# An Improved Protocol for ExactlyN with More Than 3 Players

**Lianna Hambardzumyan**
The Hebrew University of Jerusalem, Israel

**Toniann Pitassi**
Columbia University, New York, NY, USA

**Suhail Sherif**
LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal

**Morgan Shirley**
University of Toronto, Canada

**Adi Shraibman**
The Academic College of Tel Aviv-Yaffo, Israel

## Abstract

The ExactlyN problem in the number-on-forehead (NOF) communication setting asks $k$ players, each of whom can see every input but their own, if the $k$ input numbers add up to $N$. Introduced by Chandra, Furst and Lipton in 1983, ExactlyN is important for its role in understanding the strength of randomness in communication complexity with many players. It is also tightly connected to the field of combinatorics: its $k$-party NOF communication complexity is related to the size of the largest corner-free subset in $[N]^{k-1}$.

In 2021, Linial and Shraibman gave more efficient protocols for ExactlyN for 3 players. As an immediate consequence, this also gave a new construction of larger corner-free subsets in $[N]^2$. Later that year Green gave a further refinement to their argument. These results represent the first improvements to the highest-order term for $k = 3$ since the famous work of Behrend in 1946. In this paper we give a corresponding improvement to the highest-order term for $k > 3$, the first since Rankin in 1961. That is, we give a more efficient protocol for ExactlyN as well as larger corner-free sets in higher dimensions.

Nearly all previous results in this line of research approached the problem from the combinatorics perspective, implicitly resulting in non-constructive protocols for ExactlyN. Approaching the problem from the communication complexity point of view and constructing explicit protocols for ExactlyN was key to the improvements in the $k = 3$ setting. As a further contribution we provide explicit protocols for ExactlyN for any number of players which serves as a base for our improvement.

**2012 ACM Subject Classification** Theory of computation → Communication complexity; Mathematics of computing → Combinatorics

**Keywords and phrases** Corner-free sets, number-on-forehead communication

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2024.58

**Related Version** *Full Version*: https://eccc.weizmann.ac.il/report/2023/138/

**Funding** *Lianna Hambardzumyan*: Research partially supported by ISF grant 921/22.
*Toniann Pitassi*: Supported by NSF AF:Medium 2212136.
*Suhail Sherif*: Funded by the European Union (ERC, HOFGA, 101041696). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Also supported by FCT through the LASIGE Research Unit, ref. UIDB/00408/2020 and ref. UIDP/00408/2020.
*Morgan Shirley*: Supported by an NSERC grant.

**Acknowledgements** Most of the work was done while Suhail Sherif was at Vector Institute, Toronto, Canada. We thank Zach Hunter for his helpful comments on an earlier version of the paper.





# 1 Introduction

In this paper we continue a recent line of work that seeks to apply ideas from *communication complexity* to the field of *additive combinatorics*. Specifically, we study the following problems:

**(i)** ***k*-AP Problem:** What is the maximum size of a subset of $[N]$ that contains no (nontrivial) *k-term arithmetic progression* ($k$-AP for short) – a sequence $x, x+\delta, x+2\delta, \ldots, x+(k-1)\delta$ for some $\delta \neq 0$?

**(ii)** **Corners Problem:** What is the maximum size of a subset of $[N]^k$ that contains no *k-dimensional corner* – a set of $k+1$ points of the form:

$$(x_1, x_2, \ldots, x_k), (x_1+\delta, x_2, \ldots, x_k), (x_1, x_2+\delta, \ldots, x_k), \ldots, (x_1, x_2, \ldots, x_k+\delta)$$

for some $\delta \neq 0$?

Our paper is inspired by a growing body of equivalences that have been discovered between problems in additive combinatorics and communication complexity. We build on recent work that exploits these equivalences to gain new perspectives on the two main problems above.

**(i)** The $k$-AP Problem is equivalent to the deterministic *number-in-hand* (NIH) $k$-player communication complexity of the following promise version of Equality: Each of the $k$ players is given an input $x_i \in [N]$ and they want to decide if their inputs are all equal under the promise that they form a $k$-term arithmetic progression.

**(ii)** The Corners Problem is equivalent to the $(k+1)$-player *number-on-forehead* (NOF) communication complexity of ExactlyN: There are $k+1$ inputs, $x_1, \ldots, x_{k+1} \in [N]$, where Player $i$ sees all inputs except for $x_i$, and they want to decide whether or not the sum of their inputs is equal to $N$.

The main contribution of this paper is a new protocol for the ExactlyN problem that is more efficient than previously-known protocols when there are more than three players. This in turn gives a new method for constructing corner-free subsets of $[N]^k$ which improves on previous constructions for $k > 2$.

## 1.1 Background

Computational complexity and additive/extremal combinatorics have enjoyed a rich interaction in the last fifty years. On one side, extremal combinatorics has been critical for proving complexity lower bounds. For example, the Sunflower Lemma underlies Razborov's superpolynomial monotone circuit lower bound [27] as well as recent query-to-communication lifting theorems [23], and Ramsey's Theorem underlies many complexity lower bounds [25]. On the other side, tools from complexity theory have been used to resolve problems in additive/extremal combinatorics. For example, the recent breakthrough on the Sunflower conjecture [2] uses ideas behind the Switching Lemma, and the resolution of the Kakeya conjecture [8] and the Cap-Set Conjecture [7, 10] use the polynomial method from circuit complexity. Moreover, some of the main achievements in theoretical computer science – advances in error correcting codes, the PCP theorem, and pseudorandomness/extractors – have rich and deep connections with additive combinatorics [22].

In this paper we continue in this tradition by studying two fundamental problems that are well-studied from both the lenses of additive combinatorics and communication complexity. We give a brief discussion of their importance and motivations from these respective fields.

**Additive combinatorics.** A basic question in number theory and additive combinatorics is understanding the existence of additive structure in the natural numbers, and understanding how much of this structure is algebraic or combinatorial in nature. A remarkable early theorem from 1927 due to Van der Waerden states that for every $r$ and $k$, there exists $N$ such that any $r$-coloring of the numbers in $[N]$ contains a monochromatic $k$-term arithmetic progression. Later it was famously shown that in fact any dense enough subset of the natural numbers contains an arbitrarily large arithmetic progression. Subsequently, many generalizations and quantative versions have received a lot of attention in Ramsey theory, with Szemeredi's Theorem and the Multidimensional Szemeredi's Theorem proving that the density of $k$-AP free sets and corner-free sets must be sub-constant. This has led to a lot of interest both in improving the density upper bounds and in finding large $k$-AP and corner-free sets. We refer the reader to the excellent books by Tao and Vu [30] and by Zhao [32] for a comprehensive treatment.

**Communication complexity.** The additive combinatorics problems we study here, viewed through the lens of communication complexity, are essentially questions about derandomization. The $k$-AP problem, reformulated as a communication problem, is a restriction of the Equality function, which in the NIH model is easy for randomized protocols but maximally hard for deterministic protocols. The restricted version here asks how the deterministic complexity changes under the assumption that the inputs have an additive structure.

ExactlyN (the Corners Problem) has also been studied for the purpose of showing a separation between randomized and deterministic NOF communication complexity. Although a strong non-constructive separation is known even for $k = n^\epsilon$ many players [3], it was only recently that the first constructive separation was shown [17], and even then it has only been proven for $k = 3$ players.

Even though a constructive separation is now known, ExactlyN continues to be of central importance in this line of research. This is because ExactlyN is a "graph function", and the strong non-constructive separation mentioned above [3] is witnessed by most graph functions. The separating function of [17] is surprisingly not a graph function, and their lower bound technique is not known to apply to ExactlyN. New techniques developed for lower bounding the complexity of ExactlyN would then be promising to provide lower bounds when $k > 3$. This would be of much interest since NOF lower bounds when $k > \log n$ would imply breakthrough **ACC** circuit lower bounds [5, 31]. On the other hand, it is entirely possible that there are efficient protocols for ExactlyN that are waiting to be discovered.

## 1.2 Previous bounds

The current state-of-the-art reveals a significant difference in our understanding of the $k$-AP problem and the Corners problem.

**The $k$-AP Problem.** A construction by Behrend from 1946 yields a 3-AP-free subset of $[N]$ of size at least $N \cdot 2^{-2\sqrt{2}\sqrt{\log N}+o(\sqrt{\log N})}$ [4].[1] The recent breakthrough result of Kelley and Meka [18] shows that the exponent is tight to within polynomial factors for $k = 3$.

Behrend's result was extended to all $k > 3$ by Rankin [26] who obtained the following subset size lower bound: $N \cdot 2^{-t2^{(t-1)/2}\cdot(\log N)^{1/t}+o((\log N)^{1/t})}$, for $t = \lceil \log k \rceil$. Note that this matches Behrend's result (it is the same construction) when $k = 3$. The best size upper bound for $k = 4$ is $N \cdot 1/(\log N)^{\Omega(1)}$ [14] and for $k > 4$ is $N \cdot 1/(\log \log N)^{\eta}$, where $\eta = 2^{-2^{k+9}}$ [11].

[1] All logarithms in this paper are base 2.

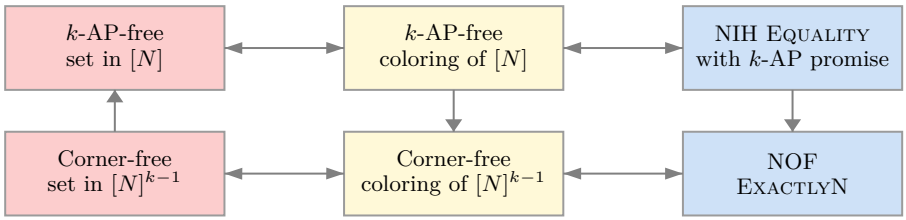


**Figure 1** The figure shows how the additive combinatorics problems are related to each other and to their communication complexity equivalents. For problems $A$ and $B$, $A \to B$ denotes $c(B) = O(c(A))$, where $c(\cdot)$ measures the problem's complexity in our context.

**The Corners Problem.** Until fairly recently, the best corner-free set construction was via a direct reduction to the $k$-AP Problem. Ajtai and Szemerédi first gave this reduction for $k = 3$ [1]; their proof easily generalizes to $k > 3$. The reduction is very clean and yields the same *density* lower bounds for the $(k-1)$-dimensional Corners Problem as for the $k$-AP Problem – if $[N]$ has a $k$-AP-free subset of size $N \cdot \delta$, then $[N]^{k-1}$ has a corner-free subset of size $N^{k-1} \cdot \delta$. In particular, the estimates of Behrend and Rankin can be directly applied to the Corners Problem.

