2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS) | 979-8-3315-7132-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/FOCS63196.2025.00062

High-to-Low Dimensional PPA-completeness: Borsuk-Ulam, Tucker, Consensus Halving, and Ham Sandwich

Ruiquan Gao
Stanford University, USA
ruiquan@stanford.edu

Alexandros Hollender
University of Oxford, UK
alexandros.hollender@cs.ox.ac.uk

Aviad Rubinstein
Stanford University, USA
aviad@cs.stanford.edu

***Abstract*—The Borsuk-Ulam theorem states that every continuous odd function $f : \mathcal{S}^n \to \mathbb{R}^n$ must have a zero, i.e., an $x \in \mathcal{S}^n$ such that $f(x) = 0$. While such a zero is guaranteed to exist, finding it is known to be computationally intractable: it is PPA-complete already for $n = 2$. In this work, we show that the problem remains just as hard even if the function is mapping from a higher to a lower dimensional space. Namely, we prove that it is PPA-complete to find a zero of $f : \mathcal{S}^k \to \mathbb{R}^n$ for any constants $k \geq n \geq 2$.**

This result has very appealing consequences for other flagship PPA-complete problems such as Tucker, Consensus Halving, and Ham Sandwich. For example, in the Consensus Halving problem from fair division, we show that finding a partition that satisfies three agents with monotone valuations is PPA-complete, even if we allow any arbitrarily large constant number of cuts.

***Index Terms*—TFNP, computational topology, fair division**

I. Introduction

The Borsuk-Ulam theorem can be stated in the following form.

Borsuk-Ulam Theorem ([Bor33])**.** Every continuous odd function $f : \mathcal{S}^n \to \mathbb{R}^n$ admits a zero.

A function f defined on the n-dimensional sphere $\mathcal{S}^n$ is *odd*, if $f(-x) = -f(x)$ for all $x \in \mathcal{S}^n$. Equivalently, the theorem can also be stated as: for every continuous (not necessarily odd) function $f : \mathcal{S}^n \to \mathbb{R}^n$ there exists $x \in \mathcal{S}^n$ such that $f(-x) = f(x)$.

The Borsuk-Ulam theorem, and its various generalizations, have applications in many different areas of mathematics, including Measure Theory, Combinatorics, Functional Analysis, Differential Equations, and Algebra. A particularly famous example in combinatorics is the use of the theorem by Lovász [Lov78] to prove Kneser's conjecture [Kne55] regarding the chromatic number of Kneser graphs. Many further examples can be found in the surveys by Steinlein [Ste85], [Ste93] and Alon [Alo88], as well as in the book by Matoušek [Mat08] dedicated to Borsuk-Ulam and its applications.

In this work, our motivating application is a fundamental problem in the field of fair division called the Consensus Halving problem. In this problem we have a resource, modeled as the interval $I = [0, 1]$, and n agents, each with their own valuation function v_i assigning a value to each measurable subset of I. The goal is to partition the resource into two sets I^+ and I^- such that all n agents agree that I^+ and I^- have the same value. Surprisingly, this can always be achieved by using at most n cuts. Namely, it is possible to divide the interval I into pieces using at most n cuts and label each piece "+" or "-" such that for each agent the portion labeled "+" has the same value as the portion labeled "-".

Consensus Halving Theorem ([SS03])**.** For any continuous valuation functions $v_1, \dots, v_n$ over the interval $I = [0, 1]$, it is possible to partition the interval into two parts I^+ and I^- using at most n cuts such that $v_i(I^+) = v_i(I^-)$ for all i.

This result, which is proved using the Borsuk-Ulam theorem, has natural applications in fair division of land, including issues related to the Law of the Sea [SS03]. Although the general version of the theorem stated above was proved by Simmons and Su [SS03], its origins can be traced as far back as the work of Neyman [Ney46] and Hobby and Rice [HR65]. Furthermore, consensus halving is the continuous version of the classic *necklace splitting* problem first studied in the 80s [GW85], [AW86], [Alo87], and which is still actively studied in combinatorics [JPv21], [AEPT24].

The second application that we consider is the Ham Sandwich problem, the most fundamental mass partition problem. Consider a sandwich in three dimensional space consisting of three different ingredients. Then, the theorem says that using a single cut we can divide the sandwich into two parts such that each of the two parts contains exactly half of each ingredient. More formally, given three sets A_1, A_2, A_3 of finite measure in $\mathbb{R}^3$, there exists a plane H that simultaneously cuts each set in half, according to the Lebesgue measure. The theorem generalizes to any number d of sets in d-dimensional space. In fact, it is usually stated in the following even more general form.

Ham Sandwich Theorem ([Ste38], [ST42])**.** For any absolutely continuous finite measures $\mu_1, \dots, \mu_n$ over $\mathbb{R}^n$, there exists a hyperplane H that bisects all measures simultaneously, i.e., $\mu_i(H^+) = \mu_i(H^-)$ for all i.

The Ham Sandwich theorem dates back to the 1930s. Steinhaus first presented a proof for the three-dimensional case using the Borsuk-Ulam theorem and attributed it to Ba-

R.G. was supported by a Stanford Graduate Fellowship. A.R. was supported by NSF CCF-2112824, and a David and Lucile Packard Fellowship.

2575-8454/25/$31.00 ©2025 IEEE
DOI 10.1109/FOCS63196.2025.00062

nach [Ste38], [BZ04]. The general form was proved soon thereafter by Stone and Tukey [ST42], again through Borsuk-Ulam. The field of mass partitions has since flourished with an impressive variety of extensions and variants [RPS21].

Computation. Although knowing that a desired solution always exists, be it a consensus halving solution or a bisecting hyperplane cutting a ham sandwich, is very satisfying, perhaps the most pressing follow-up question is

Can we compute such a solution efficiently?

In order to understand the complexity of finding these solutions one must first understand the complexity of the topological tool that was used to prove the existence of a solution in the first place. In our case, this is the Borsuk-Ulam theorem. Unfortunately, no polynomial-time algorithm is known for finding a zero of a Borsuk-Ulam function. On the other hand, one cannot hope to prove that the problem is NP-hard, because it lies in TFNP, the class of total NP search problems that are always guaranteed to have a solution.[1] Such problems cannot be NP-hard, unless NP = coNP [MP91].

Fortunately, the complexity of TFNP problems can be understood by studying subclasses of TFNP that do have complete problems. Establishing that a problem is complete for one of these classes yields strong evidence that we should not expect a polynomial-time algorithm.[2] In this framework, the complexity of Borsuk-Ulam, and its discrete analog Tucker's lemma, is now well understood, as the two problems have been shown to be PPA-complete, even in dimension two [ABB20], [Pap94]. This has then opened the way for researchers to understand the complexity of TFNP problems where existence is proved using these theorems. In particular, various versions of the consensus halving problem and a discrete ham sandwich problem were subsequently proved to also be PPA-complete [FRG23], [FRHSZ23], [Hav22], [Sch22], [DFM22], [DFHM25].

For these various application problems, the membership in PPA usually follows readily from the mathematical proof of existence by checking that it can be turned into an efficient reduction. The far more interesting result, the PPA-hardness, is proved by presenting a reduction in the other direction: from the general topological theorem to the particular application problem. This approach is ubiquitous in the context of TFNP, where understanding the complexity of topological theorems has played a crucial role in unlocking the complexity of important application problems. The most notable such example is probably the complexity of computing Nash equilibria [DGP09], [CDT09], where one has to reduce from the corresponding topological theorem—Brouwer's fixed point theorem or its discrete version, Sperner's lemma—to prove PPAD-completeness. The class PPAD is a subclass of PPA, and this is naturally reflected in the fact that Borsuk-Ulam is a stronger theorem which can be used to prove the Brouwer fixed point theorem.

Coming back to our motivating application problem, Consensus Halving, the most relevant results for our work are those obtained by Deligkas et al. [DFRH22] who proved that:

- For general valuation functions, the problem is already PPA-complete for two agents.
- For *monotone*[3] valuation functions, the problem can be solved efficiently for two agents, but is PPA-complete for three agents.

