# MULTIDIMENSIONAL ALLOCATION: IN APPORTIONMENT AND BIN PACKING

A Thesis

by

Alvin Chiu

Dr. Mohit Singh, Advisor
School of Industrial Engineering

Dr. Sahil Singla, Second Reader
School of Computer Science

Sebastian Perez Salazar, Graduate Student Advisor
School of Industrial Engineering

# Contents

# 1 Part 1: The Multidimensional Apportionment Problem

## 1.1 Introduction

In the apportionment problem, the goal is to allocate a fixed number of seats ("the House seats") to representatives, such that it is proportional to the votes according to the certain dimensions that they represent (most commonly district and party affiliation). Proportionality ensures the principle that each person's vote should have equal weight. However, true proportionality cannot be achieved: the seats of a house cannot be split into fractions as a representative can only represent one seat after all. Doing this allocation precisely is then of great importance, as deciding where the last seat goes may lead to a political majority. Many countries around the world use what are called divisor methods, which answer this question of what it means to be proportional in an integral setting. The idea is to scale the votes by some factor (which preserves the proportions), such that rounding yields an allocation of the proper House size. One common example of divisor methods include the Jefferson/D'Hont method created by Thomas Jefferson in 1792. The Huntington-Hill method is another example, which is currently what the U.S. House of Representatives uses to apportion their 435 seats to the 50 U.S. states.

However, it is also common to desire proportionality in both district population and party votes simultaneously. This extension of apportionment to 2 dimensions now is often called *biproportional apportionment*. Such methods are used by electorate systems such as in Switzerland and Finland. Balinkski and Demange (1989) showed that biproportional apportionments always exist with this divisor method approach. [2] Our research tackles this problem when generalized to an arbitrary $d$-dimensions, building off recent results in this area.

In Cembrano et al. (2021), the authors showed that the multidimensional apportionment problem does not always have a solution unlike the 2-dimensional case. Even the corresponding decision problem of whether an instance has a multidimensional apportionment is NP-complete. Hence, the goal is to find "approximate" solutions, in the sense that we allow small violations to the *marginals*, constraints on how many seats certain districts or parties receive. The apportionment problem is reformulated as an integer linear program (IP), allowing for a primal-dual analysis to show that solutions do not always exist. However, one can use rounding techniques from discrepancy theory on the linear relaxation of the IP to obtain integral solutions at some cost to the marginal constraints. Informally, the field of discrepancy theory tries to minimize how much one deviates from an "ideal" state. Here discrepancy theory allows one to round in such a way that the maximum violations to the marginals in each dimension are under control. Our goal is to tighten the bounds on the maximum violations using state-of-the-art results in discrepancy theory. Cembrano et. al draws inspiration from the classic Beck-Fiala Theorem, however there have been various improvements to that result in the literature.

Improving the approximate solutions to the multidimensional apportionment problem will bring it closer to real-world applications by political systems around the world. Many countries use biproportional (2-dimensional) apportionment methods for representation along party lines and geographic districts, such as Switzerland. But recently, countries have added another dimension to increase representation, such as Chile where the 2021 Chilean Constitutional Convention began including gender balance. Another example is in Bosnia and Herzegovina, which includes ethnicity as a dimension in its representation. Unfortunately, the nonexistence of the ideal solution in 3 or more dimensions presents a challenge to these countries that seek apportionment methods for that purpose. As such, improving the bounds on how far these solutions may deviate from the desired apportionment will allow for these countries to apportion their seats in a more representative manner.

## 1.2 Organization

This thesis will be mostly expository, building up to all the results necessary to fully understand the work being done on multi-dimensional apportionment. However there are also some minor results we obtained that are outlined in Section 2. This is followed by a brief literature review of apportionment in Section 3. Afterwards, we thoroughly investigate the 1 dimensional (vector) apportionment problem in Section 4. There we outline what fairness properties have been desired for apportionment methods, and use this to

motivate the study of just divisor methods, a certain class of apportionment methods. In Section 5, we formally tackle the 1-D and 2-D apportionment problems through the lens of constraint optimization. This allows us to understand multi-dimensional apportionment in Section 6, which begins to behave differently from the 2-D case. Finally, we go into our technical contributions in Section 7.

# 2 Our Contributions

## 2.1 Deviations by Vertex

Our work builds mainly off of Cembrano et al., which was the first work to tackle the problem of apportionment in 3 or more dimensions. They were able to find approximate solutions to the $d$-dimensional proportional apportionment problem if one allows violations in the marginals. This is dictated a deviation vector $u = (u_1, \ldots, u_d)$ satisfying their sufficient condition of $\sum_{i=1}^{d} 1/(u_l + 2) \leq 1$, where $u_l$ represents the maximum allowed violation in the marginals of dimension $l$. As an example, for $d = 3$ dimensions we can choose the deviation vector $u = (0, 2, 2)$, which means that there exists an apportionment where the marginals in the first dimension are satisfied, while the marginals in the second and third may deviate by at most 2. Our main result is to slightly generalize this result so that instead of controlling the violations by dimension, we can control the violations for each item in a dimension (i.e. control the violation in each party or district individually, rather than uniformly across all parties or all districts). Let $N_1, \ldots, N_d$ be sets where $N_l$ is the set of items in a dimension $l \in [d] = \{1, \ldots, d\}$ (i.e. if dimension 1 is districts then $N_1$ is the set of all districts). Then we show that for nonnegative integers $u_v$ for every $l \in [d]$ and every $v \in N_l$, we can find an apportionment with deviations $u_i$ in the marginal for item $i$ if $u$ satisfies the following:

$$\sum_{\ell=1}^{d} \frac{1}{\sum_{v \in N_l} (u_v + 2)/|N_l|} \leq 1$$

This allows more flexibility when it comes to controlling the deviations. One should note that setting $u_v$ to be a constant value within each dimension $l \in [d]$ for all $v \in N_l$ becomes equivalent to the constraint in Theorem 6.3 of $\sum_{l=1}^{d} 1/(u_l + 2) \leq 1$.

## 2.2 Applying Bukh's bound

The authors left as an open question whether there exist vectors $u$ such that $f(u) = \sum_{l=1}^{d} 1/(u_l + 2) > 1$ where solutions exist for any hypergraph instance. Using Bukh's result [8], we are able to answer this question in the affirmative:

**Theorem 2.1.** *The condition $f(u) = \sum_{l=1}^{d} 1/(u_l + 2) \leq 1$ is sufficient but not necessary for defining discrepancy vectors $u$ where solutions to the $d$-partite hypergraph problem always exist. In particular, there always exist solutions to the vector $u = (d - K, d - K, \ldots, d - K) \in \mathbb{Z}^d_{\geq 0}$ for $K = \log^* d/2$, where $\log^*$ is the log tower function.*

For comparison, Theorem 6.3's condition of $\sum_{i=1}^{d} 1/(u_l + 2) \leq 1$ yields the vector $u = (d - 2, \ldots, d - 2)$. If we let $d$ satisfy $6 = \log^* d$, then we get the vector $(d-3, \ldots, d-3)$ which is strictly better (less discrepancy in every dimension). Here $f(u) = d/(d-1) = 1 + 1/(d-1)$ for an extremely large value of $d$. The fact that such vectors where $f(u) > 1$ exist should not come as a surprise; Theorem 6.3 utilized Beck-Fiala and conjectured that it was not the tightest bound.

## 2.3 Applying Lovett-Meka

In 2012, Lovett and Meka created a randomized algorithm that finds a solution with discrepancy $O(\sqrt{t} \log n)$. The approaches obtaining this bound use an idea called *partial coloring*, where each step freezes around half the remaining variables and increases the discrepancy of each set by $O(\sqrt{t})$. This takes $O(\log n)$ iterations to freeze all the variables, resulting in a total discrepancy of $O(\sqrt{t} \log n)$.

We are able to directly apply their algorithm in the Beck-Fiala setting to the $d$-partite hypergraph problem, though it does not take advantage of the $d$-partite hypergraph properties. For example, it does not recover the bipartite graph case ($d = 2$), where we can achieve a discrepancy of 0 among all vertices. Note that there is a better bound of $O(\sqrt{t \log n})$ obtained by Bansal et. al. (2016) [6] which can also be directly applied, but it suffers from the same problem.

A major line for future work in this problem would be in trying to synthesize these results with the $d$-partite hypergraph problem, which would then have implications for the apportionment problem.

# 3 Related Work

The apportionment problem has a rich literature, considering its widespread usage in deomcracies around the world. Balinski and Demange's seminal work in 1989 laid the foundation for the 2-D apportionment problem, establishing the axioms by which we understand what it means to be "proportional" in an integral setting. We refer to a book by Pukelsheim (2014) [18] for more information on proportional apportionment.

The regime of 2-D apportionment without the integrality constraint is more broadly called the Matrix Scaling problem, which has been discovered independently in different fields due to its many applications. This includes transportation science, engineering, economics, numerical analysis, and so on. See Idel (2016) [14] for a literature review on matrix scaling. The most famous algorithm solving matrix scaling is known as Iterative Proportional Fitting (IPF) (also known by RAS method). Sinkhorn (1964) [21] proved the uniqueness and convergence of this iterative algorithm, providing theoretical guarantees in the matrix scaling domain for the first time. For more recent methods in matrix scaling, see Allen-Zhu (2017) [1]. The rich literature of matrix scaling has inspired the methodology used to tackle the closely related apportionment problem. Rothblum (1989) [20] was the first to view matrix scaling through the lens of convex optimization. Pukelsheim (2008) [12] and Rote (2008) [19] were then able to use this approach for the apportionment problem. This optimization approach works in the two regimes over real and integral matrices, thus providing a unifying framework between them.

A very recent work by Cembrano et al. (2021) [9] is the first to study the apportionment problem in arbitrary dimensions of 3 or greater, and makes use of the field of discrepancy theory. Informally, discrepancy theory studies deviations from some ideal state. A celebrated result in combinatorial discrepancy theory comes from Beck and Fiala (1981) [7]. If one wants to measure the "discrepancy" of a set $G$ with some "sparsity" $d$, then Beck-Fiala states that $\text{disc}(G) \leq 2d - 1$. However it is conjectured that one can obtain $\text{disc}(G) \leq O(\sqrt{d})$, so much work has been done to improve upon Beck-Fiala. Bukh (2016) [8] improved this to $\text{disc}(G) \leq 2d - \log^* d$, where $\log^* d$ is the log tower function. If we allow dependency on the ground set of size $n$, then Bansal et al. (2016) [6] constructively obtain $\text{disc}(G) \leq O(\sqrt{d \log n})$. This builds off the result by Lovett and Meka (2012) [15] which constructively obtains a weaker bound of $\text{disc}(G) \leq O(\sqrt{d} \log n)$.

# 4 Which Apportionment Method?

In the apportionment problem, the goal is to allocate a fixed number of seats ("the House seats") proportionally to the votes according to certain dimensions, most commonly the district population or the party affiliation. We first limit ourselves to the one-dimensional case first, for which we use the terms "votes" and "population", as well as "party" and "state" interchangeably since the most common dimension is population (despite not needing any votes). If we have $n$ total states to allocate seats to, then the problem can be formulated as follows:

**Problem 1.** *Given vote vector $v = (v_1, \ldots, v_n) \in \mathbb{N}^n$ and house size $H \in \mathbb{N}$, find an apportionment vector $x = (x_1, \ldots, x_n) \in \mathbb{N}^n$ such that it is proportional to the votes and $\sum_{i=1}^n x_i = H$.*

Here the vote vector $v = (v_1, \ldots, v_n)$ can be thought of as a population vector, where $v_i$ represents the population of state $i$. Now, an *apportionment method* can be understood as a function with vote vectors as input and apportionment vectors as output. We want our apportionment method to satisfy certain properties in practice to be considered fair. For example, one state with a higher population than another should not

receive less seats than it. So our apportionment method should be order-preserving: for states $i$ and $j$, if $v_i > v_j$ then $x_i \geq x_j$. We must also explore the idea of being proportional as it is not exactly well-defined: for any states $i$ and $j$, we ideally want

$$\frac{x_i}{v_j} \approx \frac{x_i}{v_j}$$

One should notice that our apportionment vector $x$ must have integral components, which is what makes this idea of achieving proportionality difficult. If we instead allowed for fractional seats in our apportionment, i.e. $x \in \mathbb{Q}^n$, then we could set our *standard divisor* $D = (\sum_{i=1}^{n} v_i)/H$ and then set our apportionment vector to be $x = v/D$. Here we see that $x$ satisfies our two constraints, the sum of its components is exactly $H$ and we have proportionality because $x_i/v_i = x_j/v_j = 1/D$ for all states $i$ and $j$. This is why we call $D$ the standard divisor, because it achieves the ideal proportion if fractional seats were allowed. As each seat is taken by a representative, the standard divisor $D$ also represents the ideal number of people per one representative. We let the *standard quota* of a state $i$ be $s_i = v_i/D$, as this is the ideal number of seats that state $i$ would receive without the integrality constraint on our apportionment.