Unlike the $k$-AP problem, where for $k = 3$ relatively tight bounds are known, there is a large gap between upper and lower bounds for the 2-dimensional Corners Problem. The best known upper bound is $N^2 \cdot 1/(\log\log N)^c$ for some constant $c$ by Shkredov [29]. For $k \geq 3$ the best upper bound is just $N^k \cdot o(1)$ [12].

Recent works have improved the Ajtai-Szemerédi reduction, yielding better lower bounds for the Corners Problem, by examining it through a communication complexity lens.

**Communication complexity and improved bounds for the Corners Problem.** In 1983, Chandra, Furst, and Lipton defined the NOF model of communication and showed the equivalence between the $k$-party NOF complexity of ExactlyN and the $(k-1)$-dimensional Corners Problem [6].

Specifically, the minimal cost of protocols for these problems is (up to a constant factor) the logarithm of the optimal solution for the closely-related *coloring* version of the additive combinatorics problems in question:

(i) ***k*-AP Problem (Coloring Version):** What is the minimum number of colors to color $[N]$ such that each color class is free of $k$-APs?

(ii) **Corners Problem (Coloring Version):** What is the minimum number of colors to color $[N]^k$ such that each color class is free of $k$-dimensional corners?

By a standard probabilistic tiling argument the coloring and subset size formulations of these problems are roughly equivalent. A $k$-AP-free subset with size $N/\delta$ implies a $k$-AP-free coloring with $\delta \cdot O(\log N)$ colors, and a similar connection holds for the corners problem. Therefore, a *lower* bound on the size of a $k$-AP-free subset (resp. corner-free subset) is the same as an *upper* bound on the $k$-AP-free coloring number (resp. corner-free coloring number) and consequently on the NIH complexity of Equality with a $k$-AP promise (resp. the NOF complexity of ExactlyN).

Figure 1 summarizes the relationships between the problems in additive combinatorics and their communication complexity reformulations.

The Chandra-Furst-Lipton equivalence, combined with the Ajtai-Szemerédi reduction to the $k$-AP Problem, shows that the NOF communication complexity of ExactlyN for $k = 3$ is at most $2\sqrt{2}\sqrt{\log N} + o(\sqrt{\log N})$ by Behrend's construction, and for $k > 3$ is at most $t2^{(t-1)/2}(\log N)^{1/t} + o\left((\log N)^{1/t}\right)$ for $t = \lceil \log k \rceil$ by Rankin's construction.

The protocols yielded by the above equivalence are *non-explicit*: we have an upper bound on their complexity but the underlying algorithms are non-constructive. This lack of explicitness comes from two places. First, the AP-free subsets of Behrend and Rankin are chosen using a generalized pigeonhole argument. Second, converting the subset size lower bounds into coloring upper bounds requires a probabilistic tiling argument. The problem for us is that such non-explicit protocols are difficult to analyze and therefore difficult to improve. Linial, Pitassi, and Shraibman remedied this situation by giving an explicit protocol for ExactlyN when $k = 3$ [20].

Recently, Linial and Shraibman gave the first protocol that improves the highest-order term for the ExactlyN problem for $k = 3$ since Behrend's original proof from 1946, yielding also an improved subset size lower bound for the 2-dimensional Corners Problem. Specifically, the constant of $2\sqrt{2} \approx 2.828$ is improved to $2\sqrt{\log e} \approx 2.402$ [21]. This protocol was found by closely examining the explicit protocol of Linial, Pitassi, and Shraibman. Linial and Shraibman's result was further improved by Green, who lowered the constant to $2\sqrt{2\log(4/3)} \approx 1.822$ [13].

## 1.3 Main result

In this paper, we begin by giving an explicit protocol for ExactlyN with cost that matches the construction of Rankin. Then we identify an optimization of this protocol which we exploit to give the first improvement in the highest-order term for every constant $k$:

▶ **Theorem 1.** *The number-on-forehead communication complexity of* ExactlyN *with $k$ players is at most*

$$\left(1 - \frac{c_k}{t}\right) t 2^{(t-1)/2} (\log N)^{1/t} + o((\log N)^{1/t}),$$

*where $t = \lceil \log k \rceil$ and $c_k$ is a constant depending on $k$.*

▶ **Corollary 2.** *Let $t = \lceil \log k \rceil$. The improved protocol from Theorem 1 yields a corner-free subset of $[N]^{k-1}$ of size*[2]

$$N^{k-1} \cdot 2^{-\left(1-\frac{c_k}{t}\right) t 2^{(t-1)/2} (\log N)^{1/t} + o((\log N)^{1/t})}.$$

This is the first improvement in the higher-order term since Rankin's 1961 construction. (Rankin's construction gives the above bound but where $c_k = 0$ for all $k$.) Similar to the recent breakthrough due to Linial and Shraibman [21] and Green [13], our protocol achieves a constant factor improvement, and for $k = 3$ and $k = 4$ we match Green's bound (and improve on Rankin's bound).

▶ Remark 3. Because $c_k$ degrades as $k$ increases, if $k$ is *not* of form $2^j + 1$ for some $j \in \mathbb{N}$ the best way to use the protocol from Theorem 1 is to reduce to the protocol for $k' = 2^{t-1} + 1$ players. This can easily be done by a communication-free transformation: the first $k'$ players solve ExactlyN′ where $N' = N - \sum_{k' < \ell \leq k} x_\ell$.

▶ Remark 4. In this paper, we are focused on improving the highest-order term in the bounds. However, we would like to highlight the work that has been done on improving the lower-order term as well. Elkin improved the lower-order term in Behrend's construction [9] (see also

[2] To get a corner-free set of $[N]^{k-1}$ we need to consider the ExactlyN problem where the inputs of $k$ player are from $[(k-1)N]$ and add up to $(k-1)N$. This results in extra terms depending on $k$ which can be pushed to the lower order term.

the note of Green and Wolf [15]) and Elkin's ideas were translated to Rankin's construction by O'Bryant [24]. Hunter [16] used similar techniques to improve the lower-order term of Green's construction. We leave applying these ideas to our new construction as an open problem (see Section 5).

**Outline of paper.** In Section 2, we give a history of the ExactlyN problem, including an outline of previous results based on Behrend and Rankin, which we hope helps the reader gain an intuition for the remainder of the paper. At the end of Section 2, we give an overview of our improved upper bound. In Section 3 we give an explicit protocol for ExactlyN for all $k$, building heavily on Rankin's construction. In Section 4, we give our improved protocol, proving Theorem 1. We conclude with some open problems in Section 5.

## 2 Overview of protocols for NOF ExactlyN

The history of the ExactlyN problem begins with the paper of Chandra, Furst, and Lipton that defines the NOF communication model [6]. By establishing a connection to the Corners Problem they obtained a non-constructive protocol for ExactlyN with cost $O(\sqrt{\log N})$, beating the cost of the trivial protocol. As mentioned in the introduction, an essential step in this protocol is a reduction to a *promise* instance of the Equality function in the NIH model. The reduction is summarized below.

**NOF ExactlyN to $k$-AP-free coloring.** First, the players each perform a reduction that yields the values $X_1, \ldots, X_k$ where $X_i$ is known only to Player $i$. These values are promised to be a $k$-AP and are equal if and only if the original instance of ExactlyN evaluates to 1. Then Player 1 announces the color of $X_1$ according to some agreed-upon $k$-AP-free coloring of $[kN]$: this is a coloring where no monochromatic subset of $[kN]$ has elements which form a non-trivial $k$-AP. Each other player then sends a single bit for whether or not the color of $X_i$ agrees with the color that Player 1 sent. They all agree if and only if $X_1, \ldots, X_k$ are all equal, as the $k$-AP promise implies that the colors can not be the same unless $X_1, \ldots, X_k$ are a trivial $k$-AP.

As discussed in the introduction the ExactlyN problem and the Corners problem in combinatorics are equivalent. Thus the Chandra-Furst-Lipton reduction can be seen as a reduction from the Corners problem to the problem of finding $k$-AP-free colorings. This latter reduction was already known before Chandra-Furst-Lipton connected these concepts to communication complexity (see [1] for the case of $k = 3$).

**$k$-AP-free coloring to $k$-AP-free set.** The reduction step of the protocol described above is conceptually simple. The technical part is finding a $k$-AP-free coloring of of $[N]$ where the number of colors is minimized.[3] This number can be estimated by the density version of the coloring problem: find the largest $k$-AP-free subset of $[N]$.

By a standard argument these problems are equivalent: a $k$-AP-free subset with size $N/\delta$ implies a $k$-AP-free coloring with $\delta \cdot O(\log N)$ colors and therefore gives a protocol with cost $\log \delta + O(\log \log N)$. Every known subset construction requires $\delta$ to be superlogarithmic in $N$, in which case the $O(\log \log N)$ term is negligible. Indeed, for $k = 3$ we know that superlogarithmic $\delta$ is necessary [18].

[3] The range of integers is $[N]$, instead of $[kN]$ as in the protocol; if we assume that $k$ is a constant this will not affect much.

In the rest of the paper we will switch freely between the coloring problems and their subset-size versions.

## 2.1 ExactlyN with 3 players

By the Chandra-Furst-Lipton reduction outlined above, a construction of a 3-AP-free subset of $[N]$ will result in a protocol for 3-player ExactlyN. Here we summarize the construction of a 3-AP-free subset due to Behrend [4]. All of the best known constructions of $k$-AP-free sets are essentially modifications of Behrend's basic framework.

Following prior work of Salem and Spencer [28], Behrend represents numbers in $[N]$ as vectors in $[q]^d$, where $q$ and $d$ are parameters to be chosen later subject to $q^d \geq N$. These vectors are the base-$q$ representations of numbers in $[N]$:

$$\text{base}_{q,d}(x) := (x_0, \ldots, x_{d-1}) \in [q]^d \text{ such that } x = \sum_{i=0}^{d-1} q^i x_i.$$

The idea behind Behrend's construction is that no three vectors in $[q]^d$ that form a line can lie on the same sphere. Suppose we had the following property: if three numbers $x, y, z \in [N]$ form a 3-AP, then their corresponding vectors $\text{base}_{q,d}(x), \text{base}_{q,d}(y), \text{base}_{q,d}(z)$ are in a line. Then one could choose the preimage of any sphere in $[q]^d$ to be the 3-AP-free set – no three distinct vectors in this sphere could be in a line, and so no three distinct numbers in the preimage could form a non-trivial 3-AP.

Unfortunately, a 3-AP in $[N]$ does not always correspond to a line in $[q]^d$. This is because of the possibility of carries: as a simple example, 9, 12, and 15 are a 3-AP but the vectors $(0, 9), (1, 2), (1, 5) \in [10]^2$ are not in a line. The strategy that Behrend takes is to avoid carries by limiting the $\ell_\infty$ norm of the vectors. Under this restriction there can never be any carries and so the desired property holds!