In this model, the number of agents n is assumed to be constant and the valuations are given in the input as circuits or Turing machines that allow efficient evaluation.[4] The reductions establishing PPA-hardness also immediately yield corresponding exponential lower bounds on the number of evaluation queries needed to find a solution. All of these hardness results continue to hold even if one allows approximate solutions where $|v_i(I^+) - v_i(I^-)| \leq \varepsilon$ for some $\varepsilon > 0$. In fact, they hold for any constant $\varepsilon < 1$, thus ruling out any efficient algorithm for approximating the problem in that way.[5]

For concreteness, let us focus on the setting with two agents which is PPA-complete for any $\varepsilon < 1$, as mentioned above. Faced with such a strong negative result, it is natural to attempt to relax the problem in a different way: instead of only allowing $n = 2$ cuts, what if we allow some larger constant number of cuts k? A positive result for some reasonable number of cuts, say $k = 5$, would indicate that the problem is in fact easy, and that the hardness only stems from the strict requirement on the number of cuts allowed.

An inspection of the proof of existence yields the following interesting observation. If one allows k cuts instead of n, where $k > n$, then the following "weaker" higher to lower dimensional version of the Borsuk-Ulam theorem suffices to prove existence.

Weak Borsuk-Ulam Theorem (High to Low)**.** Let $k > n$. Every continuous odd function $f : \mathcal{S}^k \to \mathbb{R}^n$ admits a zero.

Note that this weaker version indeed immediately follows from the standard Borsuk-Ulam theorem, since one can extend an odd function $f : \mathcal{S}^k \to \mathbb{R}^n$ to an odd function $f' : \mathcal{S}^k \to \mathbb{R}^k$ by just adding $k - n$ additional components that are identically zero. However, it is very much unclear how to reduce from the original version to this weaker one. In fact, using the general framework in the work of Deligkas et al. [DFRH22], it is not

[1] To be more precise, we consider the problem of finding an approximate solution; together with an assumption of Lipschitz-continuity for the function, this guarantees the existence of rational solutions with bounded bit complexity, that can be verified efficiently.

[2] In particular, various cryptographic lower bounds are known for these classes [Jeř16], [BPR15], [CHK+19], [JKKZ21].

[3] A valuation function v is monotone, if $v(A) \leq v(B)$ whenever $A \subseteq B$.

[4] In another commonly studied model, there are many agents and the valuations are assumed to be additive and have piecewise constant densities, which can be given explicitly in the input [FRG23]. This model is incomparable to the one we consider here and, in particular, it has no implications on the query complexity of the problem.

[5] Given the usual normalization assumption that the valuations output values in $[0, 1]$, any partition is trivially a 1-approximate solution. We note that the hardness results for constant $\varepsilon < 1$ make use of the fact that the Lipschitz constant of the valuation functions can be exponentially large in the representation of the problem. A straightforward rescaling of the valuation functions yields hardness for constant Lipschitzness and inverse-exponential ε.

too hard to establish that this weaker version of Borsuk-Ulam is computationally equivalent to our n-agent k-cut version of consensus halving. Thus, there is essentially no way around it: one must understand the complexity of this weaker theorem in order to make progress on consensus halving.

Instead of allowing additional cuts, or perhaps in addition to that, another very reasonable way to relax the consensus halving problem is to only require that some of the agents, instead of all them, are satisfied with the partition. Namely, for some $s < n$, we would be happy with a solution which ensures that s out of the n agents believe that I^+ and I^- have the same value (or approximately the same value). Importantly, here we do not specify which subset of s agents should be satisfied; otherwise, this would just boil down to the first type of relaxation introduced above. The fact that an algorithm can choose to satisfy any subset of s agents is what makes this problem potentially easier. As above, it turns out that this relaxation is equivalent to the following weakening of Borsuk-Ulam.

Weak Borsuk-Ulam Theorem (s-out-of-n)**.** Let $s < n$ and $k \geq n$. Every continuous odd function $f : \mathcal{S}^k \to \mathbb{R}^n$ admits an s-out-of-n zero, i.e., a point $x \in \mathcal{S}^k$ such that $f_i(x) = 0$ for at least s distinct coordinates $i \in [n]$.

Two observations are in order. The first observation is that this "s-out-of-n" version can be viewed as a further weakening of the "High to Low" version above by only keeping the first s output coordinates of f and applying the previous theorem to $f : \mathcal{S}^k \to \mathbb{R}^s$. The second observation is that the condition $k \geq n$ is not needed for the existence of a solution; an s-out-of-n zero is guaranteed to exist as long as $s \leq \min\{k, n\}$, by the original Borsuk-Ulam theorem. Thus, it also makes sense to consider a setting where n is larger than k. For example, if n is large, but $k = s = 2$, this corresponds to another interesting relaxation: there are many agents, but we only require the algorithm to satisfy two of them, while using two cuts.

Our results. We prove that all these weaker versions of Borsuk-Ulam remain PPA-complete for any constant parameters. In fact, we prove the following result, which yields hardness for the two versions mentioned above as a special case.

Informal Theorem 1. *For any constants $k, n \geq 2$, it is* PPA-*complete to compute a 2-out-of-n zero of a continuous odd function $f : \mathcal{S}^k \to \mathbb{R}^n$.*

The hardness result continues to hold if one allows approximate zeroes. Note that the theorem immediately also implies PPA-hardness of finding s-out-of-n zeroes for $s > 2$, since any s-out-of-n zero is a 2-out-of-n zero. Existence of an s-out-of-n solution is guaranteed whenever $s \leq \min\{k, n\}$. Furthermore, finding a 1-out-of-n solution is always easy,[6] so the parameters of the theorem are optimal. An interesting mathematical consequence of our hardness result is that these weaker versions of Borsuk-Ulam are in fact not strictly weaker, but can be used to prove the original Borsuk-Ulam theorem.

Using our main theorem, we can prove the following result for Consensus Halving with general valuations.

Informal Theorem 2. *For any $k, n \geq 2$, it is* PPA-*complete to find a Consensus Halving solution with at most k cuts that satisfies at least two out of n agents with general valuations.*

In particular, this theorem has the following consequences:

- It is hard to find a consensus halving among two agents, even if we allow 50 cuts.
- The problem remains hard even if there are 50 agents, we allow 30 cuts, and only need to make sure that we satisfy two agents.

The hardness result continues to hold for ε-approximate solutions, for any $\varepsilon < 1$.

Moving on to agents with monotone valuations, it is known that for two agents a consensus halving with two cuts can be found efficiently [DFRH22]. Our next result proves than any version of the problem that requires satisfying three agents is hard.

Informal Theorem 3. *For any $k, n \geq 3$, it is* PPA-*complete to find a Consensus Halving solution with at most k cuts that satisfies at least three out of n agents with* monotone *valuations.*

Again, these intractability results continue to hold for ε-approximate solutions, for any $\varepsilon < 1$. For monotone and *additive* valuations, Alon and Graur [AG21] have shown that an ε-approximate solution can be found efficiently using at most $O(n \log(1/\varepsilon))$ cuts. Our result shows that no such efficient algorithm exists for monotone *non-additive* valuations, unless the class PPA collapses to polynomial-time.[7]

We also obtain similar results for a generalized ham sandwich problem. Namely, instead of measures, we consider general set functions with appropriate continuity assumptions. For the version where we also require the set functions to be monotone, we obtain the following result.

Informal Theorem 4. *For any $d, n, k \in \mathbb{N}$ with $\min\{d, n\} \geq 3$, it is* PPA-*complete to find a 3-out-of-n solution to Ham Sandwich in $\mathbb{R}^d$ that uses at most k hyperplanes.*

Note that here we allow k hyperplanes instead of just a single one, and we relax the solution to be 3-out-of-n, i.e., we only have to bisect three of the n provided set functions. Some consequences of this theorem for the generalized Ham Sandwich problem:

- It is hard to bisect a ham sandwich with three ingredients, even in dimension 30.
- Furthermore, the problem remains hard even if we are allowed to cut 50 times with our knife.