## 4.1 Fairness Properties in Apportionment

What kind of deviations from the ideal apportionment are acceptable? What apportionment methods are well behaved to changes in population? One should remember that this whole notion of proportionality is meant to respect the distribution of a population among the states, and so it should also respect changes in the population. Asking these questions lead us to the following desireable properties for an apportionment method [4]:

- **Quota condition**: State $i$ should receive a number of seats within the bounds of its state quota, i.e. $\lfloor s_i \rfloor \leq x_i \leq \lceil s_i \rceil$ (Here we use the floor function and the ceiling function).
- **Population monotonocity**: If state $i$ grows larger relative to state $j$, then state $i$ should not lose seats to state $j$.
- **House monotonicity**: If the total number of house seats is increased, no state's number of seats should decrease.
- **New States consistency**: If a new state is added and its fair share of seats is added to the house size, no other state should have their number of seats changed.

These four properties are typical when considering what apportionment method to use. The quota condition demands that we be as close to the (ideal) state quotas as possible by apportioning to the integers closest to the state quota. The other three properties consider what happens when the parameters of our problem change, whether it be the populations within the state, the house size, or the number of states. There is a large desire for monotonicity in practice, one can imagine the debate that would arise from a state gaining seats despite losing population, or a state losing seats merely from a change in house size. The third parameter of the number of states demands that the method behaves as we expect if states are added (or conversely, removed!). It can be generalized to a stronger condition of coherence, which is that any subproblem of apportioning to a subset of states should agree with the apportionments in the original problem. [18]

## 4.2 Apportionment Methods

We now introduce the apportionment methods that have been used throughout history to tackle this fundamental problem, and subsequently our notion of proportionality. They broadly belong to two main classes: the Hamilton-type methods and the divisor methods.

Hamilton's method springs naturally from the desire to satisfy the Quota condition. First, it computes the standard quotas and gives the floor of it $\lfloor s_i \rfloor$ to each state $i$. For the leftover seats, it distributes them one-by-one based on a priority list from highest remainder to lowest remainder, the remainder being

$s_i - \lfloor s_i \rfloor$. We satisfy the Quota condition from the onset, and the variation in this method comes from what the priority list is based off of. Hamilton's method was the first of its kind and is the most common, hence we call any method that distributes the whole number (floor) of the state quota first a Hamilton-type method. Lowndes's method is another example, where its priority list is based off of each state's remainder divided by the whole number of its state quota.

While Hamilton-type methods worked off of the standard quotas, the divisor methods work off of adjusting the standard divisor. Typically, they start with the standard divisor $D$ to divide the vote vector with, round according to a rule specific to the method, then adjust the divisor such that the sum of the rounded seats is indeed equal to $h$. For example, Jeffersons's method rounds down the fractional seats using the floor function, so one would always need to decrease the standard divisor to reach the desired sum $h$. One can guess some other ways to round: Adam's method rounds up (using the ceiling function) and Webster's method does a natural rounding (round 0.5 up, round anything less down).

In fact, there are two more divisor methods of note that were not first formulated in the form of one, despite being algebraically equivalent to a divisor method. Dean's method also begins with choosing a divisor $d$, then gives each state the number of seats that brings it closest to this divisor, and finally adjusts $d$ to reach the desired sum $h$. If our two rounding choices are $k$ or $k+1$, then Dean's method is equivalent to using the harmonic mean of $k$ and $k+1$ as the threshold for rounding up or down (following the divisor method process). The other one was the Huntington-Hill method, which was formulated to minimize the relative differences of the representatives per people measure between any two states. This uses the geometric mean of our two rounding choices as a threshold instead. One can imagine that historically across many different countries, these apportionment methods were developed independently and in different forms despite being algebraically equivalent.

Technically, there is another class of apportionment methods called "lottery methods" that will only be mentioned to demonstrate the point of needing to use these methods in practice. One can create a roulette of the $n$ states, where the probability that a ball falls into a state is proportional to its population. Then we can play this roulette game $h$ times to determine how many seats each state gets! Such a simple method satisfies both our constraints, in particular it is perfectly unbiased and will be proportional to the populations on average. One can even propose a Hamilton-type method using this, where only the leftover seats are determined using this random process. But one can guess that in reality, such "lottery methods" will never be used because of the inherent randomness and instability that comes with them. Population monotonicity will not be satisfied, so implicit to the apportionment problem is the assumption that certain fairness properties should also hold true. [5]

So before discussing the relative advantages and disadvantages of the seven apportionment methods presented, we should first see whether they respect the fairness properties we set out in the previous section. As it turns out, Hamilton-type methods naturally satisfy the Quota condition, but all are susceptible to violating Population monotonicity. Violations of the House monotonicity and New States consistency were also observed in U.S. history, giving them alternate names (for their violation) of the Alabama paradox and the Oklahoma paradox respectively. The main issue is that once the whole number of each state quota is allocated, the leftover remainders can no longer accurately reflect the relative sizes of each state accordingly (which is core to Population monotonicity). Rounding up or down for a larger state is more restrictive than rounding for a smaller state. On the other hand, divisor methods do not always satisfy the Quota condition.

In questioning whether these seven methods are even good if they violate some fairness property, we present the following results from Balinski and Demange [4]:

**Lemma 4.1.** *Divisor methods are the only apportionment methods satisfying Population monotonicity.*

So we need only confine ourselves to the methods at hand, rather than consider endless possibilities of other apportionment methods. When one focuses on divisor methods only, one can see how they also satisfy House monotonicity and the New States consistency. For House monotonicity, the chosen divisor will need to be slightly decreased, but this can only increase the number of seats a state receives since it is rounded based on passing fixed thresholds. For New States consistency, the number of seats a state gets is fixed once the divisor is fixed, so adding new states or even removing states will not change that number. We can once again view this from the lens of subproblems, where the apportionment of any subset of states will remain

consistent and fair. But one now needs to make a choice on which fairness property to prioritize:

**Theorem 4.2.** *No apportionment method can simultaneously satisfy the Quota condition and Population monotonicity.*

This impossibility theorem then shows that there is no "best" apportionment method, if we take "best" to mean satisfying both the Quota condition and Population monotonicity at the very least. Much like other results in social choice theory, one must choose the apportionment method that best reflects one's policy goals rather than having one sound option triumphing the rest. In particular, the Population monotonicity property is regarded as the most essential in preventing vote manipulation, so divisor methods are used quite universally. As such, we will only consider divisor methods moving forward.

## 4.3 Measuring Bias

With proper motivation to restrict ourselves solely to divisor methods, we now illustrate how the five historical divisor methods are actually the five canonical divisor methods in some sense. Huntington saw two major approaches to measuring the bias of an apportionment method, either through global optimization or a pairwise comparison between states. While this thesis focuses on the global optimization approach as that generalizes to higher dimensions more readily, it is useful to understand the other approach as well.

Huntington's approach was that when comparing any two states, one state will always be favored in that it has a better representation than the other. In that scenario, we consider transferring one seat from the favored state to the less favored state if the "amount of inequality" is reduced between the pair. The key here is deciding what inequality it is that we seek to reduce. If we have states $i$ and $j$, then recall that they have populations $v_i$ and $v_j$ and receive $x_i$ and $x_j$ seats respectively. Then we can say that state $i$ is favored relative to state $j$ iff $x_i/v_i > x_j/v_j$. Recall that this ratio is the number of representatives per population of these states, so the higher this value is the better. One measure of inequality to minimize between any states $i$ and $j$ is then $|x_i/v_i - x_j/v_j|$, that is the difference in representation between people from any two states. From an algorithmic point of view, we can consider starting with an arbitrary apportionment and then make transfers whenever we can reduce this inequality. It turns out that this process will terminate, and in fact it will result in exactly Webster's method! If we can no longer make any transfers, that means that for all states $i$ and $j$ with $x_i/v_i > x_j/v_j$, we satisfy

$$x_i/v_i - x_j/v_j < (x_j + 1)/v_j - (x_i - 1)/v_i$$

Recall that Webster's method selects a divisor $d$ such that dividing the vote vector by $d$ and rounding naturally results in an apportionment vector with sum $h$. Then for any state $i$ we have:

$$\begin{aligned} x_i - 1/2 \leq v_i/d < x_i + 1/2 \\ \frac{x_i - 1/2}{v_i} \leq 1/d < \frac{x_i + 1/2}{v_i} \end{aligned}$$

Since this is also true for state $j$, we can change the indices of the upper bound with $j$:

$$\begin{aligned} \frac{x_i - 1/2}{v_i} \leq 1/d < \frac{x_j + 1/2}{v_j} \\ \frac{x_i - 1/2}{v_i} < \frac{x_j + 1/2}{v_j} \\ \frac{x_i + x_i - 1}{v_i} < \frac{x_j + x_j + 1}{v_j} \\ \frac{x_i}{v_i} - \frac{x_j}{v_j} < \frac{x_j + 1}{v_j} - \frac{x_i - 1}{v_i} \end{aligned} \qquad \text{(for any states } i \text{ and } j\text{)}$$

And note that each step is reversible, so an apportionment method that minimizes this pairwise inequality $|x_i/v_i - x_j/v_j|$ iff it is the Webster's method.

What about other measures of inequality? If we continue to start from one state being more favored than another, we base it around manipulations of $x_i/v_i > x_j/v_j$. There are four terms here which can be on either side of the inequality, so we end up with $2^4$ possible measures through cross-multiplication. Huntington showed that not all of these measures will lead to a process that terminates like in the case of Webster's, but the ones that do are exactly the five historical divisor methods: Jefferson's, Adam's, Webster's, Dean's, and Huntington-Hill's methods [13]! The table below demonstrates which measure they utilize:

| Pairwise Comparison Measure for the Five Canonical Methods | | | | |
|---|---|---|---|---|
| Jefferson's | Adam's | Webster's | Dean's | Huntington-Hill's |
| $x_i(v_j/v_i)-x_j$ | $x_i-x_j(v_j/v_i)$ | $x_i/v_i-x_j/v_j$ | $v_j/x_j-v_i/x_i$ | $\frac{x_i/v_i}{x_j/v_j}-1$ |

One can see that Huntington-Hill's method was originally formulated for exactly this purpose of minimizing the relative differences of the representatives per people measure here. It is then quite a surprise that the other divisor methods developed historically are exactly those that show up through this process of doing pairwise comparisons between states. On the other hand, we will now focus on the global optimization approach to apportionment methods.