We now outline the complete argument. For $\ell \in [dq^2]$, define $A_\ell$ as the set of $x \in [N]$ such that each coordinate of $\text{base}_{q,d}(x)$ has value less than $q/2$ and $\|\text{base}_{q,d}(x)\|_2^2 = \ell$. Then $A_\ell$ is 3-AP-free. Furthermore, $\sum_\ell |A_\ell| = (q/2)^d$, so, by pigeonhole principle, for some value of $\ell$ we must have $|A_\ell| \geq \frac{(q/2)^d}{dq^2}$. To optimize this expression we set $d = \sqrt{2 \log N}$ and $q = N^{1/d}$. This gives us a 3-AP-free set of size at least $N \cdot 2^{-2\sqrt{2}\sqrt{\log N} + o(\sqrt{\log N})}$, which via the Chandra-Furst-Lipton reduction results in an ExactlyN protocol of cost $2\sqrt{2}\sqrt{\log N} + o(\sqrt{\log N})$.

**Explicit and improved protocols.** From Behrend's construction, the Chandra-Furst-Lipton reduction shows the existence of better-than-trivial protocols for ExactlyN. We would like to give a more *explicit* protocol, as an analysis of the details of the protocol may lead to new insights to construct better protocols (and corner-free sets). This motivaton led to the better 3-player ExactlyN protocols of Linial, Pitassi, and Shraibman [20], which was followed by Linial and Shraibman [21] and Green [13].[4]

The first explicit protocol of [20] had the general idea to go through the Chandra-Furst-Lipton reduction, yielding values $X_1, X_2, X_3$; player 1 will communicate the (squared) length of $\text{base}_{q,d}(X_1)$, and the other players should agree with this length if and only if $X_1 = X_2 = X_3$. Of course, this runs up against the same carry problem as in Behrend's

[4] Green's improvement is not phrased as a communication protocol, but was developed after further analyzing the Linial-Shraibman protocol.

construction, and here we do not have the liberty of excluding some vectors, as we want this protocol to work for every possible input. Linial, Pitassi, and Shraibman remedy this by having the players explicitly communicate information about the carry. Importantly, their protocol relies on the fact that each input can be seen by two players. The cost of the Linial-Pitassi-Shraibman protocol matches the cost of the non-constructive protocol from Chandra-Furst-Lipton.

Linial and Shraibman [21] observed that with the knowledge of two of the inputs, certain carries in the base-$q$ sum of the inputs are more likely than others. In particular, the entropy of the carry (conditioned on the information shared by certain players) is less than $d$. Linial and Shraibman give a small-cost protocol that only works for the inputs that have the most likely carry. Then, they show how to translate the inputs on which their protocol does not work to those that do. This process uses communication equal to the entropy of the carry. The total cost of this ExactlyN protocol is $2\sqrt{\log e}\sqrt{\log N} + o(\sqrt{\log N})$. Subsequent work of Green refined the argument of Linial and Shraibman and yields a protocol with cost $2\sqrt{2\log\frac{4}{3}}\sqrt{\log N} + o(\sqrt{\log N})$ [13].

## 2.2 ExactlyN with more than 3 players

Ideas from Behrend's construction can be used to build a larger $k$-AP-free set for $k > 3$. Rankin was the first to give such a construction [26]; see also the independent rediscovery of this result by Łaba and Lacey for a different presentation of the proof [19].

The key to Rankin's construction is that the line on which the three vectors fall in the intuition to Behrend's construction can be replaced with a higher-degree object as long as the number of vectors is sufficiently high. This motivates the definition of *polynomial progressions*.

▶ **Definition 5.** *A tuple of integers $(x_1, \ldots, x_k) \in \mathbb{Z}^k$ is a $k$-term degree-$m$ polynomial progression (denoted $k$-PP$_m$) if there is a degree-$m$ polynomial $p$ such that $\forall i \in [k]$, $x_i = p(i)$.*

▶ **Definition 6.** *A tuple of vectors over the integers $(v_1, \ldots, v_k) \in (\mathbb{Z}^d)^k$ is a $k$-term degree-$m$ vector polynomial progression (denoted $k$-vecPP$_m$) if there are degree-$m$ polynomials $p_j$ for each dimension $j \in [d]$ such that $\forall i \in [k]$, $v_i = (p_1(i), ..., p_d(i))$.*

*This definition can be rephrased to say that these are tuples of vectors where each dimension is a $k$-PP$_m$.*

Note that a $k$-PP$_1$ is just a $k$-AP and a $k$-vecPP$_1$ is just a sequence of vectors equally spaced on a line. Now we can update our intuition of Behrend's construction to include higher-degree progressions, and make an additional observation that will allow us to exploit this fact.

- Behrend relies on the fact that no three distinct vectors on a line in $\mathbb{R}^d$ can all be on a sphere. This is the special case of a more general fact: no $2m + 1$ vectors that form a $k$-vecPP$_m$ are all on a sphere.
- If a sequence of vectors form a $k$-vecPP$_m$, their squared lengths form a $k$-PP$_{2m}$.

We begin by using the first observation to find a $k$-PP$_m$-free set where $m$ is a power of two and satisfies $2m + 1 \geq k$. This is done similarly to Behrend's construction: using a pigeonhole argument, choose the preimage of a large set of vectors with the same length.

Now we can use this $k$-PP$_m$-free set (call this set $S$) to find a larger $k$-PP$_{m/2}$-free set. For each $s \in S$, add all of the vectors of squared length $s$ to our new set. The fact that this is $k$-PP$_{m/2}$-free follows from the second observation above: any $k$-PP$_{m/2}$ here would correspond to a $k$-PP$_m$ in $S$. We repeat this process, halving the degree at each step, until we have a set with no $k$-PP$_1$, i.e. a $k$-AP-free set.

In this outline we have omitted many details. In particular, just as in Behrend's construction vectors must be excluded from consideration based on their $\ell_\infty$ norm to avoid carries. Indeed, this exclusion is much stronger than in Behrend's construction: at the step for degree $m$, the set of allowed vectors has density exponentially small in $m$. Fortunately this deficiency is more than compensated for by the fact that vectors of many lengths, instead of simply one length, are included in the sets after the first step.

If we set the parameters correctly at every step, Rankin's construction gives a $k$-AP-free set of size at least $N \cdot 2^{-t2^{(t-1)/2}(\log N)^{1/t}+o((\log N)^{1/t})}$ where $t = \lceil \log k \rceil$. For $k = 3$ and $k = 4$, this matches Behrend's construction, which is expected as the construction is exactly the same. For $k \geq 5$, though, there is an improvement in the exponent of the $\log N$ term. Consequently, the cost of the protocol for ExactlyN from this construction is $t2^{(t-1)/2}(\log N)^{1/t} + o((\log N)^{1/t})$ for $t = \lceil \log k \rceil$.

## 2.3 Our results

Our first result gives an explicit protocol for ExactlyN with any number of players which matches the cost of the non-explicit protocol implied by Rankin. Our second result is an improved protocol for ExactlyN for more than 3 players that takes advantage of information shared by the players to improve the reduction to the NIH promise Equality problem.

**Sketch of explicit protocol.** (For full details, see Section 3.) The idea of this protocol is depicted in Figure 2. As in the previous protocols, the players first locally perform the reduction to NIH Equality problem with the promise that the new values $X_1, \ldots X_k$ form a $k$-AP. Then each player computes the base-$q$ representation vector of their inputs and the problem reduces to checking vector-Equality (Equality over vectors) with the promise that the input vectors form a $k$-vecPP$_1$. Next, they compute the squared length of these vectors and reduce to Equality with $k$-PP$_2$ promise. Although this promise is not as strong as the promise of being a $k$-AP, the reduction is helpful since their new inputs are much smaller than their initial inputs. The players continue by converting their new inputs into base-$q$ representation vectors again, and then computing the lengths of those vectors and so on. Thus, they keep reducing Equality with $k$-PP$_m$ promise to vector-Equality with $k$-vecPP$_m$ promise and vector-Equality with $k$-vecPP$_m$ promise to Equality with $k$-PP$_{2m}$ promise. When reducing the vector-Equality to Equality the degree of polynomial progression in the promise doubles, but the input size decreases in each reduction. When reducing Equality to vector-Equality the degree as well as the input size stays the same, and the input is now a vector polynomial progression which allows us to continue with the reductions.

This process can repeat at most $\lceil \log k \rceil$ times, as when the degree $m \geq k-1$, the promise $k$-PP$_m$ is trivially satisfied. At this point, the players are left to solve the Equality problem on their current inputs. So one of the players communicates the final length, and all the other players verify whether they have the same length.

To avoid carries during the process, every time the players reduce Equality to vector-Equality, they need to make sure that all the obtained vectors are small. If they are not small, one of the players computes and announces a translation which will make her vector small, referred to in this paper as the *shift*. If other players need different shifts, then the vectors are not equal, and we can terminate. Otherwise, all the players shift their vectors by the same amount before computing the lengths of the vectors again.

**Sketch of improved protocol.** (For full details see Section 4.) Recall that the goal of the players is to figure out whether $\sum_{i\in[k]} x_i = N$. The protocols that arise from previous constructions of corner-free sets involve computing the values $\text{base}_{q,d}(x_i)$, the base-$q$ representations of the players' inputs, thus creating a vector variant of the task in $d$-dimensional space. Unfortunately, just as in the explicit protocol above, there is the possibility of *carries*. Therefore, it is not necessarily the case that $\sum_{i\in[k]} \text{base}_{q,d}(x_i)$ is equal to $\text{base}_{q,d}\left(\sum_{i\in[k]} x_i\right)$.

Previous protocols [20, 21] have leveraged the NOF setting to have the players reason about the exact form of the carries. Specifically, these protocols have the players communicate information about the *carry string*: the length-$d$ string representing the carries performed in the summation. We take the same approach.

Let us rephrase the objective as figuring out whether $\sum_{i\in[k-1]} x_i = N - x_k$. Player $k$ can then look at the base-$q$ representations of the $x_i$s that they see and compute the carries required in the summation on the left-hand side of the expression. They can then convey the carry string to the other players. By adjusting the inputs accordingly, the players can end up with vectors $v_1$ to $v_{k-1}$ that actually do add up to the base-$q$ representation of the left-hand side of the expression as desired. With this strategy each entry of the carry string takes a value between 0 and $k-2$, so $d\log(k-1)$ bits of communication are required.