[6] Pick an arbitrary point x on the sphere. If $f_1(x) \neq 0$, then $f_1(x)$ and $f_1(-x)$ have opposite signs and doing binary search on a curve connecting the two points will yield an approximate solution with any desired precision.

[7] Our results show that the additivity assumption is also crucial in the work of Goldberg and Li [GL23]. In particular, they show that the problem lies in PPA-3 with additional cuts. Given our PPA-hardness result, and the fact that PPA and PPA-3 are believed to be incomparable classes [GKSZ20], [Hol21], their result is unlikely to extend to non-additive valuations.

These results also hold for approximate solutions.

A. Technical Overview

In this section we give an overview of our proof techniques and how they relate to prior work. As in our actual proof, here we will also in fact work with the discrete analog of Borsuk-Ulam, namely Tucker's lemma.

Tucker's lemma is stated on a discrete n-dimensional grid where each grid point is labeled by one of the following $2n$ possible labels $\{+1, -1, +2, -2, \ldots, +n, -n\}$. Let $[-N : N]$ denote $\{-N, -(N-1), \ldots, 0, \ldots, N\}$. We say that a labeling $\ell : [-N : N]^n \to \{\pm 1, \ldots, \pm n\}$ is antipodal if $\ell(x) = -\ell(-x)$ for all x lying on the boundary of the grid $[-N : N]^n$, i.e., for all x such that $x_i \in \{-N, N\}$ for some $i \in [n]$.

Tucker's Lemma ([Tuc46]). Every antipodal labeling $\ell : [-N : N]^n \to \{\pm 1, \ldots, \pm n\}$ has two adjacent points with equal and opposite labels (i.e., x, y with $\|x - y\|_\infty = 1$ and $\ell(x) = -\ell(y)$).

The Borsuk-Ulam theorem can be proved by using Tucker's lemma, and vice-versa. This equivalence continues to hold in the computational world, as it is indeed not too hard to reduce between the two problems in polynomial-time. This means that it suffices to prove our hardness results for "High to Low" versions of Tucker, in order to obtain the corresponding results for Borsuk-Ulam.

More concretely, for any $k \geq 2$, PPA-hardness of finding a zero of a Borsuk-Ulam function $f : \mathcal{S}^k \to \mathbb{R}^2$ follows from the following result.

Hardness of Tetrachromatic Tucker. Given a circuit computing an antipodal labeling $\ell : [-N : N]^k \to \{\pm 1, \pm 2\}$, it is PPA-hard to find two adjacent points with equal and opposite labels.

Note that even though the domain has dimension k, the labeling is tetrachromatic: it can only output one of four labels $+1, -1, +2, -2$. Our goal in this overview is to provide some intuition about how to prove this hardness result. For $k = 2$, this problem corresponds to the standard 2D-Tucker problem, which is known to be PPA-complete [Pap94], [ABB20]. Thus, the first interesting case is when k is equal to 3. We will mostly focus on this case in this overview.

Before proceeding further, we modify the definition of our Tucker problem to make it more convenient to work with. Namely, we consider a hybrid version: the labels remain discrete as they are, but we make the domain continuous. Namely, we consider an antipodal labeling $\ell : \mathcal{B}^k \to \{\pm 1, \pm 2\}$, where $\mathcal{B}^k$ denotes the unit ball in $\mathbb{R}^d$ according to the ℓ_2 norm. A solution consists of two points that are sufficiently close (ε-close for some small ε) and that have equal and opposite labels. This version is easily seen to be computationally equivalent to the corresponding version defined on the grid. See Figure 1 for an example of a 2D-Tucker instance with continuous domain.

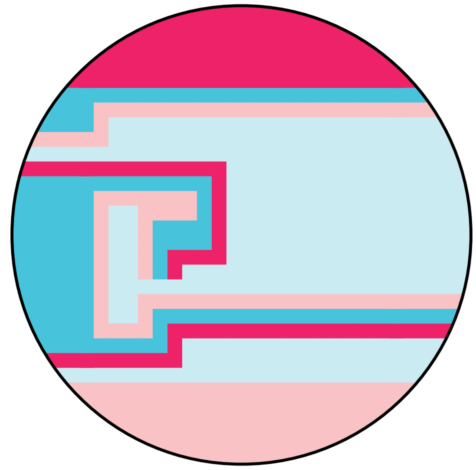

Fig. 1. A Tucker instance with continuous domain $\mathcal{B}^2$. The labels are represented as follows: dark blue $\to +1$, light blue $\to -1$, dark red $\to +2$, light red $\to -2$.

A reduction that uses additional labels. We would like to reduce from 2D-Tucker to the 3D Tetrachromatic version of Tucker. There is a simple way to reduce from 2D-Tucker to 3D-Tucker, where the labels $+3$ and -3 are allowed. We mention this next, because it is quite instructive.

Recall that 2D-Tucker simply consists of a labeling $\ell : \mathcal{B}^2 \to \{\pm 1, \pm 2\}$. A very natural way to embed $\mathcal{B}^2$ into $\mathcal{B}^3$ is to simply embed it into the horizontal slice going through the center of $\mathcal{B}^3$. Thus, we define the labeling on this slice according to ℓ and make the slice a bit thicker so that points above the slice are more than ε away from points below the slice. Finally, we label all points above the slice by $+3$ and all points below the slice by -3. It is not hard to check that this labeling is antipodal and that any solution has to occur in the middle slice and has to correspond to a solution of the original 2D-Tucker instance.

Here, of course, we have cheated, because we have used the additional labels $+3$ and -3. So the challenge can be summarized as:

Can we reduce from two dimensions to three, without introducing additional labels?

Perhaps surprisingly, this turns out to be highly non-trivial. Let us provide some intuition for why this is the case. Consider a 3D Tetrachromatic Tucker instance, namely an antipodal labeling $\ell : \mathcal{B}^3 \to \{\pm 1, \pm 2\}$. If we restrict the labeling to the horizontal slice going through the origin, we obtain an antipodal labeling on $\mathcal{B}^2$ that only uses labels $\{\pm 1, \pm 2\}$, i.e., a 2D-Tucker instance. Thus, there must exist a solution in this slice. Similarly, we could have chosen any other slice going through the origin and restricted the labeling to that slice. Again, the restricted labeling would be antipodal and thus guaranteed to contain a solution. Going even further, we can imagine continuously distorting the shape of the slice inside the ball, without moving the points where the boundary of this distorted surface touches the boundary of the ball. Again,

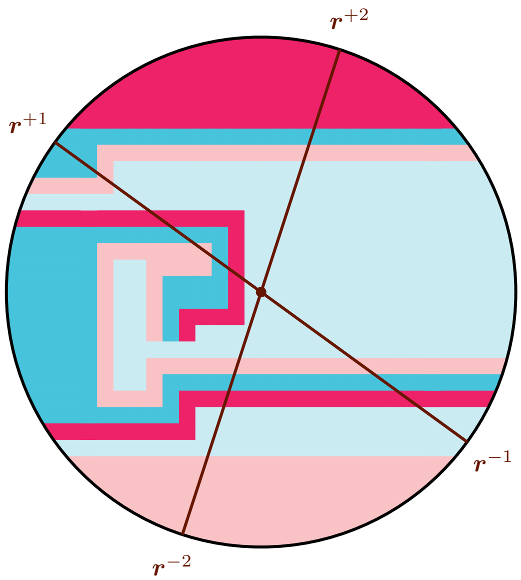

Fig. 2. The four points $\boldsymbol{r}^{+1}, \boldsymbol{r}^{-1}, \boldsymbol{r}^{+2}, \boldsymbol{r}^{-2}$ on the boundary of the 2D-Tucker instance.

restricting the labeling to this surface will give an antipodal labeling on a domain that is equivalent to $\mathcal{B}^2$, and thus this distorted surface must also contain a solution. As a special case of this, the lower and upper hemisphere of the ball must also each contain a solution. The take-away is that there are many solutions and it is not clear how to make it hard to find one.