# 5 Apportionment through Optimization

## 5.1 Formalizing Divisor Methods

In this section, we formally describe divisor methods and use *signpost sequences* to induce their rounding rules. This is a sequence of signposts $s(1), s(2), \ldots$ such that

$$s(n) \in [n-1, n] \text{ for all } n \in \mathbb{N}$$
$$s(n) \text{ is strictly increasing}$$

The strictly increasing condition is equivalent to saying that there is no $k \in \mathbb{N}$ such that $s(k) = k$ and $s(k+1) = k$. Given a signpost sequence $s(n)$, we now have the following rounding rule $[\![\cdot]\!]_s$ such that $[\![0]\!]_s = \{0\}$ and

$$[\![t]\!]_s := \{n \in \mathbb{N} \mid s(n) \leq t \leq s(n+1)\} \text{ for all } t > 0$$

Alternatively, this can be viewed as

$$[\![t]\!]_s = n \iff s(n) < t < s(n+1)$$
$$[\![t]\!]_s \in \{n-1, n\} \iff t = s(n)$$

So we can say $[\![t]\!]_s = n$ represents "t is rounded to n" when it does not equal a signpost. When $t$ does equal a signpost, these are in some sense "ties" so there are two choices that $t$ can be rounded to. This is the manner in which we "round", note that $s(n) = n$ acts like the floor function and $s(n) = n - 1/2$ acts like the "natural" rounding scheme (round up if the decimal is 0.5 or more, otherwise round down).

In the Vector Apportionment problem (Problem 1), we had vote vector $v = (v_1, \ldots, v_n)$ and house size $h$, and wanted to find an apportionment vector $x = (x_1, \ldots, x_n)$ where $n$ is the number of states. We first focus on the states that obtain a non-zero amount of votes, i.e. we define the support of our vote vector to be $supp(v) := \{j \mid v_j \neq 0\}$. Now we want an apportionment vector $x$ to be "proportional" to vote vector $v$ and satisfy the house condition $\sum_{j=1}^{n} x_j = h$.

For all states $j_1, j_2$, we want our vote vector $v$ and apportionment vector $x$ to satisfy:

$$\frac{x_{j_1}}{v_{j_1}} \approx \frac{x_{j_2}}{v_{j_2}}$$

To avoid cases where $v_j = 0$, define

$$\mathbb{N}_v^n = \{x \in \mathbb{N}^n \mid \text{supp}(x) \subseteq \text{supp}(v)\} \text{ when } s(1) > 0$$

For the sake of clarity, we will ignore signpost sequences where $s(1) = 0$, as this is an edge case that doesn't change the results. Note that we have $\text{supp}(x) = \text{supp}(v)$ in this case, that is every state with non-zero population receives at least one seat. Now an apportionment vector $x \in N_v^n$ is *feasible* when it satisfies the house condition $\sum_{j=1}^n x_j = h$. Using the idea of signpost sequences, we delimit the range in which we tolerate this approximate equality between proportions:

$$\frac{s(x_{j_1})}{v_{j_1}} \leq \frac{x_{j_1}}{v_{j_1}} \approx \frac{x_{j_2}}{v_{j_1}} \leq \frac{s(x_{j_2}+1)}{v_{j_2}} \text{ for all } j_1, j_2 \in \text{supp}(v)$$

$$\implies \max_{j \in \text{supp}(v)} \frac{s(x_j)}{v_j} \leq \min_{j \in \text{supp}(v)} \frac{s(x_j+1)}{v_j}$$

This now yields our *critical inequality*. For divisor methods, we will now refer to the divisor $d$ as a scalar $\lambda = 1/d$. Originally we divided the vote vector by $d$, but now we will just scale it by $\lambda$. So a divisor method would have a signpost sequence $s(n)$ that induces a rounding rule, from which we can take any vote vector $v$ and calculate our apportionment vector $x$ through the following process: $v_j \mapsto \lambda v_j \mapsto x_j \in [\![ \lambda v_j ]\!]_s$. The claim is that any scaling multiplier $\lambda > 0$ in the *multiplier interval*

$$\left[ \max_{j \in supp(v)} \frac{s(x_j)}{v_j}, \min_{j \in supp(v)} \frac{s(x_j+1)}{v_j} \right] \tag{1}$$

will yield an apportionment vector $x$. We have that $\lambda > 0$ in this interval satisfies $s(x_j) \leq \lambda v_j \leq s(x_j+1) \implies x_j \in [\![ v_j ]\!]_s$ for all states $j$. We can now state our vector apportionment problem via divisor methods:

**Problem 2.** *(Vector Apportionment) Given signpost sequence $s$, vote vector $v$, and house size $h$, find apportionment vector $x$ s.t.:*

- *There exists scaling multiplier $\lambda > 0$ such $x_j \in [\![ \lambda v_j ]\!]_s$ for all states $j$.*
- *Its components add up to the house size $h$: $\sum_{j=1}^n x_j = h$*

Recall that signpost sequences encounter a tie when $t = s(n)$, allowing $[\![ t ]\!]_s \in \{n-1, n\}$. This was necessary if we consider the following example: let $s(n) = n$, $v = (50, 50)$, and $h = 9$. Then we are using the floor function to round, and we only have two states with identical populations of 50. No matter how we scale $v$, the two states will always have the same population and round down to the same value. Thus, there is no way to possibly achieve an odd sum of $h = 9$ when the two components of our described apportionment vector are equal and will have an even sum. This necessitates the need for a way to break ties using our signpost sequence: we can now choose $\lambda = 1/10$, then $[\![ 5 ]\!]_s \in \{4, 5\}$ so we can end up with an apportionment vector $x = (4, 5)$ or $(5, 4)$.

This tie-breaking ability is key to showing why there is always a $\lambda$ satisfying the house condition. We have that the $\sum_{i=1}^n [\![ \lambda v_i ]\!]_s$ is the sum of integer parts, and we continuously decrease/increase $\lambda$ until it is close to $h$. If it never equals $h$, then that means this sum made a discrete "jump" of size 2 or more at some $\lambda'$. According to our signpost sequences, these jumps happen exactly when $\lambda' v_i = s(n)$ for some integer $n$. But since it made a jump of size 2 or more, we have at least two states $i$ and $j$ where $\lambda' v_i = s(n)$ and $\lambda' v_j = s(m)$ for integers $m, n$. However, due to the tie-breaking property, we are able to choose whether each component $[\![ \lambda v_i ]\!]_s$ jumps by 1 or not. Hence, we can always choose to make the sum make jumps of size 1 only, allowing us to reach the target sum $h$.

The tie-breaking property may feel like cheating in how it modifies typical rounding functions like floor, but in practice ties in apportionment almost never happen. The need to define it as such is only to overcome the theoretical edge cases as described above. We now proceed with the optimization framework for apportionment.

## 5.2 A Vector Optimization Problem

The treatment on divisor methods specifically ends up lending nicely to the optimization perspective of apportionment. Pukelsheim first introduced a novel optimization approach in 2008, where he showed that the Vector Apportionment Problem (Problem 2) is equivalent to the following problem [12]:

**Problem 3.** *Define*

$$F_v(x) := \prod_{j \in supp(v)} \prod_{n \leq x_j : s(n) > 0} \frac{s(n)}{v_j}$$

*Now the optimization problem is*

$$\begin{aligned} &\textit{Minimize } F_v(x) \\ &\textit{subject to } \sum_{j=1}^{n} x_j = h \\ &\textit{over the set } x \in \mathbb{N}_v^n \end{aligned}$$

Let's digest this goal function: first it should lead to a convex goal function. It should help to look at it after taking the log of it:

$$\begin{aligned} \log F_v(x) &= \sum_{j \in \text{supp}(v)} \sum_{n \leq x_j : s(n) > 0} \log\left(\frac{s(n)}{v_j}\right) \\ &\leq \sum_{j \in supp(v)} \log\left(\frac{x_j!}{v_j^{x_j}}\right) \\ &\approx \sum_{j \in supp(v)} \log\left(\frac{x_j^{x_j}}{v_j^{x_j}}\right) \qquad \text{(by Stirling's approximation)} \\ &= \sum_{j \in supp(v)} x_j \log\left(\frac{x_j}{v_j}\right) \end{aligned}$$

We can think of the vote vector $v$ as modeling a multinomial distribution, with cell probabilities $w_j = v_j / \sum_{j=1}^{n} v_j$ and sample size $h$. Then finding the most proportional apportionment vector $x$ (also understood as a multinomial distribution) under the house sum constraint would be that vector which is "closest" to the vote vector $v$ by some measure of distance. In this case, we make this measure of distance our goal function to minimize, which is approximately the KL-divergence $D_{KL}(x \mid\mid v)$ between these two multinomial distributions $x$ and $v$.

Now we state the claim of problem equivalence:

**Theorem 5.1.** *(Optimality) The following statements are equivalent for all feasible apportionment vectors $x \in N_v^n$:*

*(1) $x$ solves Problem 2.*

*(2) $x$ satisfies the critical max-min inequality (1).*

*(3) $x$ is an optimal solution of Problem 3.*

The proof goes (1) $\iff$ (2) by definition and then (2) $\iff$ (3). Note that Problem 3 is an optimization problem over a finite set, so a solution must exist. Hence, this theorem immediately shows that a solution to the Vector Apportionment problem (Problem 2) always exists (namely we can find a multiplier $\lambda > 0$ to satisfy the house sum constraint).

**Corollary 5.1.1.** *(Uniquness) The following statements are equivalent for all feasible apportionment vectors $x \in N_v^n$:*

*(1) There is only one feasible apportionment vector.*

*(2) The max-min inequality holds with strict inequality.*

The negation is true, i.e. if the max-min inequality holds with equality (so $\lambda$ is unique), then there are multiple feasible apportionment vectors. But one can see how this results from the possibility of ties, which are rare and barely different. Otherwise, one can imagine that if a divisor method came up with two quite different but feasible apportionments, there would be a heavy debate on which one to use.

Another benefit of this optimization approach is the primal and dual view of the problem. Define $x_+$ to be the sum of the components of a vector $x$, i.e. $x_+ = \sum_{i=1}^{n} x_i$. Our multiplier $\lambda > 0$ can be viewed as a dual variable (or Lagrange multiplier) such that for feasible apportionment vectors $x$, we have the following dual goal function:

$$F_v(x) = \lambda^{h-x_+} F_v(x) \geq \inf_{y \in N_v^n} \lambda^{h-y_+} F_v(y) := G_v(\lambda)$$

This shows that the dual goal function is a lower bound on $F_v(x)$ as desired. With this we have duality:

**Lemma 5.2.** *For all $\lambda > 0$, we have $G_v(\lambda) = \lambda^{h-z_+} F_v(z)$ if there exists $z$ such that $z_j \in [\![ \lambda v_j ]\!]_s$ for all states $j$.*

The dual problem can now be stated as follows:

$$\begin{aligned} &\text{Maximize } G_v(\lambda) \\ &\text{subject to } \lambda \in (0, \infty) \end{aligned}$$

This lemma shows that the optimal solution to both the primal and dual are equal, which by the Optimality Theorem 5.1 implies that it also solves the Vector Apportionment problem.

## 5.3 Matrix Apportionment Problem (2-D)

We now extend the apportionment problem to 2 dimensions, where we work with vote matrices and apportionment matrices instead.

**Problem 4.** *A matrix apportionment problem is a tuple $(v, r, c, h)$ where our vote matrix $v$ has nonnegative integer entries $v_ij$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$ and no row or column of all 0's, the row marginals are $r = (r_1, \ldots, r_m) > 0$ and column marginals are $c = (c_1, \ldots, c_n) > 0$ where $\sum_{r_i} = \sum_{c_j} = h$. The goal is to find an apportionment matrix $x$ with nonnegative integer entries such that $\sum_j x_{ij} = r_i$ for all $i$ and $\sum_i x_{ij} = c_j$ for all $j$.*

Here we have a $m \times n$ vote matrix $v$, where we can think of the $m$ rows as being $m$ different states and the $n$ columns as being $n$ different political parties. Then each entry $v_{ij}$ represents the votes received in state $i$ for party $j$. Then the row marginals $r_i$ represent the number of seats that state $i$ should receive, and the column marginals $c_j$ represent he number of seats that party $j$ should receive. The row marginals should sum to the house size $h$ as should the column marginals. The goal then is to find an apportionment matrix $x$ that satisfies these row and column marginals.