We can use the information shared by the players to lower the cost of this even further: we have not yet exploited the fact that each of the first $k-1$ players know $k-2$ of the inputs in the sum. Indeed, in the view of any of the first $k-1$ players there are only two values that each coordinate of the carry string can take, and these values are consecutive. Therefore, if the $k$th player simply communicates the *parity* of each coordinate of the carry string, each other player will have enough information to reconstruct the full carry string. This improves the communication to $d$ bits.

Note that using $d$ bits to communicate the carry matches the cost of just directly reducing it to an NIH problem and then switching to base-$q$ representations in the NIH model as in the explicit protocol above (see Figure 2); we need one final trick to find an advantage. Let us first consider the case where $k$ is even (so we are adding an odd number of vectors). In this situation it is more likely for the parities of entries in the carry string to take value 0, where probability is over the uniform distribution on the inputs. The idea is to use a protocol that assumes that the input is "nice": one where the parity-of-carry string takes the most likely value of 0 in every coordinate. If the input is indeed nice, the players simply proceed as if the $k$th player had communicated the all-0 string. Otherwise, we use communication to shift the inputs so that they fulfill the assumption.

The cost of this protocol is $d(1-\Omega(1))$ bits. The reason this is more efficient is that a larger-than-$2^{-d}$ fraction of inputs are nice, and hence (using a set-covering argument) fewer than $2^d$ possible shifts are required.

When $k$ is odd (so we are adding an even number of vectors), the fraction of nice inputs is $2^{-d}$. So the protocol as described above is more efficient only when $k$ is even. This can be rectified by considering the centered base-$q$ representations, where instead of using the digits $0,\ldots,q-1$ we use the digits $\lceil -(q-1)/2\rceil,\ldots,\lfloor q/2\rfloor$. This representation results in a larger-than-$2^{-d}$ fraction of nice inputs both when $k$ is even and when $k$ is odd.

## 3 Explicit NIH protocol for Rankin

In this section we give an explicit protocol for the number-in-hand Equality problem with the promise that the inputs form a $k$-AP that matches the cost of the non-explicit protocol guaranteed by Rankin's construction. As mentioned in the previous section, the general

strategy of our protocol is to convert the $k$-AP to a higher-degree polynomial progression by converting the integers into vectors, finding the squared length of those vectors (which leaves the parties again with integers), and repeating the process. Converting integers to vectors requires some care, and sidestepping potential problems in this step is the main technical contribution of this section.

Recall the definitions of $k$-$\mathrm{PP}_m$ and $k$-$\mathrm{vecPP}_m$ (Definitions 5 and 6). We define related communication tasks below. We define the following communication tasks, which are versions of the EQUALITY problem with the promise that the inputs form either a $k$-$\mathrm{PP}_m$ or $k$-$\mathrm{vecPP}_m$.

▶ **Definition 7.** *The communication task* $(k,[N])$-$\mathrm{PP}^{\mathrm{cc}}_m$ *is defined as follows.*

- *The input* $(x_1,\dots,x_k)\in[N]^k$ *is promised to be a* $k$-$\mathrm{PP}_m$*.*
- *The output is* 1 *if* $x_1=\cdots=x_k$ *(referred to as a trivial* $k$-$\mathrm{PP}_m$*) and* 0 *otherwise.*

▶ **Definition 8.** *The communication task* $(k,[q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$ *is defined as follows.*

- *The input* $(v_1,\dots,v_k)\in([q]^d)^k$ *is promised to be a* $k$-$\mathrm{vecPP}_m$*.*
- *The output is* 1 *if* $v_1=\cdots=v_k$ *(referred to as a trivial* $k$-$\mathrm{vecPP}_m$*) and* 0 *otherwise.*

We make the following observations about these tasks.

▶ **Observation 9.** $(k,[q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$ *is equivalent to* $\mathrm{AND}_d\circ(k,[q])$-$\mathrm{PP}^{\mathrm{cc}}_m$*. That is,* $(v_1,\dots,v_k)$ *is a valid input for* $(k,[q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$ *if and only if for each* $i\in[d]$*,* $(v_{1,i},\dots,v_{k,i})$ *is a valid input to* $(k,[q])$-$\mathrm{PP}^{\mathrm{cc}}_m$*. Furthermore, the output on* $(v_1,\dots,v_k)$ *is* 1 *if and only if the output of* $(k,[q])$-$\mathrm{PP}^{\mathrm{cc}}_m$ *on each* $(v_{1,i},\dots,v_{k,i})$ *is* 1*.*

▶ **Observation 10.** *When the degree* $m$ *is large enough, the promise in these tasks becomes trivially fulfilled. When* $m\geq k-1$*, any* $(x_1,\dots,x_k)\in[N]^k$ *is a valid input to* $(k,[N])$-$\mathrm{PP}^{\mathrm{cc}}_m$*. This is because you can find a degree* $k-1$ *polynomial* $p$ *such that* $p(i)=x_i$ *for all* $i\in[k]$*. Hence for* $m\geq k-1$*,* $(k,[N])$-$\mathrm{PP}^{\mathrm{cc}}_m$ *is equivalent to the* EQUALITY *function. Similarly for* $m\geq k-1$*,* $(k,[q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$ *is also equivalent to the* EQUALITY *function.*

In this section we show explicit protocols exhibiting the following upper bound for the communication tasks.

▶ **Theorem 11.** *Let* $m\leq k-1$ *and* $t=\lceil\log(k/m)\rceil$*. Then the number-in-hand communication complexity of computing* $(k,[N])$-$\mathrm{PP}^{\mathrm{cc}}_m$ *is at most*

$$t2^{(t-1)/2}\sqrt[t]{m^{t-1}\log N}+O(tk^2\log\log N).$$

*For* $m\leq(k-1)/2$*, the number-in-hand communication complexity of* $(k,[q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$ *is at most*

$$(t-1)2^{(t-2)/2}\sqrt[t-1]{(2m)^{t-2}\log(q^2d)}+O(tk^2\log\log(q^2d)).$$

As a special case (setting $m=1$) this yields the desired protocol for NIH EQUALITY with $k$-AP promise. See Figure 2 for an illustration. The figure also shows where our improvement for NOF EXACTLYN comes into play; this is described in detail in Section 4.

The proof of Theorem 11 is given in Section 3.3. It uses as subroutines two protocols that we present and analyze below.

- Protocol 1 gives us a way to reduce the vector polynomial progression task $(k,[q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$ to the integer polynomial progression task $(k,[q^2d])$-$\mathrm{PP}^{\mathrm{cc}}_{2m}$ as long as $k>2m$. Note that we have made the problem harder by moving from degree $m$ to degree $2m$ but we have also decreased the input size from $d\log q$ bits per input to $2\log q+\log d$ bits per input.
- Protocol 2 gives us a way to reduce the integer polynomial progression task $(k,[N])$-$\mathrm{PP}^{\mathrm{cc}}_m$ to the vector polynomial progression task $(k,[q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$. This protocol uses $md$ bits of communication and requires that $q$ is a multiple of $2^m$, $q^d\geq N$ and $k\geq m+2$.

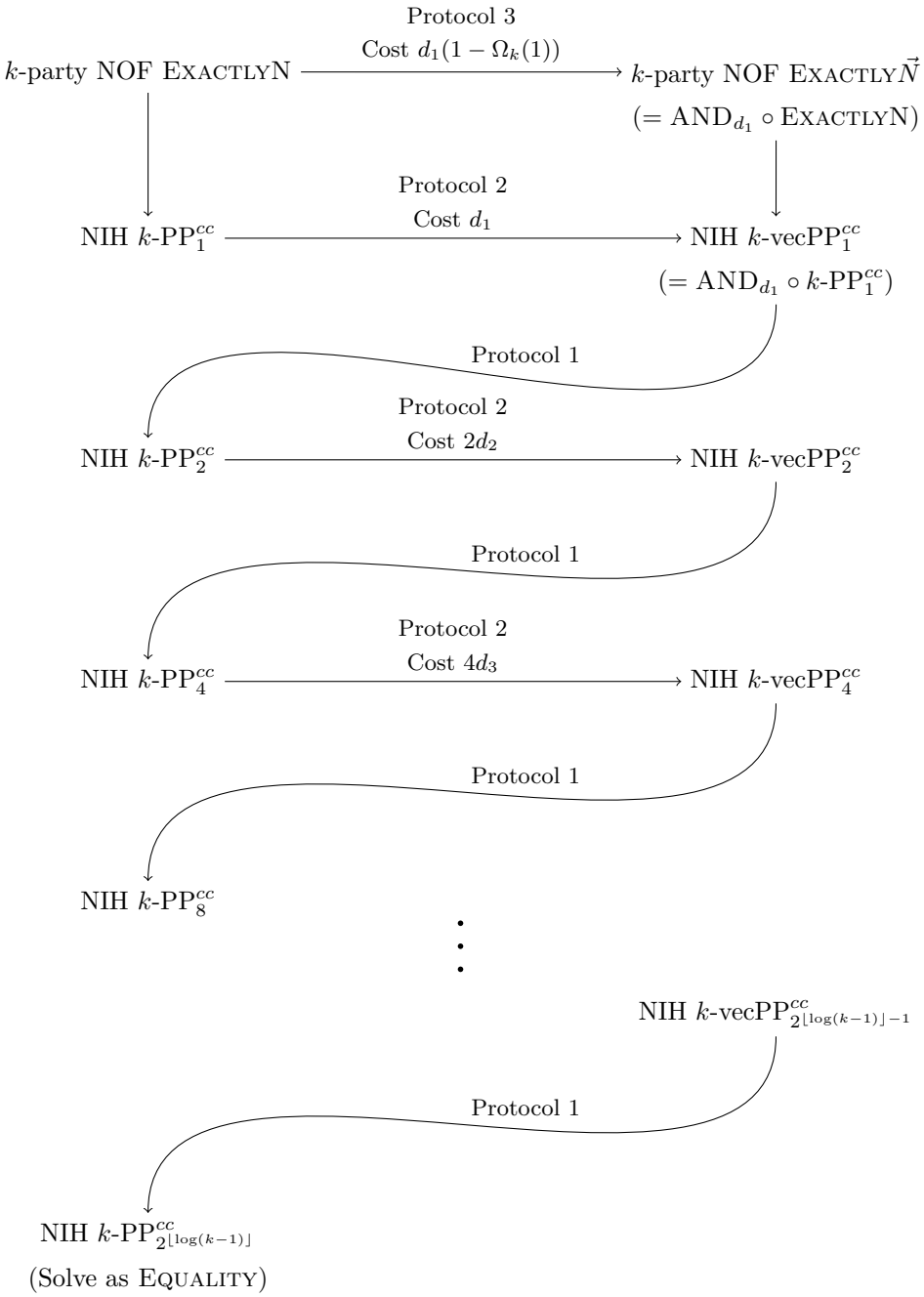


**Figure 2** The list of reductions used in protocols for $k$-party NOF EXACTLYN. Reductions that do not mention a cost are 0-cost reductions.