Our approach. Our proof starts off by embedding a 2D-Tucker instance on the top hemisphere of the three-dimensional ball. This also defines the labeling on the bottom hemisphere by antipodality. Thus, at this point we have an antipodal labeling that is defined on the surface of the sphere, and such that any solution yields a solution to the original 2D-Tucker labeling. The challenge is that we have to extend the definition of the labeling to the interior of the ball in such a way that any solution we introduce can be used to find a solution in the original 2D-Tucker labeling.

We proceed as follows. Consider the original 2D-Tucker instance and find four points $\boldsymbol{r}^{+1}, \boldsymbol{r}^{-1}, \boldsymbol{r}^{+2}, \boldsymbol{r}^{-2}$ lying on the boundary of the domain $\mathcal{B}^2$ such that for $i = 1, 2$:

1) $\boldsymbol{r}^{+i}$ and $\boldsymbol{r}^{-i}$ are antipodal, and
2) the label of $\boldsymbol{r}^{+i}$ is $+i$, and the label of $\boldsymbol{r}^{-i}$ is $-i$.

It is easy to check that there is an efficient binary search procedure that either outputs a solution, or four points that satisfy these conditions. See Figure 2 for an illustration of the four special points on the boundary.

Now, consider the following attempt at defining the labeling in the interior of the ball $\mathcal{B}^3$. Consider any point $\boldsymbol{x}$ on the boundary of the ball, and recall that the labeling is defined there. Let $j \in \{\pm 1, \pm 2\}$ denote the label of $\boldsymbol{x}$. Now, consider the radius $[0, \boldsymbol{x}]$. This is the segment going from the center of the ball to the point $\boldsymbol{x}$. Define the labeling on this radius to be equal to the labeling on the radius $[0, \boldsymbol{r}^j]$ of the 2D-Tucker instance. Since by definition the label of $\boldsymbol{r}^j$ in the 2D instance is j, this is consistent with the label at $\boldsymbol{x}$ being j.

If we do this for every point $\boldsymbol{x}$ on the boundary of the ball, we obtain a labeling of the whole ball. In particular, note that all the radii agree on the label to be assigned to the origin: this will just be the label of the origin in the 2D-Tucker instance. Now, consider a point $\boldsymbol{x}$ on the boundary of the ball such that all points in the vicinity of $\boldsymbol{x}$ on the boundary have the same label j as $\boldsymbol{x}$. In that case, all the radii corresponding to these points will be labeled in exactly the same way. Thus, any solution occurring close to the radius $[0, \boldsymbol{x}]$ must be a solution that already occurs in the radius $[0, \boldsymbol{r}^j]$ of the original 2D-Tucker instance.

But what if the point $\boldsymbol{x}$ is close to a point $\boldsymbol{y}$ on the boundary which has a different label? If $\boldsymbol{y}$ has the opposite label, i.e., $-j$, then we claim that this is not an issue. Indeed, the radius $[0, \boldsymbol{y}]$ will be labeled according to the radius $[0, \boldsymbol{r}^{-j}]$ of the 2D-Tucker instance. Thus, some new solutions could occur inside the ball because the radii $[0, \boldsymbol{x}]$ and $[0, \boldsymbol{y}]$ are close to each other, but labeled differently. However, we can easily obtain $\boldsymbol{x}$ and $\boldsymbol{y}$ from such a solution, and these two points yield a solution to the original 2D-Tucker instance. Indeed, recall that the boundary of the ball was labeled according to the 2D-Tucker instance.

But if $\boldsymbol{y}$ has a label that is neither j nor $-j$, then spurious solutions can occur. Indeed, in that case the radii $[0, \boldsymbol{x}]$ and $[0, \boldsymbol{y}]$ are labeled differently, so new solutions can occur inside the ball, but $\boldsymbol{x}$ and $\boldsymbol{y}$ do not yield a solution to the 2D-Tucker instance.

How to handle non-solution color switches. In order to resolve this issue we use the following idea. In any region on the boundary of the ball where two colors meet but do not introduce a solution, say colors $+1$ and $+2$, we modify the construction locally. Instead of using radius $[0, \boldsymbol{r}^{+1}]$ or radius $[0, \boldsymbol{r}^{+2}]$ depending on the label of the point on the boundary, we continuously interpolate between $\boldsymbol{r}^{+1}$ and $\boldsymbol{r}^{+2}$ to obtain a point $\boldsymbol{r}'$ lying between $\boldsymbol{r}^{+1}$ and $\boldsymbol{r}^{+2}$ on the boundary of the 2D instance, and use the radius $[0, \boldsymbol{r}']$. In other words, we now make use of all the radii in the 2D instance lying between $\boldsymbol{r}^{+1}$ and $\boldsymbol{r}^{+2}$. More generally, depending on which two colors are meeting, we will use the corresponding "quarter" sector of the 2D instance. For any two colors that do not constitute a solution, there is a corresponding quarter sector in the original 2D instance that we can use.[8]

To give some more intuition, let us consider a point $\boldsymbol{z}$ on the boundary of the ball, and imagine that we can move this point continuously on the surface of the ball and observe how the radius that we use to label the points $[0, \boldsymbol{z}]$ changes. At the beginning, $\boldsymbol{z}$ lies in a region where all nearby points, including $\boldsymbol{z}$, are labeled $+1$. In that case, the radius $[0, \boldsymbol{z}]$ is labeled according to how the radius $[0, \boldsymbol{r}(\boldsymbol{z})]$ is labeled in

[8] A subtle observation is that this construction modifies the labeling on the boundary of the ball, when we are using a radius that was obtained by interpolation.

the 2D instance, where $\boldsymbol{r}(\boldsymbol{z}) = \boldsymbol{r}^{+1}$. Now, imagine that we continuously move $\boldsymbol{z}$ towards a region where the color on the boundary switches from $+1$ to $+2$. As $\boldsymbol{z}$ comes closer to that region, its radius $\boldsymbol{r}(\boldsymbol{z})$ will start changing from $\boldsymbol{r}^{+1}$ to $\boldsymbol{r}^{+2}$ in a continuous way. This will continue until $\boldsymbol{z}$ has crossed the color switch region to find itself in a region labeled $+2$. Once $\boldsymbol{z}$ is safely inside that region and sufficiently far away from the color switch, its radius will simply be $\boldsymbol{r}(\boldsymbol{z}) = \boldsymbol{r}^{+2}$, as before.

This construction has the following crucial property:

For any two points $\boldsymbol{x}$ and $\boldsymbol{y}$ on the boundary that are close to each other, but do not constitute a solution, the radii $[0, \boldsymbol{x}]$ and $[0, \boldsymbol{y}]$ are labeled according to two radii of the 2D instance that are close to each other.

This ensures that any solution that occurs inside the ball in that area will also yield a solution in the original 2D instance. Furthermore, the fact that our construction uses different radii from the same original 2D-Tucker instance is crucial to ensure that the instance we construct is consistent and that antipodality is satisfied. The technical details that make this approach work can be found in the full version of the paper.

Higher dimension. With the PPA-hardness of 3D Tetrachromatic Tucker established, the next step is to generalize this to kD Tetrachromatic Tucker for any $k \geq 3$. To achieve this, we generalize the construction of the 3D instance from the 2D instance in a recursive way. Namely, we create a chain of instances, where with each additional instance the dimension increases by one. In each step, we use the general idea described above. The construction and proof can be found in the full version.

2-out-of-m solutions. Next, we present a further strengthening of our result. Namely, we consider a multidimensional labeling: every point is assigned a label in $\{\pm 1\}^m$. In other words, in each dimension $i \in [m]$, the label can be either positive or negative. A 2-out-of-m solution consists of two points $\boldsymbol{x}$ and $\boldsymbol{y}$ that are close to each other, and such that there exist distinct $i, j \in [m]$ such that $\boldsymbol{x}$ and $\boldsymbol{y}$ do not agree on the sign of the ith label, and they also do not agree on the sign of the jth label.