Note that if we did not have the integrality constraint for our matrices, then this problem becomes the classic Matrix Scaling Problem. While this problem will not be discussed, approaches for tackling the Matrix Apportionment Problem have often drawn on approaches used to solve the Matrix Scaling Problem such as the optimization framework.

In their seminal work in 1989, Balinski and Demange showed that the divisor method also obtains feasible apportionments even in the Matrix Apportionment problem. In addition, they satisfy much of the same fairness properties we established and would expect from the one-dimensional case of Vector Apportionment. [3]

**Theorem 5.3.** *Given signpost sequence $s$, vote matrix $v$, and house size $h$, there exists a feasible apportionment matrix $x$ satisfying the row and column marginals where:*

- *There exists scaling multipliers $\lambda = (\lambda_1, \ldots, \lambda_m)$ and $\mu = (\mu_1, \ldots, \mu_n)$ such that:*

$$x_{ij} \in [\![ \lambda_i v_{ij} \mu_j ]\!]_s \text{ for all states } i \text{ and parties } j$$

In this extension of divisor methods to 2-D, note that we now have scalars for each row and each column. Each entry $v_{ij}$ in our vote matrix is then scaled by their corresponding row scalar $\lambda_i$ and column scalar $\mu_j$, and then rounded according to the signpost sequence $s$ to obtain its corresponding entry $x_{ij}$.

Once again, we may view this through the lens of optimization. The optimization formulation very readily carries over to the 2-D case:

Define

$$F_v(x) := \prod_{(i,j)\in supp(v)} \prod_{n\leq x_{ij}:s(n)>0} \frac{s(n)}{v_{ij}}$$

Now the optimization problem is

$$\begin{aligned}&\text{Minimize } F_v(x)\\ &\text{subject to } \sum_j x_{ij} = r_i \ \forall i \text{ and } \sum_i x_{ij} = c_j \ \forall j\\ &\text{over the set } x \in \mathbb{N}_v^{m\times n}\end{aligned}$$

Once again, Pukelsheim showed that this optimization problem admits the same solutions of apportionment matrix $x$ as the matrix apportionment problem, and the same holds for the dual optimization problem. [12]

We can see that a benefit of the optimization approach is that it generalizes well into higher dimensions. The results in the 1-D case hold true for the 2-D case quite naturally. But once we jump to 3 or more dimensions, this no longer becomes the case.

# 6 Multidimensional Apportionment Problem

We can now discuss what occurs to the apportionment problem in the case of 3 or more dimensions. Surprisingly, the only work done on this problem is that of Cembrano et. al. [9], so that will be out sole reference. Spurred by a change in the Chilean Constitutional Convention election system in 2021, they showed that in general a multidimensional proportional apportionment may not exist (in contrast to the 1D and 2D case). Their approach leverages the aforementioned optimization framework.

For the *d-dimensional apportionment problem*, we are given as input the tuple $(\mathcal{N}, \mathcal{V}, m^-, m^+, H)$ defined as follows. The set $\mathcal{N} = \{N_1, N_2, \ldots, N_d\}$ can be understood as a family of sets $N_l$ for each dimension $l \in \{1, \ldots, d\}$. Each set $N_l$ represents all the possible values in our specified dimension. For example, we will commonly have dimension 1 be districts and dimension 2 be political party, so $N_1$ is the set of all districts and $N_2$ is the set of all political parties. Then $\mathcal{V}$ is a tensor $\mathcal{V} : N_1 \times N_2 \times \ldots \times N_d \to \mathbb{N}_{\geq 0}$, that is it assigns nonnegative integer entries to each item in the Cartesian product $\prod_{l=1}^d N_l$ (akin to the "vote" vector if we unravel the tensor). Each entry in $\mathcal{V}$ represents the votes garnered by any possible representative in $\prod_{l=1}^d N_l$. We have that for each dimension $l \in [d]$ and each $v \in N_l$, $m_v^-$ and $m_v^+ \in \mathbb{N}_{>0}$ are the *lower marginals* and *upper marginals* for $v$, much like the marginals we saw in the 2-D case but now they can act as lower and upper bounds. Finally, $H \in \mathbb{N}$ is the House size as seen before. We use $E(\mathcal{V})$ to be the support of our vote tensor $\mathcal{V}$, so we only focus on representatives that have a non-zero amount of votes.

**Problem 5.** *Given signpost sequence $s$ and d-dimensional instance $(\mathcal{N}, \mathcal{V}, m^-, m^+, H)$, does there an apportionment tensor $x \in \mathbb{N}^{E(\mathcal{V})}$ such that there exists a scalar $\mu > 0$ and a scalar tensor $\lambda : \prod_{l=1}^d N_l \to \mathbb{R}_{>0}$*

*such that:*

$$m_v^- \leq \sum_{e \in E(\mathcal{V}): e_l = v} x_e \leq m_v^+ \qquad \forall l \in [d] \text{ and } \forall v \in N_l \tag{2}$$

$$\sum_{e \in E(\mathcal{V})} x_e = H \tag{3}$$

$$s(x_e) \leq \mathcal{V}_e \cdot \mu \cdot \prod_{l=1}^{d} \lambda_{e_l} \leq s(x_e + 1) \qquad \forall e \in E(\mathcal{V}) \tag{4}$$

$$\sum_{e \in E(\mathcal{V}): e_l = v} x_e = m_v^- \qquad \lambda_v > 1, \forall l \in [d] \text{ and } \forall v \in N_l \tag{5}$$

$$\sum_{e \in E(\mathcal{V}): e_l = v} x_e = m_v^+ \qquad \lambda_v < 1, \forall l \in [d] \text{ and } \forall v \in N_l \tag{6}$$

As in the 2-D and 1-D case, one can set up this problem equivalently as an optimization problem. We omit that here due to its technical details, but this approach once again remains fruitful. In fact, the primal-dual approach is used to show that solutions do not exist for every instance of the $d$-dimensional apportionment problem with stationary signpost sequences and $d \geq 3$. Here stationary signpost sequences are those where $s(n) = n - \Delta$ for some constant $\Delta \in [0, 1]$. This is in contrast to the 2-dimensional and 1-dimensional case, where solutions always existed. The difference is that for $d = 2$ the linear relaxation of the omitted optimization formulation is integral due to total unimodularity, so solutions always exist. [9]. They also show the following hardness result on the corresponding decision problem of determining if the set of feasible apportionments is non-empty:

**Theorem 6.1.** *For every signpost sequence $s$ and every $d \geq 3$, the $(d, s)$-proportional apportionment problem is NP-complete.*

Once again, the comparison here is that the $(2, s)$-proportional apportionment problem can be solved in polynomial time due to algorithms developed by Balinski and Demange [2]. In the face of these results, some constraint in our problem needs to be relaxed in order to yield some solution. In particular, the authors of the paper by Cembrano et. al. decided to allow violations in the marginals. Their methodology is as follows:

- Find the optimal solution of the linear relaxation to the optimization formulation of apportionment.
- If it is integral, we are done (great!). Otherwise, take our optimal solution and allocate the whole number parts of the components. For the remaining fractional parts of the components, round this in such a way as to minimize the deviations from the marginals and then allocate those.

It is the second step that needs to be further investigated, as the manner in which we round is up in the air. Remarkably, the authors are able to use results from a field of mathematics called Discrepancy Theory in order to minimizing the deviations from the marginal:

**Theorem 6.2.** *Let $(\mathcal{N}, \mathcal{V}, m^-, m^+, H)$ be an instance of the $d$-dimensional apportionment problem and let $s$ be a signpost sequence such that the linear relaxation of the optimization formulation is feasible. Let $u_1, \ldots, u_d$ be nonnegative integers such that $|sum_{i=1}^{d} 1/(u_l + 2) \leq 1$. Then, there exists an integral vector $X \in \mathbb{N}^{E(\mathcal{V})}$ such that the following holds:*

- *$m_v^- - u_l \leq \sum_{e \in E(\mathcal{V}): e_l = v} X_e \leq m_v^+ + u_l$ for every $l \in [d]$ and every $v \in N_l$.*
- *There exists $\mu > 0$ and a vector $\lambda$ with strictly positive entries such that:*
  - *(1) $s(X_e) \leq \mathcal{V}_e \cdot \mu \cdot \prod_{l=1}^{d} \lambda_{e_l} \leq s(X_e + 1)$ for every $e \in E(\mathcal{V})$.*
  - *(2) For every $l \in [d]$ and every $v \in N_l$, if $\lambda_v > 1$ then $|\sum_{e \in E(\mathcal{V}): e_l = v} X_e - m_v^-| \leq u_l$.*
  - *(3) For every $l \in [d]$ and every $v \in N_l$, if $\lambda_v < 1$ then $|\sum_{e \in E(\mathcal{V}): e_l = v} X_e - m_v^+| \leq u_l$.*

*Furthermore, $X$ can be found in time polynomial in $|E(\mathcal{V})|, \sum_{l=1}^{d} |N_l|$, and $H$.*

Key to this result is the "deviation" vector $u = (u_1, \ldots, u_d)$ that satisfies $\sum_{i=1}^{d} 1/(u_l + 2) \leq 1$. Each $u_l$ for dimension $l \in [d]$ dictates how much the marginals in that dimension $l$ can be violated by at most. So for $d = 2$, the result by Balinski and Demange is recovered as the deviation vector $(0, 0)$ implies no violation in any dimension. However for $d = 3$, this establishes a Pareto frontier of deviation vectors $\{(0, 1, 4), (0, 2, 2), (1, 1, 1)\}$ that satisfy the sufficient condition. One can choose which deviation vector to use depending on the policy goals, i.e. if one demands no deviations in the marginals for the party dimension one can use the vector $(0, 1, 4)$.

In order to show Theorem 6.2, one need only focus on the step where we decide on how to round the remaining fractional parts of the optimal solution to the optimization formulation. To that end, there is no longer a need to think about apportionment, as we extract out a problem of rounding weights on a $d$-partite hypergraph. We investigate the following theorem:

**Theorem 6.3** (Cembrano et al. 2021 [9])**.** *Let $G$ be a $d$-partite hypergraph with vertex partition $V = \{P_1, \ldots P_d\}$ and hyperedges $E$, and let $x \in [0, 1]^E$ be such that $\sum_{e \in \delta(v)} x_e$ is integral for every vertex $v \in V$, where $\delta(v)$ is the set of hyperedges containing $v$. Also let $u = (u_1, \ldots, u_d) \in \mathbb{N}^d_{\geq 0}$ such that $\sum_{\ell=1}^{d} 1/(u_\ell + 2) \leq 1$. Then there exists $z \in \{0, 1\}^E$ such that for every $\ell \in [d]$ and every $v \in P_\ell$ it holds that $|\sum_{e \in \delta(v)} (z_e - x_e)| \leq u_\ell$, and additionally $z_e = x_e$ when $x_e$ is integer. The calculation of $z$ can be performed in time polynomial in $|E|$ and $\sum_{\ell=1}^{d} |P_\ell|$.*

We can understand $|\sum_{e \in \delta(v)} (z_e - x_e)|$ to be the discrepancy in each of the dimensions $l \in [d]$ for $v \in P_l$. To achieve this result, Cembrano et. al. construct an iterative rounding algorithm defined in Algorithm 1, inspired by the classic discrepancy minimization result by Beck and Fiala. Given a vector $Y \in [0, 1]^F$ with $F \subseteq E$, a subset of edges $\mathcal{E} \subseteq F$, and a subset of vertices $Q_\ell \subseteq P_\ell$ for each $\ell \in \{1, \ldots, d\}$, we consider the following linear program with variables $y_e$ for each $e \in \mathcal{E}$:

$$\sum_{e \in \delta(v) \cap \mathcal{E}} y_e = \sum_{e \in \delta(v) \cap \mathcal{E}} Y_e \quad \text{for every } v \in \bigcup_{\ell=1}^{d} Q_\ell, \tag{7}$$

$$0 \leq y_e \leq 1 \quad \text{for every } e \in \mathcal{E} \tag{8}$$

We denote $\mathcal{K}(Y, \mathcal{E}, Q)$ to be the polytope of feasible solutions for this linear program. Algorithm 1 iteratively solves a linear program in the form so the condition in the loop guarantees that the algorithm makes progress in fixing at least one new variable into a binary value. Once the loop condition is not satisfied, the algorithm rounds up or down the rest of the fractional variables and its output satisfies the properties guaranteed by Theorem 6.3.