**Algorithm 1** A reduction from $(k, [q]^d)$-vecPP$^{\text{cc}}_m$ to $(k, [q^2 d])$-PP$^{\text{cc}}_{2m}$.

**Input:** $v_1, v_2, \ldots, v_k \in [q]^d$ distributed among $k$ players in the NIH model
**Promise:** $v_1, v_2, \ldots, v_k$ form a $k$-vecPP$_m$ with $k > 2m$
**Output:** $x_1, x_2, \ldots, x_k \in [q^2 d]^k$ distributed among the $k$ players in the NIH model such that $x_1, x_2, \ldots, x_k$ form a $k$-PP$_{2m}$, trivial if and only if $(v_1, \ldots, v_k)$ is trivial

1: For each $i \in [k]$, Player $i$ computes $x_i := \|v_i\|^2$.

**Algorithm 2** A reduction from $(k, [N])$-PP$^{\text{cc}}_m$ to $(k, [q]^d)$-vecPP$^{\text{cc}}_m$.

**Input:** $x_1, x_2, \ldots, x_k \in [N]$ distributed among $k$ players in the NIH model
(Assume $2^m | q, q^d \geq N$)
**Promise:** $x_1, x_2, \ldots, x_k$ form a $k$-PP$_m$ with $k \geq m + 2$
**Output:** $v_1, v_2, \ldots, v_k \in ([q]^d)^k$ distributed among the $k$ players in the NIH model such that either
(a) $v_1, v_2, \ldots, v_k$ form a $k$-vecPP$_m$, trivial if and only if $(x_1, \ldots, x_k)$ is trivial, or
(b) $x_1, x_2, \ldots, x_k$ was a non-trivial $k$-PP$_m$ and at least one of the players knows this.

1: For each $i \in [k]$, Player $i$ computes $w_i \leftarrow \text{base}_{q,d}(x_i)$.
2: $c \leftarrow q/2^m$
3: For each $i \in [k]$, Player $i$ computes two vectors:
- $s_i = (\lfloor w_{i,1}/c \rfloor, \ldots, \lfloor w_{i,d}/c \rfloor)$ and
- $v_i = (w_{i,1} \pmod c, \ldots, w_{i,d} \pmod c)$.

4: Player 1 broadcasts $s_1$.
5: For each $i \in [k]$, Player $i$ checks if $s_i = s_1$. If they are not equal, player $i$ notes that the input was a non-trivial $k$-PP$_m$.

## 3.1 Analysis of Protocol 1

The input $(v_1, \ldots, v_k)$ is promised to be a $k$-vecPP$_m$. Let $p_1, \ldots, p_d$ be the degree-$m$ polynomials associated with them, in the sense that $v_i = (p_1(i), \ldots, p_d(i))$. Define the degree-$2m$ polynomial $p' := \sum_{j \in [d]} p_j^2$. Note that the $x_i$ computed in the protocol is merely $p'(i)$. Hence $(x_1, \ldots, x_k)$ is a $k$-PP$_{2m}$. If the original $k$-vecPP$_m$ was trivial, then the computed $k$-PP$_{2m}$ is also trivial. On the other hand if any $p_j$ is non-constant, then $p'$ is also non-constant (any monomial of maximal degree among the $p_j$s will get squared and hence not get cancelled in $p'$). Assuming $k > 2m$, the non-constant polynomial $p'$ cannot take the same value on $k$ different points and so the $k$-PP$_{2m}$ is non-trivial.

The cost of this protocol is 0 since there is no communication during the protocol.

## 3.2 Analysis of Protocol 2

We start with a useful statement about polynomials. Define the function $L$ as follows:

$$L(a_0, \ldots, a_{m+1}) = \sum_{i=0}^{m+1} (-1)^i \binom{m+1}{i} a_i.$$

▷ Claim 12 (folklore). Let $k \geq m + 2$. The sequence $(x_1, \ldots, x_k)$ forms a $k$-PP$_m$ if and only if

$$L(x_1, \ldots, x_{m+2}) = \cdots = L(x_{k-m-1}, \ldots, x_k) = 0.$$

A proof of this claim may be found in the full version. From Observation 9 it immediately follows that vectors $(v_1, \ldots, v_k)$ form a $k$-vecPP$_m$ if and only if $L(v_1, \ldots, v_{m+2}) = \cdots = L(v_{k-m-1}, \ldots, v_k) = \vec{0}$.

Now we can analyze the correctness of Protocol 2. Recall that parameters $q$ and $d$ are set such that $q^d \geq N$ and $q$ is a multiple of $2^m$. Let $S$ be the set of numbers in $[N]$ whose base-$q$ representations only have entries less than $q/2^m$.

▷ Claim 13. Let $(x_1, \ldots, x_k)$ be a $k$-PP$_m$ with each $x_i \in S$. Then their base-$q$ representations $(v_1, \ldots, v_k)$ form a $k$-vecPP$_m$, trivial if and only if the $k$-PP$_m$ was trivial.

Proof. Define the vector $w$ as $w := L(v_1, \ldots, v_{m+2})$. The sum of the positive coefficients in the map $L$ is $\sum_{i \in [m+1],\, i \text{ even}} \binom{m+1}{i} = 2^m$, so each entry in $w$ is less than $q/2^m \cdot 2^m = q$. Similarly we can see that each entry is larger than $-q$. Rearranging the summations in the definition of $w$, we obtain

$$\begin{aligned}\sum_{j \in [d]} w_j q^{j-1} &= \sum_{j \in [d]} L(v_{1,j}, \ldots, v_{m+2,j}) q^{j-1} \\ &= L\left( \sum_{j \in [d]} v_{1,j} q^{j-1}, \ldots, \sum_{j \in [d]} v_{m+2,j} q^{j-1} \right) \\ &= L(x_1, \ldots, x_{m+2}) = 0.\end{aligned}$$

The first non-zero entry of $w$, say $w_i$, must be a multiple of $q$, otherwise $\sum w_j q^{j-1} \mod q^i \neq 0$. Since each entry of $w$ is larger than $-q$ and smaller than $q$, $w$ must be equal to $\vec{0}$. The same argument works to show that $L(v_2, \ldots, v_{m+3}) = \cdots = L(v_{k-m-1}, \ldots, v_k) = \vec{0}$. So we can conclude that $(v_1, \ldots, v_k)$ form a $k$-vecPP$_m$. Since the operation of taking the base-$n$ representation is a bijection, $x_1 = \cdots = x_k$ if and only if $v_1 = \cdots = v_k$. ◁

Clearly if in line 5 a player notes that $s_i \neq s_1$, that player's input is different from the input of Player 1, and so the $k$-PP$_m$ held by the players must have been non-trivial. We now prove that if no player has $s_i \neq s_1$, then the vectors they compute at the end form a $k$-vecPP$_m$. Note that the $v_i$ computed in line 3 can equivalently be written as $v_i := w_i - cs_i$. Since we are now analyzing the case when the locally-computed $s_i$s are all equal, the vector $v_i$ can be written as $v_i = w_i - cs_1$. Since it lies in $\{0, 1, \ldots, c-1\}^d$, it is the base-$q$ representation of an integer $T(v_i) := \sum_j v_{i,j} q^{j-1}$.

Since $T : (a_1, \ldots, a_d) \mapsto \sum_j a_j q^{j-1}$ is a linear transform, $T(v_i) = T(w_i) - T(cs_1)$. We know $T(w_i) = x_i$, so $T(v_i) = x_i - T(cs_1)$. Hence $T(v_1), \ldots, T(v_k)$ are just $x_1, \ldots, x_k$ shifted by the integer $T(cs_1)$. Hence $T(v_1), \ldots, T(v_k)$ also form a $k$-PP$_m$. Since every entry of their base-$q$ representation is at most $c - 1 < q/2^m$, we can use Claim 13 to conclude that $v_1, \ldots, v_m$ are a $k$-vecPP$_m$, trivial if and only if the $x_i$s were. This proves the correctness of the protocol.

The cost of this protocol is $md$ since the only communication that occurs is in Line 4 where Player 1 broadcasts an element of $\{0, \ldots, 2^m - 1\}^d$.

## 3.3 Combining Protocols 1 and 2

Our protocol for $(k, [N])$-PP$^{\text{cc}}_m$ uses Protocols 1 and 2 to repeatedly reduce the problem until it becomes an instance of the form $(k, [N'])$-PP$^{\text{cc}}_{m'}$ with $m' \geq k/2$. At this point they can no longer reduce the input size through these reductions, and so they solve this problem as an Equality problem: Player 1 reveals their input and all the other players communicate 0 if

their input differs or if at any point in the reductions via Protocol 2 they noted that the input was a non-trivial $k$-PP (see line 5). They communicate 1 otherwise. The output of the protocol is 1 if all the players communicate 1. The correctness of this protocol is easy to verify. The cost of the protocol depends on the parameters chosen during the reductions, and we analyze this in the proof.

**Proof of Theorem 11.** We prove the claim by induction on $t = \lceil \log(k/m) \rceil$.

The base case is when $t = 1$, corresponding to $k/2 \leq m \leq k-1$. Since $(k, [N])$-$\mathrm{PP}^{\mathrm{cc}}_m$ is a promise version of Equality on $\log N$ bits it can be solved by player 1 broadcasting their input and the other players using 1 bit each to convey whether their inputs match that of player 1. This protocol requires $\log N + k$ bits and works for all $m$.

For the inductive step, let $\lceil \log(k/m) \rceil = i + 1$. Since $i + 1$ is at least 2, we have $k > 2m$. This means we can use Protocol 2 to reduce it to $(k, [q]^d)$-$\mathrm{vecPP}^{\mathrm{cc}}_m$ and then Protocol 1 to reduce that to $(k, [q^2 d])$-$\mathrm{PP}^{\mathrm{cc}}_{2m}$. Since $\lceil \log(k/2m) \rceil = i$, by our induction hypothesis we already have an upper bound on the communication complexity of $(k, [q^2 d])$-$\mathrm{PP}^{\mathrm{cc}}_{2m}$.

Going via this reduction we get a protocol of cost

$$md + i2^{(i-1)/2} \sqrt[i]{(2m)^{i-1} \log q^2 d} + O(ik^2 \log\log q^2 d),$$

assuming $q^d \geq N$ and $q$ is a multiple of $2^m$ (this condition is required for us to run Protocol 2 with cost $md$). We can easily find the minimum of a closely related quantity that captures the essence of the minimization task.