For $m = 2$, this problem is equivalent to Tetrachromatic Tucker. Indeed, we can reinterpret the labels as follows: $(+1, +1) \to +1$, $(-1, -1) \to -1$, $(+1, -1) \to +2$, and $(-1, +1) \to -2$. It is easy to check that a 2-out-of-2 solution with respect to the two-dimensional labels exactly corresponds to a solution with respect to the standard labels.

We prove that the problem remains PPA-hard for any constant $m \geq 2$ and in any constant dimension $k \geq 2$. To prove this, we start with a Tetrachromatic 2D-Tucker instance and we show how to reduce it to a 2D instance that uses m-dimensional labels. Then, we apply the recursive construction mentioned above to obtain a k-dimensional hard instance. This corresponds to proving PPA-hardness of finding a 2-out-of-m zero of a Borsuk-Ulam function $f : \mathcal{S}^k \to \mathbb{R}^m$, for any $k, m \geq 2$. This is the culmination of our hardness results, in the sense that all other hardness results follow from this one.

Monotone Borsuk-Ulam and applications. Finally, in Section IV we present the consequences of these hardness results for our application problems. The main technical contribution in this section is the proof that it is PPA-hard to find a 3-out-of-m solution of a *monotone* Borsuk-Ulam function. We then use this result in a crucial way to obtain intractability for various relaxations of Consensus Halving and Ham Sandwich.

Comparison to prior work on PPAD-hardness of relaxations of Sperner. Sperner's lemma, which is the discrete analog of Brouwer's fixed point theorem, is known to be PPAD-complete [Pap94]. In this problem we are given a labeling of a triangulation of the k-dimensional simplex, and the goal is to find a simplex in the triangulation which contains all $k + 1$ labels. Such a simplex is guaranteed to exist by Sperner's lemma, under some simple boundary conditions. The recent work [GRRS24] studied a relaxation of kD Sperner where the goal is to find a point where three distinct labels meet, instead of the usual $k+1$ labels. They were able to show that this remains PPAD-hard, and used the result to obtain stronger intractability results for the envy-free cake cutting problem with very general utility functions.

Using their techniques, it is possible to establish PPAD-hardness of the Tetrachromatic kD Tucker problem, but new ideas are needed to show our PPA-completeness result.[9] In more detail, we make use of the following ideas from their work:

- Working with a hybrid version of the problem, where the domain is continuous, but the labels remain discrete.
- In a region where an unwanted solution would occur, embed a hard instance of the two-dimensional problem to hide the exact location of the solution, and force an algorithm to solve the 2D problem. This general idea has also been used in other prior work [FGHS22].
- Use a recursive construction to prove the hardness result for any dimension k.

To establish PPA-hardness we have to handle the following new challenges:

- In order to prove PPA-hardness, we have to construct instances that have "complicated" boundaries. Instances with "simple" boundaries lie in PPAD. In order to achieve this, we first define our instances everywhere on the boundary of the domain, and then find a way to extend the labeling in the interior.
- In order to extend the labeling to the interior without introducing unwanted solutions, we use the new idea of breaking down the original 2D instance into different pieces, and then carefully embedding the appropriate piece depending on the labels that we see on the boundary. This is in stark contrast to prior work, where the whole 2D instance is embedded.

[9] Recall that PPAD $\subseteq$ PPA, and oracle separations between the two classes are known [BCE+98].

- Embedding these pieces requires locally modifying the labeling on the boundary that we started with. This needs to be done very carefully, since we have to ensure that the labeling remains antipodal on the boundary. This is again in contrast with [GRRS24], where the labeling is first defined on a facet of the simplex and then extended to the whole simplex, but importantly this original labeling on the facet does not need to be modified.
- In order to prove PPA-hardness for our 2-out-of-m versions, we use a new idea for reducing from the usual tetrachromatic labels to m-dimensional labels in 2D instances.[10]

II. Preliminaries

We will use $[k]$ to denote the set $\{1, 2, \ldots, k\}$ for any positive integer k. We will use $[l : r]$ to denote the set $\{l, l+1, \ldots, r\}$ for any integers $l \le r$. We will use $\pm k$ or $\{\pm k\}$ to denote the set $\{k, -k\}$ of integers with absolute value of k.

We will use $\mathcal{B}^k[p]$ to denote the k-dimensional ℓ_p-ball, i.e., $\mathcal{B}^k[p] = \{\boldsymbol{x} \in \mathbb{R}^k : \|\boldsymbol{x}\|_p \le 1\}$. For simplicity, we will also use $\mathcal{B}^k$ for the standard ℓ_2-ball $\mathcal{B}^k[2]$. We will use $\mathcal{S}^{k-1}[p]$ to denote the sphere of the k-dimensional ℓ_p-ball, i.e., $\mathcal{S}^{k-1}[p] = \partial\mathcal{B}^k[p] = \{\boldsymbol{x} \in \mathbb{R}^k : \|\boldsymbol{x}\|_p = 1\}$, and also use $\mathcal{S}^{k-1}$ for $\mathcal{S}^{k-1}[2]$. We will use $(x)_+$ to denote $\max(0, x)$ and use $(x)_-$ to denote $\min(1, x)$. For short, we will use $(x)_{[0,1]}$ for $((x)_+)_-$.

The Complexity Class: TFNP**.** Search problems are defined via relations $R \subseteq \{0,1\}^* \times \{0,1\}^*$, where the goal is to find an output vector $\boldsymbol{y}$ for the input vector $\boldsymbol{x}$ such that $(\boldsymbol{x}, \boldsymbol{y}) \in R$, or to claim no such $\boldsymbol{y}$ exists. The class FNP consists of all such search problems in which R can be computed in polynomial time, which means that whether any $(\boldsymbol{x}, \boldsymbol{y}) \in R$ can be decided in polynomial time, and is polynomially balanced, which means that there exists a polynomial $p(n)$ such that $(\boldsymbol{x}, \boldsymbol{y}) \in R$ only if $|\boldsymbol{y}| \le p(|\boldsymbol{x}|)$. Here, $|\boldsymbol{x}|$ is defined as the number of bits in bitstring $\boldsymbol{x}$. The class TFNP consists of all FNP problems whose relation R is total, i.e., for any input $\boldsymbol{x}$, there always exists a vector $\boldsymbol{y}$ such that $(\boldsymbol{x}, \boldsymbol{y}) \in R$.

A polynomial-time reduction between two TFNP problems R, R' is defined by two polynomial-time functions $f : \{0,1\}^* \to \{0,1\}^*$ and $g : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that for any instance $\boldsymbol{x}$ of R, we can always reduce it to the instance $f(\boldsymbol{x})$ (of R') and then recover a solution from any output $\boldsymbol{y}$ of the instance for R', using $g(\boldsymbol{x}, \boldsymbol{y})$. That is, for any input $\boldsymbol{x}$ of R and any output $\boldsymbol{y}$ of R', we have

$$(f(\boldsymbol{x}), \boldsymbol{y}) \in R' \Longrightarrow (\boldsymbol{x}, g(\boldsymbol{x}, \boldsymbol{y})) \in R \ .$$

The complexity class: PPA**.** The class PPA consists of all TFNP problems that are polynomial-time reducible to the following Leaf problem.

Definition 1 (Leaf)**.** In Leaf, we are given an undirected graph G over vertex set $\{0,1\}^n$ with maximum degree 2, in which node 0^n has degree 1. The graph is represented by a Boolean circuit, which for each node outputs its neighbors. The goal of Leaf is to find another node $x \in \{0,1\}^n \setminus \{0^n\}$ with degree 1.

A standard result for PPA is that it captures the complexity of the following computational problem of Tucker's lemma [Tuc46].

Definition 2 (kD-Tucker)**.** In kD-Tucker, we are given a Boolean circuit computing[11] a function $C : [-N : N]^k \to \{\pm 1, \pm 2, \ldots, \pm k\}$ satisfying the following property on antipodal points: $C(\boldsymbol{x}) = -C(-\boldsymbol{x})$ for each $\boldsymbol{x}$ with some $x_i \in \{-N, N\}$. The goal of kD-Tucker is to find two points $\boldsymbol{x}^+, \boldsymbol{x}^- \in [-N : N]^k$ such that

1) **they are close to each other**, i.e., $\|\boldsymbol{x}^+ - \boldsymbol{x}^-\|_\infty \le 1$; and
2) **they have opposite colors**, i.e., $C(\boldsymbol{x}^+) = -C(\boldsymbol{x}^-)$.