# 7 Our Technical Contributions

## 7.1 Applying Bukh's bound

With regards to the bound in Cembrano et al., they showed that it cannot be strictly improved. Denote $f(u) = \sum_{l=1}^{d} \frac{1}{u_l + 2}$. For $d = 3$, a solution does not exist with discrepancy vector $u = (0, 0, K)$ for any constant $K$. By induction over the dimension, Cembrano et al. showed that finding solutions is also impossible for $u = (0, 0, K, \ldots, K)$. Note that this vector can be made arbitrarily close to 1 with $f(u) > 1$. The authors leave as an open question whether there exist other vectors $u$ such that $f(u) > 1$ where solutions do exist for any hypergraph instance. Using Bukh's result [8], we are able to answer this question in the affirmative. This shows that their condition that $f(u) \leq 1$ for possible discrepancy vectors is not necessary.

**Theorem 7.1.** *The condition $f(u) = \sum_{l=1}^{d} 1/(u_l + 2) \leq 1$ is sufficient but not necessary for defining discrepancy vectors $u$ where solutions to the $d$-partite hypergraph problem always exist. In particular, there always exist solutions to the vector $u = (d - K, d - K, \ldots, d - K) \in \mathbb{Z}^d_{\geq 0}$ for $K = \log^* d/2$, where $\log^*$ is the log tower function.*

**Algorithm 1** Iterative Rounding Algorithm in [9]

**Require:** A $d$-partite hypergraph $G$ with vertex partition $\{P_1, ..., P_d\}$ and hyperedges $E$, a vector $x \in [0,1]^E$ and nonnegative integer values $u_1, ..., u_d$.
**Ensure:** Binary vector $z \in \{0,1\}^E$.
  Initialize $y^0 \leftarrow x$
  Let $\mathcal{E}^0 = \{e \in E : y_e^0 \text{ is fractional}\}$
  For each $\ell \in 1, ..., d$, let $Q_\ell^0 = \{v \in P_\ell : \left|\delta(v) \cap \mathcal{E}^0\right| \geq u_\ell + 2\}$
  Let $z_e = y_e^0$ for every $e \notin \mathcal{E}^0$.
  Initialize $t \leftarrow 0$
  **while** there exists $\ell \in 1, ..., d$ such that $Q_\ell^t \neq \emptyset$ **do**
    Compute an extreme point $y^{t+1}$ of $\mathcal{K}(y^t, \mathcal{E}^t, Q^t)$
    Let $\mathcal{E}^{t+1} = \{e \in E : y_e^{t+1} \text{ is fractional}\}$
    For each $\ell \in 1, ..., d$, let $Q_\ell^{t+1} = \{v \in P_\ell : |\delta(v) \cup \mathcal{E}^{t+1}| \geq u_\ell + 2\}$
    Let $z_e = y_e^{t+1}$ for every $e \in \mathcal{E}^t \setminus \mathcal{E}^{t+1}$. Update $t \leftarrow t+1$
  **end while**
  Let $T$ be the value of $t$ that did not satisfy the loop condition
  Let $z_e \in \{\lfloor y_e^T \rfloor, \lceil y_e^T \rceil\}$ for every $e \in \mathcal{E}^T$
  Return $z$.

*Proof.* Theorem 6.3 shows that the condition $f(u) \leq 1$ is sufficient. Now consider the vector $u = (d-K, d-K, \ldots, d-K) \in \mathbb{Z}_{\geq 0}^d$ for $K = \log^* d/2$. Because $u$ is a vector with uniform elements, we can directly apply Bukh's discrepancy bound of $2d - \log^* d$ to the $d$-partite hypergraph [8]. Given a hypergraph $G = (P, E)$ and starting point $x' \in [-1,1]^E$, there always exist $z' \in \{-1,1\}^E$ that satisfying our discrepancy constraint such that $|\sum_{e \in \delta(v)}(z'_e - x'_e)| \leq 2d - 2K$ for all $v \in P$. Now we transform $x'$ to $x$ where $x \in [0,1]^E$ as in the $d$-partite hypergraph problem. By setting $x = (x'+1)/2$ and $z = (z'+1)/2$, we get that:

$$\begin{aligned}\left|\sum_{e \in \delta(v)} (z_e - x_e)\right| &= \left|\sum_{e \in \delta(v)} ((z'_e + 1)/2 - (x'_e + 1)/2)\right| \\ &= \frac{1}{2}\left|\sum_{e \in \delta(v)} (z'_e - x'_e)\right| \\ &\leq d - K\end{aligned}$$

Now we can check that the vector $u = (d-K, d-K, \ldots, d-K)$ does not satisfy the condition $f(u) \leq 1$:

$$f(u) = \sum_{l=1}^{d} \frac{1}{u_l + 2} = \frac{d}{d-K+2} = 1 + \frac{K-2}{d-K+2} > 1$$

where the last step comes from the fact that $K = \log^* d/2 \ll d$. Hence, $f(u) \leq 1$ is a sufficient but not necessary condition. □

For comparison, Theorem 6.3 yields the vector $u = (d-2, \ldots, d-2)$. If we let $d$ satisfy $6 = \log^* d$, then we get the vector $(d-3, \ldots, d-3)$ which is strictly better (less discrepancy in every dimension). Here $f(u) = d/(d-1) = 1 + 1/(d-1)$ for an extremely large value of $d$. The fact that such vectors where $f(u) > 1$ exist should not come as a surprise; Theorem 6.3 utilized Beck-Fiala and conjectured that it was not the tightest bound.

## 7.2 Applying Lovett-Meka

In 2012, Lovett and Meka created a randomized algorithm that finds a solution with discrepancy $O(\sqrt{t}\log n)$. In 2016, Bansal et. al. improved upon this to obtain a better bound of $O(\sqrt{t\log n})$.

We are able to directly apply Lovett and Meka's algorithm in the Beck-Fiala setting to the $d$-partite hypergraph problem:

**Theorem 7.2** (Main Partial Coloring Lemma, Lovett-Meka 2012 [15])**.** *Let $x^* \in [0,1]^E$ be a "starting point". Let $c_v$ be thresholds s.t. $\sum_{v \in P} \exp(-c_v^2/16) \leq |E|/16$. Then there exists an efficient randomized algorithm which finds a point $x \in [0,1]^E$ such that:*

- $|\sum_{e \in \delta(v)} (x_e - x_e^*)| \leq c_v \sqrt{|\delta(v)|}/2$.
- *$|x_i| \geq 1 - \delta$ for at least $|E|/2$ indices $i \in E$.*

*Moreover, the algorithm runs in time $O((|P| + |E|)^3 \cdot \delta^{-2} \cdot \log(|P||E|/\delta))$.*

Note that we can take $\delta = 1/|E|$ and then round all those values to incur an additional constant of 1 in all the discrepancies. We can set the thresholds $c_v = 0$ for slightly less than $|E|/16$ of the vertices, then let the rest be $c_v = K$ for some large constant $K$ to satisfy $\sum_{v \in P} \exp(-c_v^2/16) \leq |E|/16$. Comparatively, Theorem 6.3 only allows for one partition of vertices $P_i$ to have discrepancy 0, i.e. $u = (0, u_2, \ldots, u_d)$ as long as $\sum_{l=2}^{d} 1/(u_l + 2) \leq 1/2$. This approach is generally better as the vertex partition $P_i$ of size greater than $|E|/16$. It does not allow one to control the discrepancy along each vertex individually, while the Lovett-Meka approach does.

To find such a solution, Lovett and Meka used a linear algebraic argument called an "Edge-Walk". We have variable constraints $|x_i| \leq 1$ and discrepancy constraints $|\sum_{e \in \delta(v)} (x_e - x_e^*)| \leq c_v \sqrt{|\delta(v)|}$ for all $v \in P$. The space of all points $x \in \mathbb{R}^E$ which satisfy this forms a polytope $\mathcal{P}$. A point $x \in \mathcal{P}$ as far as possible from the origin will likely have many coordinates close to 1. In fact, the partial coloring lemma states that we can always find a point $x \in \mathcal{P}$ where at least half of the coordinates are integral. To do this, they perform a constrained random walk via discrete Gaussian steps that sticks to "edges". For example in a 3D polytope, the random walk will stick to the first 2D polytope (polygon) it reaches, then stick to first 1D polytope (edge) it reaches, and finally end near a 0D polytope (vertex).

## 7.3 Deviations by Vertex

Our main result is the following:

**Theorem 7.3.** *Let $G$ be a $d$-partite hypergraph with vertex partition $V = \{P_1, \ldots P_d\}$ and hyperedges $E$, and let $x \in [0,1]^E$ be such that $\sum_{e \in \delta(v)} x_e$ is integral for every vertex $v \in V$, where $\delta(v)$ is the set of hyperedges containing $v$. Also let $u_v$ be nonnegative integers for every $l \in [d]$ and every $v \in P_l$ such that:*

$$\sum_{\ell=1}^{d} \frac{|P_l|}{\sum_{v \in P_l} (u_v + 2)} \leq 1$$

*Then there exists $z \in \{0,1\}^E$ such that for every $\ell \in [d]$ and every $v \in P_\ell$ it holds that $|\sum_{e \in \delta(v)} (z_e - x_e)| \leq u_v$, and additionally $z_e = x_e$ when $x_e$ is integer. The calculation of $z$ can be performed in time polynomial in $|E|$ and $|V|$.*

We achieve this with Algorithm 2, a slight modification of Algorithm 1 where it takes in deviations $u_v$ for every $l \in [d]$ and every $v \in P_l$ instead of uniform deviations $u_l$ across each dimension $l \in [d]$. We then replace each instance of $u_l$ with $u_v$.

To show Theorem 7.3, we need to show that this Algorithm 2 terminates if the following condition is satisfied by the deviations $u_v$:

$$\sum_{\ell=1}^{d} \frac{|P_l|}{\sum_{v \in P_l} (u_v + 2)} \leq 1$$

The rest of the proof remains identical to the proof of Theorem 6.3 in the work by Cembrano et. al.

**Algorithm 2** Vertex Rounding Algorithm

**Require:** A $d$-partite hypergraph $G$ with vertex partition $\{P_1, ..., P_d\}$ and hyperedges $E$, a vector $x \in [0,1]^E$ and nonnegative integer values $u_v$ for every $l \in [d]$ and every $v \in P_l$.
**Ensure:** Binary vector $z \in \{0,1\}^E$.
Initialize $y^0 \leftarrow x$
Let $\mathcal{E}^0 = \{e \in E : y_e^0 \text{ is fractional}\}$
For each $\ell \in 1, ..., d$, let $Q_\ell^0 = \{v \in P_\ell : \left|\delta(v) \cap \mathcal{E}^0\right| \geq u_v + 2\}$
Let $z_e = y_e^0$ for every $e \notin \mathcal{E}^0$.
Initialize $t \leftarrow 0$
**while** there exists $\ell \in 1, ..., d$ such that $Q_\ell^t \neq \emptyset$ **do**
  Compute an extreme point $y^{t+1}$ of $\mathcal{K}(y^t, \mathcal{E}^t, Q^t)$
  Let $\mathcal{E}^{t+1} = \{e \in E : y_e^{t+1} \text{ is fractional}\}$
  For each $\ell \in 1, ..., d$, let $Q_\ell^{t+1} = \{v \in P_\ell : |\delta(v) \cup \mathcal{E}^{t+1}| \geq u_v + 2\}$
  Let $z_e = y_e^{t+1}$ for every $e \in \mathcal{E}^t \setminus \mathcal{E}^{t+1}$. Update $t \leftarrow t+1$
**end while**
Let $T$ be the value of $t$ that did not satisfy the loop condition
Let $z_e \in \{\lfloor y_e^T \rfloor, \lceil y_e^T \rceil\}$ for every $e \in \mathcal{E}^T$
Return $z$.