▷ Claim 14. The following equality holds:

$$\min_{q', d' \in \mathbb{R}_+, q'^{d'} = N} md' + i2^{(i-1)/2} \sqrt[i]{(2m)^{i-1} \log q'^2} = (i+1) 2^{i/2} \sqrt[i+1]{m^i \log N}.$$

The minimum is achieved when $md' = 2^{(i-1)/2} \sqrt[i]{(2m)^{i-1} 2 \log q'} = 2^{i/2} \sqrt[i+1]{m^i \log N}$.

Proof. Since $d' (\sqrt[i]{\log q'})^i = \log N$, we have

$$(md') \left( 2^{(i-1)/2} \sqrt[i]{(2m)^{i-1} 2 \log q'} \right)^i = 2^{i(i+1)/2} m^i \log N.$$

This is the product of $i + 1$ terms: one term is $md'$ and the other $i$ terms are equal to $2^{(i-1)/2} \sqrt[i]{(2m)^{i-1} 2 \log q'}$. The quantity we want to minimize is exactly the sum of these terms. This sum is minimized when each of the terms are the same, and hence equal to the $i + 1$th root of the product. ◁

In our actual minimization problem we want to ensure that $q$ is a multiple of $2^m$ and $d$ is a natural number, and we also are minimizing a larger quantity. In the rest of the proof we show that accounting for these only adds to the lower order term. Let $q', d'$ be the optimal values in Claim 14. We can always find a $q \in [q', q' + 2^m)$ and $d \in [d', d' + 1)$ that satisfy our conditions. Plugging these in to our original minimization task, we get an upper bound of

$$m(d' + 1) + i2^{(i-1)/2} \sqrt[i]{(2m)^{i-1} \log((q' + 2^m)^2 (d' + 1))} + O(ik^2 \log\log q^2 (d' + 1)).$$

Using $\sqrt[i]{a + b} \leq \sqrt[i]{a} + \sqrt[i]{b}$ and $\log(a + b) \leq \log a + \log b$ for $a, b \geq 2$, this is in turn upper bounded by

$$\begin{aligned} &md' + i2^{(i-1)/2} \sqrt[i]{(2m)^{i-1} \log q'^2} \\ &+ m + i2^{(i-1)/2} \sqrt[i]{(2m)^{i-1}} (\sqrt[i]{2 \log 2^m} + \sqrt[i]{\log d}) + O(ik^2 \log\log q^2 d). \end{aligned}$$

We know the first two terms add up to $(i+1) 2^{i/2} \sqrt[i+1]{m^i \log N}$. We analyze the other terms using the fact that $2^{i+1} \leq k/m$.

- $i2^{(i-1)/2}\sqrt[i]{(2m)^{i-1}\log d}$: Since we choose a value of $d$ that is at most $k^{\,i+1}\sqrt{\log N}+1$, this term is at most $\log\log N+1$ when $i=1$ and $o(k^2\log\log N)$ otherwise.
- $i2^{(i-1)/2}\sqrt[i]{(2m)^{i-1}2\log 2^m}$: This is just $i2^{(i-1)/2}2m$, which is at most $k$.
- $ik^2\log\log q^2 d$: This is at most $ik^2\log\log N$ since $q^2 d \ll q'^{d'}=N$.
- $m$ is at most $k$.

Hence our final bound is

$$(i+1)2^{i/2}\,\sqrt[i+1]{m^i\log N}+O((i+1)k^2\log\log N).$$ ◀

## 4 Improved NOF protocol for ExactlyN

In this section we will show how to use information shared by the players to improve the reduction to the NIH promise Equality problem.

Recall that the goal of the players is to figure out whether $\sum_{i\in[k-1]}x_i = N - x_k$. We will use the high-level ideas described in Section 2.3. We now formally define the centered base-$q$ representation and carry-related notions, and then present the protocol.

### 4.1 Centered base-$q$ representations, carry strings and carry vectors

For simplicity, assume $q$ is odd. For an integer $x\in\{-(q^d-1)/2,\dots,(q^d-1)/2\}$, the centered base-$q$ representation of $x$ is a vector $\mathrm{base}^{\pm}_{q,d}(x)$ defined as the unique $v\in\{-(q-1)/2,\dots,(q-1)/2\}^d$ such that $x=\sum_{j\in[d]}v_jq^{j-1}$.

When adding together numbers $x_1$ through $x_t$ which have centered base-$q$ representations $v_1$ through $v_t$, we can get the centered base-$q$ representation of the sum by adding $v_1$ through $v_t$ but then modifying the result to take care of the carries. This is captured by the following process. (We require here that $t<q$, and this will be the case whenever we use this.)

- Let $w=\sum_{i\in[t]}v_i$.
- Define a *carry string* $s\in\mathbb{Z}^d$ as follows
  - $s_1$ is the unique integer such that $w_1\in\{s_1q-(q-1)/2,\dots,s_1q+(q-1)/2\}$.
  - For $j\in\{2,\dots,d\}$, $s_j$ is the unique integer such that $w_j+s_{j-1}\in\{s_jq-(q-1)/2,\dots,s_jq+(q-1)/2\}$.
- Define a *carry vector* $v_s\in\mathbb{Z}^{d+1}$ as $\sum_{j\in[d]}s_j(e_{j+1}-qe_j)$.
- Then $w+v_s=\mathrm{base}^{\pm}_{q,d+1}\left(\sum_{i\in[t]}x_i\right)$. (Here $w$ is viewed as a $(d+1)$-dimensional vector with $w_{d+1}=0$.)

The following claim will be useful for communicating the carry to players in the NOF model.

▷ Claim 15. Let $v_1,\dots,v_t\in\{-(q-1)/2,\dots,(q-1)/2\}^d$ and $s$ be the carry string of $\sum_{i\in[t]}v_i$. Given only $\{s_j \pmod 2\}_{j\in[d]}$ and $v_2,\dots,v_t$, one can reconstruct $s$ entirely.

Proof. We prove this by induction. The base case is that we can reconstruct $s_1$, and the inductive step shows that given $s_{j-1}$ and the information provided to us we can reconstruct $s_j$. Let $v_x=\sum_{i\in\{2,\dots,t\}}v_i$. We can compute $v_x$ with the information provided. Although we do not know $v_1$, we know that each entry of $v_1$ lies in $\{-(q-1)/2,\dots,(q-1)/2\}$.

For the base case, let $\alpha$ be the unique integer such that $v_{x,1}\in\{\alpha q-(q-1)/2,\dots,\alpha q+(q-1)/2\}$. If $v_{x,1}=\alpha q$, then with the addition of $v_{1,1}$ it will still remain in this interval and so $s_1=\alpha$. If $v_{x,1}<\alpha q$, then with the addition of $v_{1,1}$ it will either remain in the same interval or move to the interval corresponding to $\alpha-1$. So $s_1\in\{\alpha-1,\alpha\}$. Similarly if $v_{x,1}>\alpha q$, we know $s_1\in\{\alpha,\alpha+1\}$. In any of these cases finding out $s_1 \pmod 2$ will specify $s_1$ exactly.

The inductive step is similar. Assume we know $s_{j-1}$. By definition $s_j$ is defined by which interval $v_{x,j} + v_{1,j} + s_{j-1}$ lies in. We know the value of $v_{x,j} + s_{j-1}$ and so again $s_j$ depends on where the addition of $v_{1,j}$ can move it. With the same reasoning as before, finding out $s_j \pmod 2$ will specify $s_j$ exactly. ◁

## 4.2 A reduction to a vector variant

Protocol 3 is a reduction from ExactlyN to a vector variant that we term Exactly$\vec{N}$. In this protocol, players have as inputs (in the NOF model) $x_1, \ldots, x_k$. Player $k$ then broadcasts a shift so that all the players can compute new inputs $a_1$ to $a_k$ (still in the NOF model) such that $\sum_{i\in[k]} x_i = N \iff \sum_{i\in[k-1]} a_i = a_k$. These new inputs are also designed to have the property that if you take the base-$q$ representations of these inputs (called $w_1, \ldots, w_k$ in the protocol), and you look at the carry string obtained by adding $w_1$ through $w_{k-1}$, all of its entries are even. From Claim 15, this will allow all of the players to know the exact carry string $w_s$ and for them to shift the vector $w_k$ by it in order to ensure that $\sum_{i\in[k]} x_i = N \iff \sum_{i\in[k-1]} w_i = w_k - w_s$.

This vector variant of ExactlyN is then used to create a protocol for ExactlyN in Section 4.3.

**Algorithm 3** A reduction from NOF ExactlyN to NOF Exactly$\vec{N}$.

**Input:** $x_1, x_2, \ldots, x_k \in [N]$ are distributed among the $k$ players in the NOF model $q^d \geq N$

**Output:** $v_1, v_2, \ldots, v_k \in \{-kq, \ldots, kq\}^{d+1}$ are distributed among the $k$ players in the NOF model, with $\sum_{i\in[k]} v_i = \vec{0}$ if and only if $\sum_{i\in[k]} x_i = N$.

1: Player $k$ broadcasts a $\delta \in \mathbb{Z}^{k-1}$ such that
   **(a)** for each $i \in [k-1]$, $x_i + \delta_i \in \{-(q^d-1)/2, \ldots, (q^d-1)/2\}$, and
   **(b)** the assertion in Line 5 holds.
2: For $i \in [k-1]$, $a_i \leftarrow x_i + \delta_i$, $a_k \leftarrow N - x_k + \sum_{i\in[k-1]} \delta_i$.
3: For $i \in [k-1]$, let $w_i \leftarrow \text{base}^{\pm}_{q,d}(a_i)$ and let $w_k \leftarrow \text{base}^{\pm}_{q,d+1}(a_k)$.
4: Player $k$ computes $s \in \{-kq, \ldots, kq\}^d$, the carry string of $\sum_{i\in[k-1]} w_i$.
5: **Assert:** For each $j \in [d]$, $s_j \pmod 2 = 0$.
6: For each $i \in [k]$, Player $i$ computes $s$ and the carry vector $w_s$.
7: For each $i \in [k-1]$, $v_i := w_i$ and $v_k := -w_k + w_s$.

### 4.2.1 Correctness of the reduction

Let us first note that Line 1 is always achievable. That is, that there is always a $\delta$ that player $k$ can compute such that the assertion in Line 5 holds. One such $\delta$ is $(-x_1, \ldots, -x_{k-1})$, which player $k$ can compute. With this $\delta$, each $a_i$ is 0 for $i \in [k-1]$. The corresponding $w_i$s would also be 0 vectors and the carry string of $\sum_{i\in[k-1]} w_i$ would also be a string of 0s. This carry string satisfies the assertion that for each $j \in [d]$, $s_j \pmod 2 = 0$.

Now we prove the correctness of the protocol assuming only that the assertion in Line 5 holds.