Theorem II.1 ([Pap94], [ABB20])**.** *For any $k \ge 2$, kD-Tucker is* PPA*-complete.*

III. High-to-Low Tucker: Formulations and Results

In this section, we formally define our high-to-low versions of Tucker's lemma and state our main results.

Definition 3 (Tetrachromatic kD-Tucker)**.** In the Tetrachromatic kD-Tucker problem, we are given a Boolean circuit $C : [-N : N]^k \to \{\pm 1, \pm 2\}$ such that we have $C(\boldsymbol{x}) = -C(-\boldsymbol{x})$ for any $\boldsymbol{x}$ on the boundary (i.e., for any $\boldsymbol{x}$ with $x_i = -N$ or N for some $i \in [k]$), and we are asked to find two points $\boldsymbol{x}^+, \boldsymbol{x}^- \in [-N : N]^k$ such that

- **they are close to each other:** $\|\boldsymbol{x}^+ - \boldsymbol{x}^-\|_\infty \le 1$; and
- **they have opposite colors:** $C(\boldsymbol{x}^+) = -C(\boldsymbol{x}^-)$.

Our main result is the PPA-completeness of this problem. The proof can be found in the full version.

Theorem III.1. *For any constant $k \ge 2$, Tetrachromatic kD-Tucker is* PPA*-complete.*

Another natural variant of Tucker's lemma that we will use to obtain stronger hardness results for our applications in fair division problems is the following p-out-of-m version. In this version, we work with a more expressive coloring which is an m-dimensional indicator, with the ℓ-th entry indicating whether the ℓ-th label is positive or negative. Then, a p-out-of-m solution is a set of nearby points where opposite labels meet in at least p out of the m dimensions.

Definition 4 (p-out-of-m kD-to-mD-Tucker)**.** Suppose $p \le \min\{k, m\}$. In the p-out-of-m kD-to-mD-Tucker problem, we are given a size-poly(n) circuit $\boldsymbol{C} : [-N :$

[10]Unlike the 3-out-of-$k+1$ Sperner problem considered by [GRRS24], our 2-out-of-m versions allow for any $m \ge 2$, in particular for $m > k$. This requires the new idea mentioned here.

[11]The inputs to the circuit are represented in binary, so N can be exponential in the size of the instance.

$N]^k \to \{\pm 1\}^m$ such that we have $\boldsymbol{C}(\boldsymbol{x}) = -\boldsymbol{C}(-\boldsymbol{x})$ for any $\boldsymbol{x}$ on the boundary (i.e., we have $x_i = -N$ or N for some $i \in [k]$), and we are asked to find p points $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(p)} \in [-N : N]^k$ and a subset of indices $S \subseteq [m]$ with size at least p such that

- **they are close to each other:** for any $j, j' \in [p]$, $\|\boldsymbol{x}^{(j)} - \boldsymbol{x}^{(j')}\|_\infty \leq 1$; and
- **there exist opposite colors restricted to S:** for any $i \in S$, there exists $j, j' \in [p]$ such that $C_i(\boldsymbol{x}^{(j)}) = -C_i(\boldsymbol{x}^{(j')})$.

Remark 1. When $p = k = m = 2$, we have that 2-out-of-2 2D-TO-2D-TUCKER is equivalent to the 2D-TUCKER problem. This is easy to show by a simple renaming of the labels, e.g., $(+1, +1) \to +1$, $(-1, -1) \to -1$, $(+1, -1) \to +2$, and $(-1, +1) \to -2$.

We show that this 2-out-of-m version is also PPA-complete. The proof can be found in the full version.

Theorem III.2. *For any $k, m \geq 2$, 2-out-of-m kD-TO-mD-TUCKER is PPA-complete.*

IV. APPLICATIONS

In this section, we explore some consequences of our new hardness results for Tucker's lemma on two application problems: the Consensus Halving problem from fair division and the Ham Sandwich problem from computational geometry. In order to obtain results for these problems we first extend our new hardness results to the continuous version of Tucker's lemma, Borsuk-Ulam.

A. Borsuk-Ulam

For computational purposes, it is more convenient to work on the ℓ_∞-sphere, instead of the ℓ_2-sphere. Furthermore, for our applications it will be useful to have Borsuk-Ulam functions that are not only defined (and continuous) on the sphere, but on the whole ball. However, note that solutions are still required to lie on the sphere.

We are now ready to define our computational version of Borsuk-Ulam. We use the notation $\partial([-1,1]^{k+1})$ to denote the boundary of $[-1,1]^{k+1}$, i.e., the set of all points $x \in [-1,1]^{k+1}$ such that $|x_i| = 1$ for some i.

Definition 5. Let $k, n, s \in \mathbb{N}$ with $s \leq \min\{k, n\}$, and $\varepsilon \in [0, 1]$ be constants. An instance of kD-TO-nD-BORSUK-ULAM consists of a function[12] $F : [-1,1]^{k+1} \to [-1,1]^n$ that satisfies

- F is L-Lipschitz-continuous for some $L > 0$ given in the input,
- F is an odd function, i.e., $F(-x) = -F(x)$ for all $x \in [-1,1]^{k+1}$,
- $F(1, \ldots, 1) = (1, \ldots, 1)$ (normalization)

[12]Following [DFRH22], we assume that the function is given as a well-behaved arithmetic circuit that uses gates $+, -, \times, \max, \min$, and rational constants [FGHS22]. Furthermore, the function is promised to be odd and L-Lipschitz-continuous with the provided constant L. The hard instances we construct below will always satisfy all the promised properties.

The goal is to find a point $x \in \partial([-1,1]^{k+1})$ that is an ε-approximate s-out-of-n solution to the instance, i.e., such that there exists $S \subseteq [n]$ with $|S| = s$ such that $|F_i(x)| \leq \varepsilon$ for all $i \in S$.

As long as $s \leq \min\{n, k\}$, a solution is guaranteed to exist by the Borsuk-Ulam theorem. Since the problem with $n = k = s$ lies in PPA [Pap94], [ABB20] (see also [DFRH22] for this particular version), it follows that the weaker problem for any $k, n, s \in \mathbb{N}$ with $s \leq \min\{k, n\}$ also lies in PPA.

An instance of kD-TO-nD-BORSUK-ULAM is *monotone*, if

$$x \leq y \implies F(x) \leq F(y)$$

for all $x, y \in [-1,1]^{k+1}$, where "$a \leq b$" denotes "$a_i \leq b_i$ for all i."

Remark 2. If the function F is not odd, then it can be made odd by letting $F'(x) = (F(x) - F(-x))/2$. A solution x to the instance with F' will thus satisfy that $F(x)$ and $F(-x)$ are 2ε-close (in s out of the n coordinates).

1) From Tucker to Borsuk-Ulam: In this section, we directly transfer our new hardness results for Tucker to the corresponding hardness results for Borsuk-Ulam. The following result is tight, since finding a 1-out-of-n solution is easy for any $k, n \geq 1$ by using a simple binary search approach.

Theorem IV.1. *For any $k, n \geq 2$, and any $\varepsilon \in [0, 1)$, finding an ε-approximate 2-out-of-n solution to kD-TO-nD-BORSUK-ULAM is PPA-complete.*

Proof. We adapt the corresponding proof in [DFRH22, Theorem 4.4] to our more general setting. We reduce from 2-out-of-n kD-TO-nD-TUCKER, which is PPA-hard by Theorem III.2.

Consider a Tucker labeling $\ell : [-m : m]^k \to \{-1, 1\}^n$. Without loss of generality, we can assume that $\ell(m, \ldots, m) = (1, \ldots, 1)$. This is easy to ensure by modifying the labeling to $p \mapsto \ell(p) \odot \ell(m, \ldots, m)$, where $\odot$ denotes component-wise multiplication of vectors. Note that this gives a valid Tucker labeling with the same set of solutions.