**Lemma 7.4.** *Given an integer $t$, if there exists $l \in \{1, \ldots, d\}$ for which $Q_l^t \neq \emptyset$ and if $\sum_{\ell=1}^d \frac{|P_l|}{\sum_{v \in P_l}(u_v+2)} \leq 1$, we have that every extreme point of $\mathcal{K}(y^t, \mathcal{E}^t, Q^t)$ has at least one integral entry.*

*Proof.* We have that at any iteration $t$ and $l \in [d]$,

$$|Q_l^t| \leq \left\lfloor \frac{|\mathcal{E}^t|}{\sum_{v \in P_l}(u_v+2)/|P_l|} \right\rfloor$$

This step follows by a proof by contradiction:

$$\begin{aligned}
\sum_{v \in Q_l^t} |\delta(v) \cap \mathcal{E}^t| &\geq \sum_{v \in Q_l^t} (u_v + 2) \\
&\geq |Q_l^t| \sum_{v \in Q_l^t} \frac{u_v + 2}{|Q_l^t|} \\
&\geq \left( \sum_{v \in Q_l^t} \frac{u_v + 2}{|Q_l^t|} \right) \left( 1 + \left\lfloor \frac{|\mathcal{E}^t|}{\sum_{v \in P_l}(u_v+2)/|P_l|} \right\rfloor \right) \\
&> |\mathcal{E}^t|
\end{aligned}$$

And this is a contradiction, since each hyperedge in $\mathcal{E}^t$ contains at most one vertex $v \in Q_l^t$, so the LHS sums over disjoint subsets of $\mathcal{E}^t$. Note that we take advantage of the specific structure of the $d$-partite hypergraph here.

With this, we can now show that there are more variables (8) than equality constraints (7) in the LP at each time step $t$, i.e. $|\cup_{l=1}^d Q_l^t| < |\mathcal{E}^t|$.

$$\left| \bigcup_{l=1}^d Q_l^t \right| \leq \sum_{l=1}^d |Q_l^t| \leq \sum_{l=1}^d \left\lfloor \frac{|\mathcal{E}^t|}{\sum_{v \in P_l}(u_v+2)/|P_l|} \right\rfloor \leq |\mathcal{E}^t|$$

For the last inequality, we use the following lemma from Cembrano et. al., except we generalize $w_1, \ldots, w_d$ to be rational numbers rather than integer numbers:

**Lemma 7.5.** *Let $w_1, dotsc, w_d$ be rational numbers such that $w_l \geq 2$ for every $l \in [d]$. Then the following holds:*

*(i)* $q \geq \sum_{l=1}^{d} \lfloor q/w_l \rfloor$ *for every strictly positive* $q$ *iff* $\sum_{l=1}^{d} 1/w_l \leq 1$.

*(ii)* $q > \sum_{l=1}^{d} \lfloor q/w_l \rfloor$ *for every strictly positive* $q$ *iff* $\sum_{l=1}^{d} 1/w_l < 1$.

*(iii) Let $\bar{q}$ be a strictly positive integer such that $\bar{q} = \sum_{l=1}^{d} \lfloor \bar{q}/w_l \rfloor$. Then, $\bar{q}$ is a common multiple of the rational numbers $w_1, \ldots, w_d$.*

We can use the lemma stated above with our condition $\sum_{\ell=1}^{d} \frac{|P_l|}{\sum_{v \in P_l}(u_v+2)} \leq 1$ to show the last step in our inequality. Note that in this case, we have $w_l = \sum_{v \in P_l}(u_v + 2)/|P_l|$ for $l \in [d]$. Hence, this ensures that at any extreme point $y$ of the LP, at least one hyperedge will have integral value at each time step $t$ due to equality in one of the variable constraints.

□

This Lemma implies that Algorithm 1 terminates. Specifically, $\mathcal{E}^{t+1}$ is strictly contained in $\mathcal{E}^t$ since there is at least one hyperedge with integral value at each time step $t$. If the final time step in the loop is $T$ then we have that $T \leq |E|$.

Now that we can control the violations along each vertex rather than along each dimension, we can compare this to the direct application of Lovett-Meka to the hypergraph problem. There we could only set violations to be exactly 0 for around $|E|/16$ of the vertices, with the other violations incurring some large constant $K$. With this result, we can actually set violations to be 0 in all but one vertex in every dimension.

# 8 Concluding Remarks

In this thesis, we further explore the realm of approximate multi-dimensional apportionments. First, we outlined the theory that has led to divisor methods being the predominate type of apportionment methods used in practice. Then we explore the 1-D and 2-D apportionment problems, where an optimization approach becomes very valuable. This allows one to understand apportionment in 3 or more dimensions, where results that were previously true in 2-D and 1-D no longer hold as apportionments may not always be found. We slightly generalize the main theorem in Cembrano et al. [9] by allowing one to control the deviations by each individual vertex (a category within a dimension) rather than only along each dimension. This allows for greater flexibility when finding suitable apportionments in practice. For example, for the dimension of political parties there may be many small, independent parties able to get one seat. In this case, it is clear that the same tolerance for deviations should not be applied across the board to all parties, when a deviation of three seats might mean little to the largest party but it could effectively triple the seats for the smallest party. We refer to an example used in Cembrano et al. of the 2021 Chilean Constitutional Convention. The (party) list XA received almost 4% of the total votes yet obtained 0 seats in practice, whereas their method gave it 5 seats. This example makes it clear that the method used in practice may have over-represented the larger lists, however one should take care not to over-represent the smaller lists in response.

We also formally showed that the condition on which deviation vectors admit a proportional apportionment can be further improved, though informally this is not all too surprising. Theorem 6.3 makes use of ideas from the Beck-Fiala Theorem, but it is conjectured that the $O(t)$ discrepancy it obtains can be improved to $O(\sqrt{t})$. We specifically make use of the Bukh's bound, the best discrepancy result with only dependence on the sparsity t, to show that Theorem 6.3's conditions are necessary but not sufficient. On the other hand, we were unsuccessful in adapting the methods of Lovett-Meka and Bansal which improve the discrepancy if dependence on another parameter is allowed. These approaches can be applied in a black-box manner to the hypergraph problem and immediately give one control along each vertex. However, the black-box applications do not improve the approximation as they do not take advantage of the specific structure of apportionment. We hope that future work can successfully synthesize these works with the multidimensional apportionment problem.

# 9 Part 2: Adaptive Vector Bin Packing with Overflow

## 9.1 Introduction

This work extends the problem considered in "Adaptive Bin Packing with Overflow" by S. Perez-Salazar, M. Singh, and A. Toriello to multiple dimensions [17]. The classic problem of online bin packing has many applications, examples include cargo shipping and cloud computing. In this classic problem, $n$ items with sizes between $[0, 1]$ arrive one-by-one and are packed into bins of capacity 1, with the goal of minimizing the total number of bins used. The above paper considers the case where the size of the items are only known by a probability distribution and their actual size is only observed after being packed. This uncertainty is realistic in many cases, for the cloud computing example one must assign virtual machines (VM) to a server but the resources used by this VM is unknown until it is actually assigned and running on the server. As the actual size of an item is not known until it is packed, it is possible that a bin goes over its capacity (it is overflowed), in which a constant penalty is incurred. Thus the paper considers the problem of minimizing the total cost calculated as the sum of the number of opened bins and the overflow penalty cost. This work considers this same exact problem, but the items and the bins may have dimension greater than 1 and incur an overflow if they exceed the capacity in any dimension. In practice, a server has more than one resource in use, which include the CPU, RAM, I/O bandwidth, energy, etc. [16] Considering this problem in multiple dimensions is thus natural, and has been done in many other formulations of the classic bin packing problem.

We are specifically extending this work to $D$ dimensions via vector bin packing. Because items and bins now have multiple dimensions, we incorporate that into the notation of the original paper. The distribution of item $i$ in dimension $d$ is $X_{i,d}$, and its observed outcome is $X_{i,d} = x_{i,d}$. We let the usage of bin $j$ in dimension $d$ at the beginning of round $i$ be $S_{j,d}^{i-1}$, which is the sum of all item vectors packed into bin $j$ in the first $i-1$ rounds. If the usage of a bin exceeds 1 in any of its $D$ dimensions, then it will be overflowed, which means it can no longer be used and incurs a penalty constant $C$. Formally, a bin $j$ will be overflowed in the event that $\bigcup_{d=1}^{D} X_{i,d} + S_{j,d}^{i-1} > 1$. Due to the stochastic nature of this problem, we aim to minimize the cost of an algorithm $\mathcal{P}$ in expectation. Let $N_{\mathcal{P}}$ be the number of bins opened and $O_{\mathcal{P}}$ be the number of bins overflowed by the algorithm $\mathcal{P}$, then $\text{cost}(\mathcal{P}) = \mathbf{E}[N_{\mathcal{P}}] + C \cdot \mathbf{E}[O_{\mathcal{P}}]$.

In this report, the main result is that the *natural extension* of Budgeted Greedy algorithm maintains the same approximation ratio in $D$ dimensions as it did in the 1-D case when the items are i.i.d. The stated result is identical except for the fact that items $X_i$ can now be random vectors.

**Theorem 9.1.** *If the input sequence of random vectors $X_1, X_2, \ldots, X_n$ is i.i.d., Budgeted Greedy guarantees that cost$(ALG) \leq (3 + 2\sqrt{2})$ cost$(OPT)$, where $OPT$ denotes the optimal policy that knows $n$ in advance.*

What follows is a section on related work, a section describing the Budgeted Greedy algorithm, and then three sections dedicated to the proof of Theorem 9.1. In the first proof section, we bound our objective to only be in terms of $\mathbf{E}[N_{\mathcal{P}}]$ rather than the two parameters $\mathbf{E}[N_{\mathcal{P}}]$ and $\mathbf{E}[O_{\mathcal{P}}]$. The balance between opening a new bin versus overflowing an existing bin is controlled by the Budgeted Greedy algorithm according to the risk budget. In the second section, we can transform any arbitrary policy into a budgeted policy, i.e. it does not exceed the risk budget of any bin, by incurring a small multiplicative cost. Finally in the third section, we show that for i.i.d inputs, the Budgeted Greedy algorithm minimizes the number of bins opened $\mathbf{E}[N_{\mathcal{P}}]$ within the space of budgeted algorithms. Altogether we can take the OPT policy for i.i.d. inputs, transform it into a budgeted policy OPT$'$, then the Budgeted Greedy policy $BG$ will obtain $\mathbf{E}[N_{BG}]$ which is less than $\mathbf{E}[N_{\text{OPT}'}]$. This guarantees that the cost of Budgeted Greedy is bounded by an approximation ratio of the optimal cost.

# 10 Related Works

For a survey on approximation and online algorithms for multi-dimensional bin packing, see the following [10]. For the field of adaptive combinatorial optimization, see the following for one of the earliest works [11].

# 11 The Budgeted Greedy Algorithm

In this section, we formally define the Budgeted Greedy algorithm for Vector Bin Packing. Our set $I$ keeps track of the bins that are opened. We then keep track of the usage and accumulated risk of all bins $j \in I$ at round $i$ by $S_j^i$ and $r_j^i$ respectively. The usage of the bin will be the sum of the observed values of all item vectors packed into it, while the risk of the bin will be the sum of the probabilities that an item overflows the bin. For simplicity, we will call $O_{i\to j}$ the event of overflowing when placing item $i$ into bin $j$. As this is vector bin packing, we have that $O_{i\to j} := \bigcup_{d=1}^{D} X_{i,d} + S_{j,d}^{i-1} > 1$. The probability of overflowing is defined as $p_i(S_j^{i-1}) = P(O_{i\to j})$. We then have that

$$risk(B_j) = \sum_{i=1}^{n} P(O_{i\to j})\, \mathbf{1}_{i\to j}\,.$$

The algorithm is now described below.