- We start by showing that $(v_1, \ldots, v_k)$ are indeed known to the players in the NOF model. The vector $w_i$ depends only on $x_i$ and $\delta_i$, which are known to all players except player $i$. Since the assertion in Line 5 holds, every player knows that each entry of $s$ is even. Along with the fact that every player misses at most one of the summands in $\sum_{i\in[k-1]} w_i$, from Claim 15 we see that every player does in fact know the string $s$. The carry vector $w_s$ is a function of $s$, and hence they know $w_s$ as well. The vector $v_i$ depends only on $w_i$ and $w_s$, so all the players other than player $i$ can compute $v_i$.

- We finish by showing that $\sum_{i\in[k]} v_i = \vec{0}$ if and only if $\sum_{i\in[k]} x_i = N$.

$$\begin{aligned}
\sum_{i\in[k]} x_i = N &\iff \sum_{i\in[k-1]} a_i = a_k && \text{(definition of } a_i\text{'s)}\\
&\iff \sum_{i\in[k-1]} w_i + w_s = w_k && \text{(definition of } w_i\text{'s and the carry vector)}\\
&\iff \sum_{i\in[k]} v_i = \vec{0}. && \text{(definition of } v_i\text{'s)}
\end{aligned}$$

It is easy to see that for each $i \in [k-1]$ $v_i \in \{-(q-1)/2, \ldots, (q-1)/2\}^d$, (which we will be viewing as a $d+1$-dimensional vector with $v_{i,d+1} = 0$). Since $v_k$ has a carry vector added to it, with the carries being as large as $(k-1)q$, $v_k \in \{-kq, \ldots, kq\}^{d+1}$.

### 4.2.2 Cost of the reduction

The communication in the protocol is entirely in Line 1. The cost of this line depends on the size of the smallest set $\Delta \subset \mathbb{Z}^{k-1}$ such that for any $x_1, \ldots, x_{k-1} \in [N]$ there exists $\delta \in \Delta$ which satisfies the requirements in Line 1. The communication cost is then merely $\lceil \log |\Delta| \rceil$ since Player $k$ only needs to send the index of an element of $\Delta$.

The size of $\Delta$ is related to the size of the set

$$\begin{aligned}
S := \{(a_1, \ldots, a_{k-1}) \in \{-(q^d-1)/2, \ldots, (q^d-1)/2\}^{k-1} \mid \\
\text{the carry string of } \sum_{i\in[k-1]} \text{base}^{\pm}_{q,d}(a_i) \text{ has only even entries}\}.
\end{aligned}$$

$\Delta$ is the smallest set of shifts of $S$ that covers $[N]^{k-1}$. We can show the following bounds on $|\Delta|$.

$$N^{k-1}/|S| \le |\Delta| \le ((2q^d)^{k-1}/|S|) \cdot k \log N.$$

The lower bound on $|\Delta|$ is straightforward. For the upper bound we use the probabilistic method. Choose shifts $\delta^{(1)}, \ldots, \delta^{(t)}$ uniformly at random from $\{-N-(q^d-1)/2, \ldots, (q^d-1)/2\}^{k-1}$. For any $\overline{x} = (x_1, \ldots, x_{k-1})$, there are exactly $|S|$ different shifts that would land $\overline{x}$ in $S$. Hence the probability that a uniformly random shift is good for $\overline{x}$ is $|S|/(q^d+N)^{k-1} \ge |S|/(2q^d)^{k-1}$. The probability that none of the $t$ shifts are good for $\overline{x}$ is at most $(1-|S|/(2q^d)^{k-1})^t$. Setting $t = ((2q^d)^{k-1}/|S|) \cdot k \log N$, this probability is at most $e^{-k\log N} \le 1/N^k$. Hence by a union bound over all $N^{k-1}$ possible values of $\overline{x}$, there is a positive probability that (and hence there exists a set of $t$ shifts such that) each $\overline{x}$ has a shift that is good for it.

The cost of the protocol is hence at most $k - 1 + \log(q^{d(k-1)}/|S|) + \log k + \log\log N + 1$.

So how large is $S$? Note that the integers from $-(q^d-1)/2$ to $(q^d-1)/2$ have centered base-$q$ representations ranging over all vectors in $\{-(q-1)/2, \ldots, (q-1)/2\}^d$. Hence

$$\begin{aligned}
\frac{|S|}{q^{d(k-1)}} &= \Pr_{x_1,\ldots,x_{k-1}\in\left\{-\frac{q^d-1}{2},\ldots,\frac{q^d-1}{2}\right\}} [(x_1, \ldots, x_{k-1}) \in S]\\
&= \Pr_{v_1,\ldots,v_{k-1}\in\left\{-\frac{q-1}{2},\ldots,\frac{q-1}{2}\right\}^d} \left[\text{carry string of } \sum_{i\in[k-1]} v_i \text{ has only even entries}\right].
\end{aligned}$$

We now use the following claim. The proof may be found in the full version.

▷ Claim 16. Let $r_1, \ldots, r_{k-1}$ be real numbers uniformly sampled from $[-1/2, 1/2)$.

$$\Pr_{r_1,\ldots,r_{k-1}}\left[\sum_{i\in[k-1]} r_i \pmod 2 \in [-1/2, 1/2)\right] = \frac{1}{2} + \frac{E_{k-1}}{2(k-1)!},$$

where $E_n$ is the $n$th Euler zigzag number.[5]

Observe that the above quantity represents the limiting behaviour, as $q \to \infty$, of a specific entry of the carry string being even. The rest of the proof will show that the probability that a specific entry (say, the $i$th entry) of the carry string is even is within an additive $3k/2q$ of the probability in Claim 16, regardless of what we fix the entries of $v_1$ to $v_{k-1}$ to be outside of their $i$th entries.

- The probability that $s_1$ is even is the probability that $k-1$ random numbers $a_1, \ldots, a_{k-1}$ chosen from $\{-(q-1)/2, \ldots, (q-1)/2\}$ add up to give an even carry. Note that the carry is even if and only if the sum modulo $2q$ lies in $\{-(q-1)/2, \ldots, (q-1)/2\}$. We approximate this by a probability arising from the following real-valued experiment. Take $k-1$ real numbers $r_1, \ldots, r_{k-1}$ from the interval $[-1/2, 1/2)$. Find the probability that their sum modulo 2 lies in $[-1/2, 1/2)$. The two processes are related as follows.
  Let the set $B = \{-(q-1)/2, \ldots, (3q-1)/2\}$ represent the set of integers modulo $2q$. Divide $[-1/2, 3/2)$ into $2q$ intervals of size $1/2q$ each. Let $i_1, \ldots, i_{k-1}$ be the index of the intervals that $r_1, \ldots, r_{k-1}$ lie in. Each $i$ is a uniformly random number from 1 to $q$, and so $a_j$ is distributed as the $i_j$th element of $B$. Let $i_s$ be the interval that the sum $\sum_j r_j \pmod 2$ lies in. Then $\sum_j a_j$ modulo $2q$ lies within the $i_s$ through $i_{s+k-2}$th elements of $B$.
  So either we have $\sum_j r_j \pmod 1 \in [1/2 - k/2q, 1/2)$, or else it must be the case that $\sum_j r_j \pmod 2 \in [-1/2, 1/2) \iff \sum_j a_j \pmod{2q} \in \{-(q-1)/2, \ldots, (q-1)/2\}$. Hence the difference in probabilities of the experiments is at most $\Pr[\sum_j r_j \pmod 1 \in [1/2 - k/2q, 1/2)]$. This is $k/2q$, since the addition of a uniformly random number between $[0, 1]$ to any random variable makes its distribution modulo 1 the uniform distribution.
- For other coordinates of the carry string another complication arises. Since the sum in a coordinate is the sum of $k-1$ random numbers plus the carry from the previous coordinate, that adds another change in the experiment. However, the carry from the previous coordinate is always within $\{-k+1, \ldots, k-1\}$ so it adds an uncertainty of $\pm k/2q$ to the sum in the real-valued experiment. Hence we can use the same real-valued experiment, except this time we bound the difference in probabilities as $\Pr[\sum_j r_j \pmod 1 \in [1/2 - k/q, 1/2 + k/2q)] = 3k/2q$.

Hence the probability that all entries of the carry string are even is at least $(1/2 + E_{k-1}/2(k-1)! - 3k/2q)^d$. The cost of the protocol is at most

$$d \log\left(\frac{1}{1/2 + E_{k-1}/2(k-1)! - 3k/2q}\right) + k + \log k + \log\log N.$$

Since $k/q \ll 1$ and $\frac{d}{dt}\log\left(\frac{1}{1/2+t}\right) = -\frac{2}{\ln 2} > -3$ at $t = 0$, this quantity is at most

$$d \log\left(\frac{1}{1/2 + E_{k-1}/2(k-1)!}\right) + d \cdot \frac{9k}{2q} + O(k + \log\log N),$$

with $9dk/2q$ being $o(1)$ if $d \le \log N / \log\log N$. In our usage we will have $d \le \sqrt{\log N}$.

[5] See entry A000111 in The On-Line Encyclopedia of Integer Sequences (starts at $E_0$) for more details.

To simplify this expression, define

$$c_k \triangleq 1 - \log\left(\frac{1}{1/2 + E_{k-1}/2(k-1)!}\right). \tag{1}$$

As $k$ grows, $c_k \to \frac{2}{\ln 2}\left(\frac{2}{\pi}\right)^k$. Protocol 3 uses $(1-c_k)d + O(k + \log\log N)$ bits of communication.

## 4.3 Putting everything together

Our protocol starts by running Protocol 3 with parameters $q, d$ such that $q^d \geq N$. The players end up with vectors $v_1, \ldots, v_k$, each in $\{-kq, \ldots, kq\}^{d+1}$, (in the NOF setting) and they want to know whether $\sum_{i\in[k]} v_i = \vec{0}$. Note that this sum is equal to $\vec{0}$ if and only if for each $j \in [d+1]$, $\sum_{i\in[k]} v_{i,j} = 0$. Each of these is an instance of ExactlyN with the inputs coming from $\{-kq, \ldots, kq\}$.

Now they can make a cost-0 reduction to NIH $(k, \{-k^3q, \ldots, k^3q\}^{d+1})$-vecPP$_1^{\text{cc}}$. This is because each instance of ExactlyN has a cost-0 reduction to $(k, \{-k^3q, \ldots, k^3q\})$-PP$_1^{\text{cc}}$ (see the full version for the proof) and because $(k, \{-k^3q, \ldots, k^3q\}^{d+1})$-vecPP$_1^{\text{cc}}$ is equivalent to $\text{AND}_{d+1} \circ (k, \{-k^3q, \ldots, k^3q\})$-PP$_1^{\text{cc}}$ (see Observation 9). One should note here that the reduction in works even when the input is allowed to include negative numbers. This is also true of Protocol 1, which is the first step in the NIH protocol for $(k, [q]^d)$-vecPP$_1^{\text{cc}}$ and which outputs a nonnegative $k$-PP$_2$.