The first step of the reduction is to interpolate the labeling ℓ into a continuous function $h : [-1,1]^k \to [-1,1]^n$ such that

1) h is antipodally antisymmetric, i.e., $h(-\boldsymbol{x}) = -h(\boldsymbol{x})$ for all $\boldsymbol{x} \in \partial[-1,1]^k$
2) h is $2km$-Lipschitz-continuous (with respect to the ℓ_∞-norm)
3) $h(1, \ldots, 1) = (1, \ldots, 1)$
4) Any point $\boldsymbol{x} \in [-1,1]^k$ such that $|h_i(\boldsymbol{x})| < 1$ for at least two distinct coordinates $i \in [n]$, yields a 2-out-of-n solution of the original Tucker labeling ℓ.

In more detail, we embed the grid $[-m : m]^k$ into $[-1,1]^k$ as the grid $\{-1, -(m-1)/m, \ldots, 0, \ldots, 1\}^k$. Then, we define h on these grid points so that it agrees with ℓ, i.e., $h(\boldsymbol{x}) := \ell(m \cdot \boldsymbol{x})$. Finally, we define h on the whole domain $[-1,1]^k$ by linear interpolation on a standard triangulation of the grid. For details, see, e.g., [DFRH22, Appendix C], where points 1 and 2 are also argued. Point 3 immediately follows from $\ell(m, \ldots, m) = (1, \ldots, 1)$. Finally, for point 4, note that if all vertices of a simplex agree on the ith label, then $|h_i(\boldsymbol{x})| = 1$ for all points $\boldsymbol{x}$ in the simplex. As a result, if $|h_i(\boldsymbol{x})| < 1$, then any simplex containing $\boldsymbol{x}$ must have two vertices with opposite labels in coordinate i. Thus, in point 4, the vertices of any simplex containing $\boldsymbol{x}$ constitute a 2-out-of-n solution to the Tucker labeling ℓ.

In the second and final step of the reduction, we construct a kD-TO-nD-BORSUK-ULAM function $F : [-1,1]^{k+1} \to [-1,1]^n$ as follows

$$F(\boldsymbol{x}', x_{k+1}) := \frac{1 + x_{k+1}}{2} \cdot h(\boldsymbol{x}') + \frac{1 - x_{k+1}}{2} \cdot (-h(-\boldsymbol{x}'))$$

for all $\boldsymbol{x} = (\boldsymbol{x}', x_{k+1}) \in [-1,1]^k \times [-1,1]$. Note that F is carefully constructed so as to be an odd function. Furthermore, it is easy to check that F is $(2km+1)$-Lipschitz-continuous and $F(1,\dots,1) = (1,\dots,1)$. Now consider a point $\boldsymbol{x} = (\boldsymbol{x}', x_{k+1}) \in \partial[-1,1]^{k+1}$ that is an ε-approximate 2-out-of-n solution of F, for some $\varepsilon < 1$. In other words, there exists $S \subseteq [n]$, $|S| = 2$, such that $|F_i(\boldsymbol{x})| \le \varepsilon < 1$ for all $i \in S$. Since F is odd, we can assume without loss of generality that $x_{k+1} \ge 0$. If $x_{k+1} = 1$, then $F(\boldsymbol{x}) = h(\boldsymbol{x}')$, and $\boldsymbol{x}'$ is a 2-out-of-n solution to h and thus ℓ. If, on the other hand, $x_{k+1} \in [0,1)$, then it must be that $\boldsymbol{x}' \in \partial[-1,1]^k$, since $\boldsymbol{x} \in \partial[-1,1]^{k+1}$. But in that case $h(\boldsymbol{x}') = -h(-\boldsymbol{x}')$ by antipodal symmetry of h, and thus $F(\boldsymbol{x}) = h(\boldsymbol{x}')$, as before. ☐

2) Monotonicity: In this section, we prove that our results continue to hold even if we require the Borsuk-Ulam function to be monotone, as long as we consider 3-out-of-n, instead of 2-out-of-n solutions. The monotonicity will be crucially used for our two application problems in the next two sections. Once again, the following theorem is tight, since a 2-out-of-n solution can be found efficiently for any $k, n \ge 2$ in the monotone case. This is not trivial, but follows from the fact that Consensus Halving with two agent with monotone valuations can be solved efficiently with at most two cuts; see Theorem IV.5 and the paragraph preceding it.

Theorem IV.2. *For any $k, n \ge 3$, and any $\varepsilon \in [0,1)$, finding an ε-approximate 3-out-of-n solution to* monotone kD-TO-nD-BORSUK-ULAM *is* PPA*-complete.*

Proof. In order to prove the statement, we use the corresponding proof in [DFRH22, Theorem 5.4] as a starting point. Here we require significant new ideas to extend their proof to our setting. The original proof embeds a k-dimensional Borsuk-Ulam instance into a k-dimensional diagonally-placed hyperplane in $(k+1)$-dimensional space. Then, the function is extended to cover the whole $(k+1)$-dimensional hypercube. It is shown that this can be done in a way that yields a monotone function. Finally, an additional output coordinate is added such that when the constructed Borsuk Ulam function is small in that output coordinate, the input point has to lie very close to the diagonal hyperplane where the original lower-dimensional instance was embedded.

Their setting corresponds to the case where $k = n$ and we are looking for an n-out-of-n solution. The extension to the case where k is different from n is relatively straightforward. The challenging point is making this work for 3-out-of-n solution (or, in fact, already just $(n-1)$-out-of-n solutions.) The crucial problem is that the trick of adding an additional output coordinate does not work anymore, because we are not guaranteed to obtain a solution where the function is small in this output coordinate. As a result, we use a different idea with a more careful extension of the function on the diagonal hyperplane to the whole $(k+1)$-dimensional hypercube. At a high level, we extend the function in a staggered manner, treating one output coordinate after the other, as we move further away from the diagonal hyperplane. This ensures that we can recover a 2-out-of-n solution of the original lower-dimensional function from a 3-out-of-n solution to the constructed instance.

Let $k, n \ge 3$, and $\varepsilon \in [0,1)$ be given. First of all, note that it suffices to prove the hardness result for some particular ε, say $\varepsilon = 1/2$. Indeed, if we have a monotone Borsuk-Ulam function $H : [-1,1]^{k+1} \to [-1,1]^n$ for which it is PPA-hard to find a $1/2$-approximate 3-out-of-n solution, then we can easily obtain a monotone Borsuk-Ulam function H' for which it is PPA-hard to find an ε-approximate 3-out-of-n solution for any $\varepsilon \in [0,1)$. Indeed, it suffices to construct the function $H' : [-1,1]^{k+1} \to [-1,1]^n$, $H'_i(\boldsymbol{x}) = \mathsf{T}_{[-1,1]}(2 \cdot H_i(\boldsymbol{x}))$ for all $i \in [n]$. Here $\mathsf{T}_{[-1,1]} : \mathbb{R} \to [-1,1]$ denotes truncation (or, equivalently, projection) to $[-1,1]$. Thus, from now on we just assume that $\varepsilon = 1/2$.

We reduce from the problem of finding a $3/4$-approximate 2-out-of-n solution to kD-TO-nD-BORSUK-ULAM, which is PPA-hard by Theorem IV.1. Let $F : [-1,1]^{k+1} \to [-1,1]^n$ be the input function. Recall that F is odd and L-Lipschitz for some $L \ge 1$ given in the input.

The function F is not motonone and so our goal is to construct a Borsuk-Ulam function $H : [-1,1]^{k+1} \to [-1,1]^n$ that is monotone, and such that any solution to H yields a solution to F. At a high level, we are going to appropriately restrict the function F on a k-dimensional hyperplane of $[-1,1]^{k+1}$, and then define H by carefully extending to the whole $(k+1)$-dimensional domain.