**Algorithm 3** BUDGETED-GREEDY$(\gamma, X_1, \ldots, X_n)$

Initialize: $I = \emptyset$

**for** $i = 1, \ldots, n$ **do** $j \in I$ such that $r_j^{i-1} + p_i(S_j^{i-1}) \le \gamma/C$ $\;S_j^i = S_j^{i-1} + X_i$
$r_j^i = r_j^{i-1} + p_i(S_j^{i-1})$
Define $r_j^i = p_i(0)$ for j such that $j = \inf\{j \ge 0 : j \notin I\}$ $\;S_j^i = X_i$ Update $I = I \cup \{j\}$
  **for** $j' \ne j$ **do** $S_{j'}^i = S_{j'}^{i-1}$ $\;r_{j'}^i = r_{j'}^{i-1}$

## 11.1 Total cost bounded by total number of opened bins

What follows now is the analysis of the Budgeted Greedy algorithm, in order to provide an approximation ratio. First, the objective we want to minimize is $\mathbf{E}[N_{\mathcal{P}}] + C \cdot \mathbf{E}[O_{\mathcal{P}}]$. However, the following lemma will allow us to bound this objective in terms of just one parameter $N_{\mathcal{P}}$ when we use a budgeted policy. The notion of risk budget guarantees that we only overflow a bounded amount of bins in expectation.

**Lemma 11.1.** *Let $X_1, X_2, \ldots, X_n$ be nonnegative i.i.d. random vectors with dimension $D$, and let policy $\mathcal{P}$ be any policy that sequentially packs these items. The expected number of bins overflowed by the policy $\mathcal{P}$ is*

$$\mathbf{E}[O_{\mathcal{P}}] = \sum_{j=1}^{n} \mathop{\mathbf{E}}_{X_1,\ldots,X_n}\left[\sum_{i=1}^{n} P_{X_i}(O_{i\to j})\, \mathbf{1}_{\{i\to j\}}^{\mathcal{P}}\right]$$

**Proof:** The proof is similar to that of Proposition 3.1 in the original paper. The only difference is the notion of overflowing the bin has changed, as we have multiple dimensions now. Note that exceeding the capacity in any dimension of the bin will incur the overflow penalty and stop it from being used by future items. We first find the value of $P(\mathcal{P}$ breaks bin j). By linearity of expectation,

$$\mathbf{E}\left[\sum_{i=1}^{n} P_{X_i}(O_{i\to j})\, \mathbf{1}_{\{i\to j\}}^{\mathcal{P}}\right] = \sum_{i=1}^{n} \mathbf{E}\left[P_{X_i}(O_{i\to j})\, \mathbf{1}_{\{i\to j\}}^{\mathcal{P}}\right]$$

Now the usage of a bin $S_{j,d}^{i-1} = \sum_{i=1}^{i-1} X_k\, \mathbf{1}_{\{i\to j}^{\mathcal{P}}$ only depends on the outcomes of $X_1, \ldots, X_{i-1}$. Then

$$\begin{aligned}
\mathbf{E}\left[\mathop{P}_{X_i}(O_{i\to j})\, \mathbf{1}_{\{i\to j\}}^{\mathcal{P}}\right] &= \mathop{\mathbf{E}}_{X_1,\ldots,X_{i-1}}\left[\mathop{P}_{X_i}(O_{i\to j})\, \mathbf{1}_{\{i\to j\}}^{\mathcal{P}}\right] \\
&= \mathop{\mathbf{E}}_{X_1,\ldots,X_{i-1}}\left[\mathop{\mathbf{E}}_{X_i}\left[\mathbf{1}_{\{O_{i\to j}\}}\right] \mathbf{1}_{\{i\to j\}}^{\mathcal{P}}\right] \\
&= \mathop{\mathbf{E}}_{X_1,\ldots,X_i}\left[\mathbf{1}_{\{O_{i\to j}\}}\, \mathbf{1}_{\{i\to j\}}^{\mathcal{P}}\right]
\end{aligned}$$

The event $\{O_{i\to j}, i \to j\}$ is exactly when $X_i$ *breaks bin* $j$. It occurs when we pack item $i$ into bin $j$ and it overflows, i.e. $\bigcup_{d=1}^{D} X_{i,d} + S_{j,d}^{i-1} > 1$ is true. Thus,

$$\mathbf{E}\left[\underset{X_i}{P}(O_{i\to j})\,\mathbf{1}^{\mathcal{P}}_{\{i\to j\}}\right] = P(X_i \text{ breaks bin } j)$$

And so,

$$\mathbf{E}\left[\sum_{i=1}^{n} P_{X_i}(O_{i\to j})\,\mathbf{1}^{\mathcal{P}}_{\{i\to j\}}\right] = \sum_{i=1}^{n} P(X_i \text{ breaks bin } j) = P(\mathcal{P} \text{ breaks bin } j)$$

□

Now we can use lemma 4.1 to say that

**Lemma 11.2.** *Let $\gamma \geq 1$ and assume that for all $i$, $P(\bigcup_{d=1}^{D} X_{i,d} > 1) \leq \gamma/C$. For any bin $j$, budgeted algorithms guarantee*

$$P(ALG \text{ breaks bin } j) \leq \frac{\gamma}{C} P(ALG \text{ opens bin } j)$$

**Proof:** We use Lemma 11.1 to get

$$\begin{aligned} P(\text{ALG breaks bin } j) &= \mathbf{E}\left[\left(\sum_{i=1}^{n} P_{X_i}(O_{i\to j})\,\mathbf{1}^{\mathcal{P}}_{\{i\to j\}}\right)\mathbf{1}_{\{\text{ALG opens bin } j\}}\right] \\ &= \mathbf{E}\left[Risk(B_j)\,\mathbf{1}_{\{\text{ALG opens bin } j\}}\right] \\ &\leq \frac{\gamma}{C}\cdot P(\text{ALG opens bin } j) \end{aligned}$$

Now with Lemma 4.2, we can say that

$$\text{cost}(\text{ALG}) \leq (1+\gamma)\,\mathbf{E}[N_{\text{ALG}}]$$

## 11.2 Policy-Tree Analysis Transformation

In this section, we will convert any given policy into a budgeted policy at the cost of a small multiplicative factor.

Formally, the policy $\mathcal{P}$ is represented by a policy tree $\mathcal{T}_{\mathcal{P}}(u)$ with root node $u$ which maps out all the possible values of the items. First, we modify the policy so that it only exceeds the risk of each bin at most once. Then, we simply place the (one) item that does surpass the risk in each bin into its own bin. This converts the arbitrary policy into a budgeted policy, where each bin never surpasses the risk budget of $\gamma/C$.

For an arbitrary policy $\mathcal{P}$ rooted at node $u$, the cost is $\text{cost}_{l,c}(\mathcal{T}_{\mathcal{P}}(u))$. We then augment the cost of overflow from $C$ to $C+2\delta$, which modifies to cost to $\text{cost}_{l,\hat{c}}(\mathcal{T}_{\mathcal{P}}(u))$. We want to transform this policy $\mathcal{P}$ into a budgeted policy $\mathcal{P}'$ with cost represented by $\text{cost}_{l',c'}(\mathcal{T}_{\mathcal{P}'}(u))$. Now the brunt of the proof is showing that we can move around the extra $2\delta$ from augmenting the cost to pay for the new bin opened in the transformed budgeted policy, which would not increase the cost any further. This is done using the lemma below.

**Lemma 11.3.** $cost_{l,\hat{c}}(\mathcal{T}_{\mathcal{P}}(u)) \geq cost_{l',c'}(\mathcal{T}_{\mathcal{P}'}(u))$

**Proof:** One iteration of the transformation will only modify the policy with respect to bin $j$, so we find node $u$ which first opens bin $j$ then root our policy tree at that node. Now we will define the cost of just bin $j$ in policy tree $\mathcal{T}_{\mathcal{P}}(u)$ as follows:

$$\text{cost}_{l,\hat{c}}(\mathcal{T}_{\mathcal{P}}(u))_j = 1 + \mathbf{E}\left[(C+2\delta)\sum_{i=1}^{n} \mathbf{1}^{\mathcal{P}}_{i\to j}\,\mathbf{1}^{\mathcal{P}}_{O_{i\to j}} \mid \text{Reach node } u\right]$$

We want to compare this to the cost of bin $j$ and bin $j'$ in our new policy $\mathcal{P}'$, which we call $\text{cost}_{l',c'}(\mathcal{T}_{\mathcal{P}'}(u))_j$. Based on our transformation, the cost of overflowing bin $j$ is $C+\delta$, while the cost of overflowing bin $j'$ is $C+2\delta$. Thus, we can calculate it as follows:

$$\text{cost}_{l',c'}(\mathcal{T}_{\mathcal{P}'}(u))_j = 1 + \mathbf{E}\left[\mathbf{1}^{\mathcal{P}'}_{\text{Open bin } j'} + (C+\delta)\sum_{i=1}^{n}\mathbf{1}^{\mathcal{P}'}_{i\to j}\,\mathbf{1}^{\mathcal{P}'}_{O_{i\to j}} + (C+2\delta)\sum_{i=1}^{n}\mathbf{1}^{\mathcal{P}'}_{i\to j'}\,\mathbf{1}^{\mathcal{P}'}_{O_{i\to j'}} \mid \text{Reach node } u\right]$$

Now, we have that

$$\mathbf{1}^{\mathcal{P}}_{i\to j} = \mathbf{1}^{\mathcal{P}'}_{i\to j} + \mathbf{1}^{\mathcal{P}'}_{i\to j'}$$

This is true, because if policy $\mathcal{P}$ packs $i\to j$, then policy $\mathcal{P}'$ packs $i$ into either $j$ or $j'$. We also have that

$$\mathbf{1}^{\mathcal{P}}_{O_{i\to j}} \geq \mathbf{1}^{\mathcal{P}'}_{O_{i\to j}}$$
$$\mathbf{1}^{\mathcal{P}}_{O_{i\to j}} \geq \mathbf{1}^{\mathcal{P}'}_{O_{i\to j'}}$$

since the items packed by policy $\mathcal{P}$ into bin $j$ will be split between bin $j$ and $j'$ in policy $\mathcal{P}'$. Hence, bin $j$ in $\mathcal{P}$ will always be overflowed before. Now the difference between our two costs will be

$$\begin{aligned}\text{cost}_{l,\hat{c}}(\mathcal{T}_{\mathcal{P}}(u)) - \text{cost}_{l',c'}(\mathcal{T}_{\mathcal{P}'}(u)) &= \text{cost}_{l,\hat{c}}(\mathcal{T}_{\mathcal{P}}(u))_j - \text{cost}_{l',c'}(\mathcal{T}_{\mathcal{P}'}(u))_j \\ &\geq \mathbf{E}\left[-\mathbf{1}^{\mathcal{P}'}_{\text{Open bin } j'} + \delta\sum_{i=1}^{n}\mathbf{1}^{\mathcal{P}'}_{i\to j}\,\mathbf{1}^{\mathcal{P}'}_{O_{i\to j}} \mid \text{Reach node } u\right]\end{aligned}$$