We can now use the NIH protocol for $(k, \{-k^3q, \ldots, k^3q\}^{d+1})$-vecPP$_1^{\text{cc}}$ (Theorem 11) to complete the protocol. Let $t = \lceil \log k \rceil$. The cost of the NIH protocol is

$$C := (t-1)2^{(t-2)/2}\sqrt[t-1]{2^{t-2}\log(k^6q^2(d+1))} + O(tk^2 \log\log(k^6q^2(d+1))).$$

The total cost of the protocol is then $(1-c_k)d + O(k + \log k + \log\log N) + C$. As done in the proof of Theorem 11 we can optimize $d$ and $q$ to end up with a complexity of

$$\begin{aligned} &t2^{(t-1)/2}\sqrt[t]{(1-c_k)\log N} + O(tk^2 \log\log N) \\ &\leq \left(1 - \frac{c_k}{t}\right) t2^{(t-1)/2}\sqrt[t]{\log N} + O(tk^2 \log\log N). \end{aligned}$$

# 5 Open problems

In this paper we give the first explicit protocol for ExactlyN that matches the performance of Rankin's construction. We then use the details of this explicit protocol to find an improvement that relies on knowledge shared by the parties.

However, this improvement itself relies on an existential argument: there is a probabilistic argument in Section 4.2.2. Therefore our final improved protocol has a non-constructive part.

▶ **Open Problem 1.** *Give a completely explicit protocol that matches the performance of the NOF protocol from Theorem 1.*

The constructions of Behrend and Rankin use a pigeonhole argument over spheres in some vector space. As mentioned in Remark 4, there is a line of work that improves the lower-order terms of these constructions [9, 15, 24, 16]. The general strategy is to replace the spheres with thin *annuli*. We have not attempted to use annuli in our construction, but it seems to us that this might lead to an improvement in lower order terms in our case too.

▶ **Open Problem 2.** *Improve the lower-order terms of our corner-free set construction by replacing spheres with annuli.*

Our protocol exploits the shared information between the players in the NOF setting. As the number of parties increases the amount of shared information also increases. One might think that this would lead to a corresponding increase in the magnitude of the improvement in the NOF setting over the protocol described in Section 3, which makes no use of the shared information. However, this is not what we see: the factor of $(1 - c_k/t)$ from Theorem 1 actually grows as $k$ increases.

▶ **Open Problem 3.** *Give a corner-free set construction whose advantage over Rankin's construction improves as $k$ grows.*

The structure of Rankin's protocol seems to necessitate a lack of smoothness in the parameters of the construction. Namely, the best-known $k$-AP-free set construction when $k$ is not of the form $2^t + 1$ (for an integer $t$) is to round down to the nearest such value and proceed with the corresponding construction. Is it possible to obtain a bound that depends on $\log k$ instead of $\lceil \log k \rceil$? This would be exciting as it would require a different argument than the degree-doubling method used by Rankin.

▶ **Open Problem 4.** *Give a $k$-AP-free set construction that improves for each increase of the value $k$.*

Finally, an important open problem is to improve the large gap between the upper and lower bounds on the size of corner-free sets, where progress has been stuck for more than 15 years. We feel that it may be possible to substantially improve the NOF communication complexity of ExactlyN, by further exploiting the shared information in the NOF model. On the other hand, if substantial improvements are not possible for ExactlyN, strong lower bounds for ExactlyN would give a breakthrough separation of deterministic from randomized NOF protocols for an explicit and well-studied function. As mentioned in the introduction, the recent breakthrough result of Kelley and Meka proved an upper bound for 3-AP-free sets [18], nearly matching Behrend's construction.

However, corners appear to be a much more complicated combinatorial object, and upper bounds on corner-free sets have historically lagged behind those for 3-AP-free sets. Thus narrowing this gap is an important problem in additive combinatorics as well.

▶ **Open Problem 5.** *Narrow the gap between the best known upper and lower bounds on the NOF complexity of* ExactlyN.

## References


1 Miklós Ajtai and Endre Szemerédi. Sets of lattice points that form no squares. *Studia Scientiarum Mathematicarum Hungarica*, 9:9–11, 1974.

2 Ryan Alweiss, Shachar Lovett, Kewen Wu, and Jiapeng Zhang. Improved bounds for the sunflower lemma. *Annals of Mathematics*, 194(3):795–815, November 2021. doi:10.4007/annals.2021.194.3.5.

3 Paul Beame, Matei David, Toniann Pitassi, and Philipp Woelfel. Separating deterministic from randomized multiparty communication complexity. *Theory of Computing*, 6(9):201–225, November 2010. doi:10.4086/toc.2010.v006a009.

4 Felix A. Behrend. On sets of integers which contain no three terms in arithmetical progression. *Proceedings of the National Academy of Sciences of the United States of America*, 32(12):331–332, December 1946. doi:10.1073/pnas.32.12.331.

5 Richard Beigel and Jun Tarui. On ACC. *computational complexity*, 4(4):350–366, December 1994. doi:10.1007/BF01263423.

6 Ashok K. Chandra, Merrick L. Furst, and Richard J. Lipton. Multi-party protocols. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, STOC '83, pages 94–99, New York, NY, USA, December 1983. Association for Computing Machinery. doi:10.1145/800061.808737.

7 Ernie Croot, Vsevolod F. Lev, and Péter Pál Pach. Progression-free sets in $\mathbb{Z}_4^n$ are exponentially small. *Annals of Mathematics*, 185(1):331–337, 2017. doi:10.4007/annals.2017.185.1.7.

8 Zeev Dvir. On the size of Kakeya sets in finite fields. *Journal of the American Mathematical Society*, 22(4):1093–1097, 2009. doi:10.1090/S0894-0347-08-00607-3.

9 Michael Elkin. An improved construction of progression-free sets. *Israel Journal of Mathematics*, 184(1):93, July 2011. doi:10.1007/s11856-011-0061-1.

10 Jordan S. Ellenberg and Dion Gijswijt. On large subsets of $\mathbb{F}_q^n$ with no three-term arithmetic progression. *Annals of Mathematics*, 185(1):339–343, 2017. doi:10.4007/annals.2017.185.1.8.

11 W. Timothy Gowers. A new proof of Szemerédi's theorem. *Geometric & Functional Analysis GAFA*, 11(3):465–588, August 2001. doi:10.1007/s00039-001-0332-9.

12 W. Timothy Gowers. Hypergraph regularity and the multidimensional Szemerédi theorem. *Annals of Mathematics*, 166(3):897–946, 2007. doi:10.4007/annals.2007.166.897.

13 Ben Green. Lower bounds for corner-free sets. *New Zealand Journal of Mathematics*, 51:1–2, July 2021. doi:10.53733/86.

14 Ben Green and Terence Tao. New bounds for Szemerédi's theorem, III: A polylogarithmic bound for $r_4(N)$. *Mathematika*, 63(3):944–1040, 2017. doi:10.1112/S0025579317000316.

15 Ben Green and Julia Wolf. A note on Elkin's improvement of Behrend's construction. In David Chudnovsky and Gregory Chudnovsky, editors, *Additive Number Theory: Festschrift In Honor of the Sixtieth Birthday of Melvyn B. Nathanson*, pages 141–144. Springer, New York, NY, 2010. doi:10.1007/978-0-387-68361-4_9.

16 Zach Hunter. Corner-free sets via the torus, October 2022. doi:10.48550/arXiv.2209.10012.

17 Zander Kelley, Shachar Lovett, and Raghu Meka. Explicit separations between randomized and deterministic number-on-forehead communication, August 2023. arXiv:TR23-124.

18 Zander Kelley and Raghu Meka. Strong bounds for 3-progressions. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 933–973. IEEE, 2023. doi:10.1109/FOCS57990.2023.00059.

19 Izabella Łaba and Michael T. Lacey. On sets of integers not containing long arithmetic progressions, August 2001. doi:10.48550/arXiv.math/0108155.

20 Nati Linial, Toniann Pitassi, and Adi Shraibman. On the communication complexity of high-dimensional permutations. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, volume 124 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54:1–54:20, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2019.54.

21 Nati Linial and Adi Shraibman. An improved protocol for the Exactly-N problem. In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:8, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2021.2.

22 Shachar Lovett. Additive combinatorics and its applications in theoretical computer science. *Theory Comput.*, 8:1–55, 2017. doi:10.4086/toc.gs.2017.008.

23 Shachar Lovett, Raghu Meka, Ian Mertz, Toniann Pitassi, and Jiapeng Zhang. Lifting with sunflowers. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 104:1–104:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.104.

24 Kevin O'Bryant. Sets of integers that do not contain long arithmetic progressions. *The Electronic Journal of Combinatorics*, 18, November 2008. doi:10.37236/546.

**25** Pavel Pudlák. Boolean complexity and Ramsey theorems. In Jaroslav Nešetřil and Vojtěch Rödl, editors, *Mathematics of Ramsey Theory*, Algorithms and Combinatorics, pages 246–252. Springer, Berlin, Heidelberg, 1990. doi:10.1007/978-3-642-72905-8_17.

**26** Robert A. Rankin. Sets of integers containing not more than a given number of terms in arithmetical progression. *Proceedings of the Royal Society of Edinburgh Section A: Mathematical and Physical Sciences*, 65(4):332–344, 1961. doi:10.1017/S0080454100017726.

**27** Alexander A. Razborov. Lower bounds on the monotone complexity of some boolean functions. *Dokl. Akad. Nauk SSSR*, 281:354–357, 1985.

**28** Raphaël Salem and Donald C. Spencer. On sets of integers which contain no three terms in arithmetical progression. *Proceedings of the National Academy of Sciences of the United States of America*, 28(12):561–563, 1942. arXiv:87810.

**29** Ilya D. Shkredov. On a generalization of Szemeredi's theorem. *Proceedings of the London Mathematical Society*, 93(3):723–760, November 2006. doi:10.1017/S0024611506015991.

**30** Terence Tao and Van H. Vu. *Additive combinatorics*, volume 105. Cambridge University Press, 2006.

**31** Andrew Chi-Chih Yao. On ACC and threshold circuits. In *31st Annual Symposium on Foundations of Computer Science*, pages 619–627. IEEE, October 1990. doi:10.1109/FSCS.1990.89583.

**32** Yufei Zhao. *Graph Theory and Additive Combinatorics: Exploring Structure and Randomness*. Cambridge University Press, 2023.