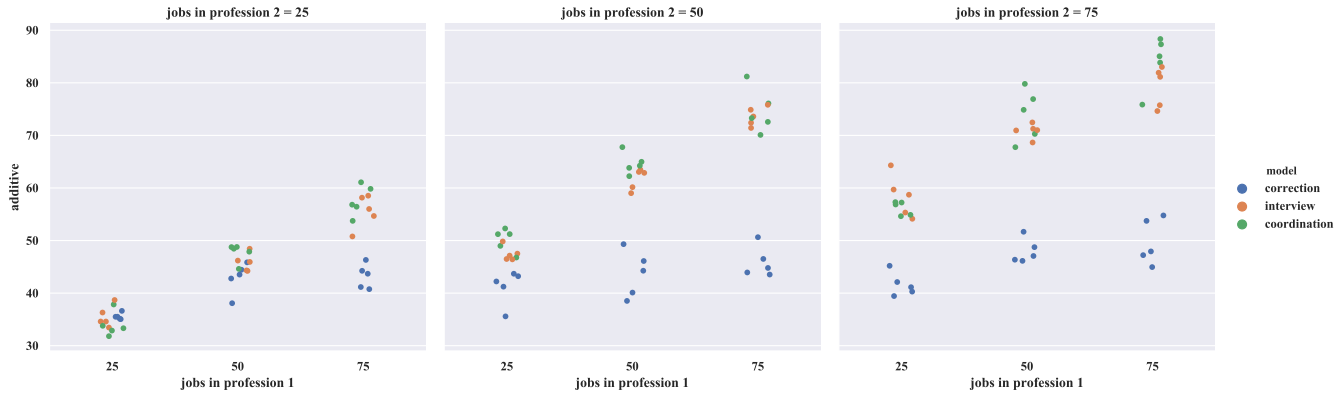

The corresponding notebook can be found at `https://github.com/pgoelz/migration/blob/master/num_professions.ipynb`.

## C.4 Job Availability

In the following experiment, we fix 10 localities, each with a capacity of 10. We have 100 agents, with 50 agents for each of the two professions. Independently of the caps, we vary the number of jobs for both professions, which we distribute randomly between the localities.



Especially when jobs are scarce, i.e., when at least one profession has only 25 jobs for 50 agents, the greedy approach greatly improves upon the additive one. When there is a surplus of both kinds of jobs, the gains again decrease since additive optimization already performs well in these situations, as can be seen in the following plot of absolute utilities:



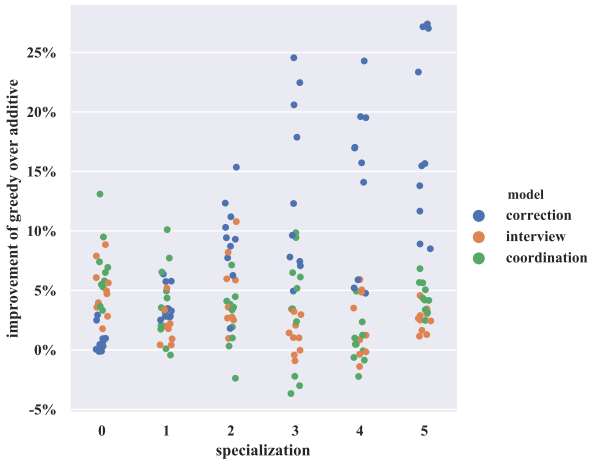

In the correction model, the expected number of people qualifying for employment continues to be 50, which is why employment barely increases when going from 125 total jobs to 150.

Please refer to `https://github.com/pgoelz/migration/blob/master/job_availability.ipynb` for the simulation notebook.

## C.5 Specialization

Our last setting again has 100 agents and 10 localities of capacity 10. The total number of jobs is also 10 per locality now, but we do not distribute the jobs of the two professions randomly.

Instead, we define two kinds of localities, *specialized* and *unspecialized* ones. In an unspecialized locality, both professions have 5 jobs each. In a specialized locality, one profession has 8 jobs, the other just 2. To keep the total number of jobs equal between both professions, we always have $s$ many localities specialized on profession 1 and $s$ many specialized on profession 2, where the *specialization* $s$ is an integer between 0 and 5.




In general, this setting seems to lead to less pronounced gains in terms of utility. Interestingly, by far the biggest improvements can be made in the correction model, probably because the additive algorithm performs particularly bad in this setting. For specialization levels of 3 and 4, we actually see a few cases where the additive algorithm performs slightly better than the greedy one. Still, a majority of simulations in this setting — whose only randomness comes from the agents' probabilities and the sampling noise — see an improvement in utility, and the potential gains are stronger than the potential losses.

We direct the reader to `https://github.com/pgoelz/migration/blob/master/specialization.ipynb` for the corresponding notebook.