So it remains to show that this value is non-negative. Specifically, this is exactly where we need the extra $\delta$ cost to cancel out the cost of opening a new bin $j'$. To do this, we first decouple the expectation from the condition that it must reach node $u$. For $i \geq j$,

$$\begin{aligned}\mathbf{E}\left[\mathbf{1}^{\mathcal{P}'}_{i\to j}\,\mathbf{1}^{\mathcal{P}'}_{O_{i\to j}} \mid \text{Reach node } u\right] &= \mathop{\mathbf{E}}_{X_1,\ldots,X_{i-1}}\left[\mathop{\mathbf{E}}_{X_i}\left[\mathbf{1}^{\mathcal{P}'}_{i\to j}\,\mathbf{1}^{\mathcal{P}'}_{O_{i\to j}}\right] \mid \text{Reach node } u\right] \\ &= \mathop{\mathbf{E}}_{X_1,\ldots,X_{i-1}}\left[\mathbf{1}^{\mathcal{P}'}_{i\to j}\mathop{\mathbf{E}}_{X_i}\left[\mathbf{1}^{\mathcal{P}'}_{O_{i\to j}}\right] \mid \text{Reach node } u\right] \\ &= \mathop{\mathbf{E}}_{X_1,\ldots,X_{i-1}}\left[\mathbf{1}^{\mathcal{P}'}_{i\to j}\,P_{X_i}(O_{i\to j}) \mid \text{Reach node } u\right]\end{aligned}$$

For $i \leq j-1$, we have

$$\mathbf{E}\left[\mathbf{1}^{\mathcal{P}'}_{i\to j}\,\mathbf{1}^{\mathcal{P}'}_{O_{i\to j}} \mid \text{Reach node } u\right] = 0$$

Thus,

$$\begin{aligned}\text{cost}_{l,\hat{c}}(\mathcal{T}_{\mathcal{P}}(u)) - \text{cost}_{l',c'}(\mathcal{T}_{\mathcal{P}'}(u)) &\geq \mathbf{E}\left[-\mathbf{1}^{\mathcal{P}'}_{\text{Open bin } j'} + \delta\sum_{i=1}^{n}\mathbf{1}^{\mathcal{P}'}_{i\to j}\,\mathbf{1}^{\mathcal{P}'}_{O_{i\to j}} \mid \text{Reach node } u\right] \\ &= \mathbf{E}\left[-\mathbf{1}^{\mathcal{P}'}_{\text{Open bin } j'} + \delta\cdot\text{Risk}(B_j) \mid \text{Reach node } u\right] \\ &\geq \mathbf{E}\left[\delta\frac{\gamma}{C} - 1 \mid \text{Reach node } u\right] \\ &= 0\end{aligned}$$

□

With this lemma, we can iteratively apply the policy tree transformation to every bin without modifying the cost, resulting in a policy where the risk budget is only surpassed at most once. Once we have that condition, we apply a similar argument, opening a new bin for each item that surpasses the risk budget, and shifting the cost from $C+\delta$ to $C$ to do that.

## 11.3 I.I.D: BG opens minimal bins of all budgeted policies

Here note that our items are required to be i.i.d. with each other, however the joint distributions of the random vectors themselves do not necessarily have to be. The idea is that our items are i.i.d, so being greedy will open the minimum number of boxes within the space of budgeted policies. The order in which items are placed does not matter in expectation if the distributions are identical and independent. The proof outline uses this idea to constantly swap items until the policy becomes Budgeted Greedy.

**Lemma 11.4.** *Suppose $X_1, X_2, \ldots, X_n$ are nonnegative i.i.d. random vectors. Then,*

$$\mathbf{E}[N_{ALG}] = \min_{\substack{\mathcal{P} \text{ budgeted with} \\ \text{risk budget } \gamma/C}} \mathbf{E}[N_{\mathcal{P}}]$$

**Proof:** The goal is to transform any budgeted policy $\mathcal{P}$ with risk budget $\gamma/C$ into a Budgeted Greedy policy $\mathcal{P}_{BG}$ without adding any bins. Let $\mathcal{T}_{\mathcal{P}}$ be the policy tree representation of $\mathcal{P}$. In terms of bin usage, Budgeted Greedy only uses one bin at a time for i.i.d. inputs, so our resulting policy tree must also satisfy that property. In a policy tree, this property means that for any branch going from the root to a leaf we have labels of the form $1 \to j_1, 2 \to j_2, \ldots, n \to j_n$ such that $j_1 \leq j_2 \leq \ldots \leq j_n$. Now we state a lemma that formalizes the notion of "swapping" items between bins.

**Lemma 11.5.** *Let $j = 1, \ldots, n$ be any bin opened by the policy $\mathcal{P}$. Suppose that node $u$ in level $k$ is labeled $k \to j'$, where $j' \neq j$. Furthermore, suppose that at node $u$, bin $j$ is open, its usage does not exceed 1, and its risk budget can accommodate $k$. Then $u$ can be relabeled $k \to j$ without increasing the expected number of bins opened by the policy.*

We proceed with backward induction on the level $k$ of node $u$ in the policy tree. For the base case, we have that node $u$ is on level $n$ with label $n \to j'$, $j' \neq j$. Assuming that bin $j$ satisfies our conditions, then we can relabel node $u$ to $n \to j$ without increasing the expected number of bins because no bin was opened in this process. This may increase the overflow cost, but it will not increase the opened bin cost.

Now our inductive hypothesis assumes that the lemma is true for levels $k+1, k+2, \ldots, n$. Once again, we consider a node $u$ at level $k$ with label $k \to j'$, $j' \neq j$ and bin $j$ satisfying the conditions. We have two cases depending on the children of node $u$. If all its children are labeled $k+1 \to j$, then we can simply swap bins by relabeling node $u$ to $k \to j$ and all its children to $k+1 \to j'$. The i.i.d. condition is used here, as the distribution of items $X_k$ and $X_{k+1}$ must be the same in order to swap them without modifying the risk budget. Now the other case is where at least one children of node $u$, say node $v$, has label $k+1 \to m$ where $m \not\to j$. The state of bin $j$ at node $v$ is identical to its state at node $u$ (its usage is less than 1 and risk budget is less than $\gamma/C$), so we can apply the inductive hypothesis here. That is, we can relabel $v$ with $k+1 \to j$ and repeat this with all other children of node $u$ with labels $k+1 \to m$, $m \neq j$. The end result brings us to the first case, where we can swap bins. This completes the induction proof for the lemma. □

Now with Lemma 11.5, we can start at the root $r$ and iteratively apply it to bins $1, 2, \ldots, n$ in order. The process will stop for bin 1 once any branch from the root has the form $1 \to 1, 2 \to 1, \ldots, i \to 1, i+1 \to 2, \ldots$ and bin 1 is overflowed or exceeds its risk budget. The process is then applied to bin $2, 3$, and so on with the end result being that any branch from the root to a leaf has the form $1 \to j_1, 2 \to j_2, \ldots, n \to j_n$ such that $j_1 \leq j_2 \leq \ldots \leq j_n$. The resulting policy tree only opens one bin at a time, acting like Budgeted Greedy without increasing the cost at any step. Thus, Budgeted Greedy must always open the minimum number of expected bins over the space of policies with risk budget $\gamma/C$. □

All together, the preceding three sections yield the proof for the main theorem:

**Theorem 11.6.** *If the input sequence of random vectors $X_1, X_2, \ldots, X_n$ is i.i.d., Budgeted Greedy guarantees that cost$(ALG) \leq (3 + 2\sqrt{2})$ cost$(OPT)$, where $OPT$ denotes the optimal policy that knows $n$ in advance.*

**Proof:** We have that

$$\text{cost}(\mathcal{P}) \leq (1+\gamma)\,\mathbf{E}[N_{\mathcal{P}}] \leq (1+\gamma)\,\text{cost}(OPT*) \leq (1+\gamma)\left(1 + \frac{2}{\gamma}\right)\text{cost}(OPT)$$

where $OPT*$ is the budgeted version of OPT per the policy tree transformation. Then the coefficient is minimized at $\gamma = \sqrt{2}$, yielding the coefficient $3 + 2\sqrt{2}$.

# 12 Conclusion

In this work, we extended the Adaptive Bin Packing Problem from the 1-dimensional case to the vector bin packing case, for i.i.d. input sequences. We extended naturally Budgeted Greedy from the 1-dimensional version to the vector version with the notion of joint risk. The proof follows the same structure as the 1-dimensional case, and the guarantees obtained are the same constant $3 + 2\sqrt{2}$. This result is surprising as increasing the dimension worsens the approximation guarantees possible in the standard bin packing problems. Possibly this is due to the i.i.d. assumption. Although, for non-identical distributions, we cannot guarantee an approximation of the optimal offline cost independent of the dimension $D$. This is because we can solve the vector bin packing problem (e.g., by considering a large penalty $C$ and deterministic items).

We believe that $3 + 2\sqrt{2}$ is not the best approximation that we can achieve for the i.i.d. case, and improvements during the reduction to budgeted policies can save some approximation factors. Indeed, the reduction used is very general and works for any distributions. Furthermore, we expect that using the i.i.d. assumption in this reduction decreases the multiplicative loss.

Another interesting modeling extension of the 1-dimensional version is the multidimensional packing version, where we can choose the place in the space to place the incoming item. We believe that similar reasoning and extensions as presented in this report can solve this case.

# References


[1] Zeyuan Allen-Zhu, Yuanzhi Li, Rafael Oliveira, and Avi Wigderson. Much faster algorithms for matrix scaling. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 890–901, 2017.

[2] M. L. Balinski and G. Demange. Algorithms for proportional matrices in reals and integers. *Mathematical Programming*, 45(1):193–210, Aug 1989.

[3] M. L. Balinski and G. Demange. An axiomatic approach to proportionality between matrices. *Mathematics of Operations Research*, 14(4):700–719, Nov 1989.

[4] M. L. Balinski and H. P. Young. The webster method of apportionment. *Proceedings of the National Academy of Sciences*, 77(1):1–4, 1980.

[5] M. L. Balinski and H. P. Young. *Fair Representation: Meeting the Ideal of One Man, One Vote*. Brookings Institution Press, 2001.

[6] Nikhil Bansal, Daniel Dadush, and Shashwat Garg. An algorithm for komlós conjecture matching banaszczyk's bound. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 788–799, 2016.

[7] József Beck and Tibor Fiala. "integer-making" theorems. *Discrete Applied Mathematics*, 3(1):1–8, 1981.

[8] Boris Bukh. An improvement of the beck–fiala theorem. *Combinatorics, Probability and Computing*, 25(3):380–398, 2016.

[9] Javier Cembrano, José Correa, and Victor Verdugo. Multidimensional political apportionment. *Proceedings of the National Academy of Sciences*, 119(15):e2109305119, 2022.

[10] Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79, 2017.

[11] B.C. Dean, M.X. Goemans, and J. Vondrdk. Approximating the stochastic knapsack problem: the benefit of adaptivity. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 208–217, 2004.

[12] Pukelsheim F. Gaffke, N. Vector and matrix apportionment problems and separable convex integer optimization. *Mathematical Methods in Operations Research*, 67:133–159, 2008.

[13] Edward V. Huntington. The mathematical theory of the apportionment of representatives. *Proceedings of the National Academy of Sciences of the United States of America*, 7(4):123–127, 1921.

[14] Martin Idel. A review of matrix scaling and sinkhorn's normal form for matrices and positive maps, 2016.

[15] Shachar Lovett and Raghu Meka. Constructive discrepancy minimization by walking on the edges. *SIAM Journal on Computing*, 44(5):1573–1582, 2015.

[16] Rina Panigrahy, Kunal Talwar, Lincoln Uyeda, and Udi Wieder. Heuristics for vector bin packing. January 2011.

[17] Sebastian Perez-Salazar, Mohit Singh, and Alejandro Toriello. Adaptive bin packing with overflow. *Mathematics of Operations Research*, 2022.

[18] Friedrich Pukelsheim. *Proportional Representation*. Springer Cham, 2014.

[19] Günter Rote and Martin Zachariasen. Matrix scaling by network flow. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 848–854, USA, 2007. Society for Industrial and Applied Mathematics.

[20] Uriel G. Rothblum and Hans Schneider. Scalings of matrices which have prespecified row sums and column sums via optimization. *Linear Algebra and its Applications*, 114-115:737–764, 1989.

[21] Richard Sinkhorn. A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices. *The Annals of Mathematical Statistics*, 35(2):876 – 879, 1964.