In the first step, we scale and restrict the function $F : [-1,1]^{k+1} \to [-1,1]^n$ into the following k-dimensional hyperplane in $(k+1)$-dimensional space

$$D = \left\{ \boldsymbol{x} \in \mathbb{R}^{k+1} : \sum_{i=1}^{k+1} x_i = 0 \right\}.$$

Namely, we define the function $G : D \to [-1,1]^n$ as

$$G(\boldsymbol{x}) = F\left(\mathsf{T}_{[-1,1]}(2 \cdot x_1), \dots, \mathsf{T}_{[-1,1]}(2 \cdot x_{k+1})\right).$$

Claim 1. *The function G has the following properties.*

1) *G is odd.*
2) *G is L_G-Lipschitz-continuous for $L_G = 2L$.*
3) *Any point $\boldsymbol{x} \in D$ that is a solution to G yields a $3/4$-approximate 2-out-of-n solution of the original Borsuk-Ulam function F. We say that a point $\boldsymbol{x} \in D$ is a solution to G, if it satisfies the following two properties:*

(i) There exists $j \in [k+1]$ such that $|x_j| \geq 1/2$, and
(ii) There exist two distinct coordinates $i_1, i_2 \in [n]$ such that $|G_i(\boldsymbol{x})| \leq 3/4$ for $i \in \{i_1, i_2\}$.

Proof. The fact that G is an odd function immediately follows from the fact that F and $\mathsf{T}_{[-1,1]}$ are odd. Since F is L-Lipschitz and $\mathsf{T}_{[-1,1]}$ is 1-Lipschitz, it follows that G is $2L$-Lipschitz. Now consider a point $\boldsymbol{x} \in D$ that is a solution to G as defined above. Let $\boldsymbol{x}' \in [-1,1]^{k+1}$ be defined as $\boldsymbol{x}' = (\mathsf{T}_{[-1,1]}(2 \cdot x_1), \ldots, \mathsf{T}_{[-1,1]}(2 \cdot x_{k+1}))$, and note that $G(\boldsymbol{x}) = F(\boldsymbol{x}')$. We will show that $\boldsymbol{x}'$ is a 3/4-approximate 2-out-of-n solution of the original Borsuk-Ulam function F. First of all, by condition (ii) we immediately obtain that $|F(\boldsymbol{x}')| \leq 3/4$ for $i \in \{i_1, i_2\}$. Thus, it only remains to show that $\boldsymbol{x}'$ lies on the boundary of $[-1,1]^{k+1}$. By condition (i), we have that there exists $j \in [k+1]$ with $|x_j| \geq 1/2$. But this implies that $|x'_j| = 1$, and thus $\boldsymbol{x}'$ lies on the boundary of $[-1,1]^{k+1}$, as desired. ☐

In the second step, we define the Borsuk-Ulam function $H : [-1,1]^{k+1} \to [-1,1]^n$ as follows

$$H_i(\boldsymbol{x}) = \mathsf{T}_{[-1,1]}\Big(G_i(\Pi(\boldsymbol{x})) + c_i\left(S(\boldsymbol{x})\right)\Big)$$

for all $i \in [n]$. Here $S(\boldsymbol{x}) = \sum_{i=1}^{k+1} x_i$ and $\Pi : [-1,1]^{k+1} \to D$ denotes the projection onto D, i.e.,

$$\Pi(\boldsymbol{x}) = \boldsymbol{x} - \frac{S(\boldsymbol{x})}{k+1} \cdot \mathbf{1}_{k+1}.$$

For each $i \in [n]$, the function $c_i : [-(k+1), k+1] \to \mathbb{R}$ is defined as

$$c_i(r) = \begin{cases} -L_G \cdot i \cdot d + M(r + i \cdot d) & \text{if } r \leq -i \cdot d \\ L_G \cdot r & \text{if } |r| \leq i \cdot d \\ L_G \cdot i \cdot d + M(r - i \cdot d) & \text{if } r \geq i \cdot d \end{cases}$$

where $d = 1/4nL_G$ and $M = 8nL_G$. Recall that L_G is the Lipschitz constant of G. Note that c_i is M-Lipschitz, since $M \geq L_G$.

Claim 2. *H has the following properties.*

1) *H is an odd function.*
2) *H is $16n(k+1)L_G$-Lipschitz-continuous.*
3) *$H(1, \ldots, 1) = (1, \ldots, 1)$.*
4) *H is monotone.*
5) *Any point $\boldsymbol{x} \in \partial[-1,1]^{k+1}$ such that $|H_i(\boldsymbol{x})| \leq 1/2$ for at least three distinct coordinates $i \in [n]$, yields a solution to G, and thus to the original Borsuk-Ulam instance.*

Proof. It is not hard to see that H is an odd function, since $\mathsf{T}_{[-1,1]}$, G_i, Π, c_i and S are all odd. One can verify that $\mathsf{T}_{[-1,1]}$ is 1-Lipschitz, G_i is L_G-Lipschitz, c_i is M-Lipschitz, S is $(k+1)$-Lipschitz, and Π is 2-Lipschitz. As a result, H is Lipschitz-continuous with Lipschitz constant $2L_G + (k+1)M \leq 2(k+1)M = 16n(k+1)L_G$.

Furthermore, we have that for $\boldsymbol{x} = (1, \ldots, 1)$ and all $i \in [n]$

$$\begin{aligned} G_i(\Pi(\boldsymbol{x})) + c_i(S(\boldsymbol{x})) &\geq -1 + c_i(k+1) \\ &\geq -1 + M(k+1-nd) \\ &\geq Mk - 1 \geq 1 \end{aligned}$$

since $nd \leq 1/4L_G \leq 1/4$ and $k+1 \geq 1/4 \geq nd \geq i \cdot d$. As a result, $H_i(1, \ldots, 1) = (1, \ldots, 1)$.

In order to prove that H is monotone, it suffices to prove that for any $\boldsymbol{x} \in [-1,1]^{k+1}$ and for any $i \in [n]$, $j \in [k+1]$, and $t \geq 0$ such that $\boldsymbol{x} + t \cdot e_j \in [-1,1]^{k+1}$ we have

$$G_i(\Pi(\boldsymbol{x} + t \cdot e_j)) + c_i(S(\boldsymbol{x} + t \cdot e_j)) \geq G_i(\Pi(\boldsymbol{x})) + c_i(S(\boldsymbol{x})). \tag{1}$$

Indeed, consider any $\boldsymbol{x}, \boldsymbol{y} \in [-1,1]^{k+1}$ with $x \leq y$. Then we can create a monotone path from $\boldsymbol{x}$ to $\boldsymbol{y}$ that only travels along cardinal directions and apply (1) on each straight portion of the path.

It remains to prove (1). We have

$$\begin{aligned} &G_i(\Pi(\boldsymbol{x} + t \cdot e_j)) + c_i(S(\boldsymbol{x} + t \cdot e_j)) \\ &= G_i(\Pi(\boldsymbol{x}) + \Pi(t \cdot e_j)) + c_i(S(\boldsymbol{x}) + t) \\ &\geq G_i(\Pi(\boldsymbol{x})) - L_G\|\Pi(t \cdot e_j)\|_\infty + c_i(S(\boldsymbol{x})) + L_G \cdot t \\ &\geq G_i(\Pi(\boldsymbol{x})) + c_i(S(\boldsymbol{x})) + L_G(t - \|\Pi(t \cdot e_j)\|_\infty) \\ &\geq G_i(\Pi(\boldsymbol{x})) + c_i(S(\boldsymbol{x})) \end{aligned}$$

where we used the fact that $c_i(r+t) \geq c_i(r) + L_G \cdot t$, since $M \geq L_G$, as well as the fact that $\|\Pi(t \cdot e_j)\|_\infty \leq t(1 - 1/(k+1)) \leq t$.

Finally, we prove the fifth point. Consider a point $\boldsymbol{x} \in \partial[-1,1]^{k+1}$ such that $|H_i(\boldsymbol{x})| \leq 1/2$ for $i \in \{i_1, i_2, i_3\} \subseteq [n]$, where $i_1 < i_2 < i_3$. Without loss of generality, we assume that $S(\boldsymbol{x}) \geq 0$. If this is not the case, just use $-\boldsymbol{x}$ instead of $\boldsymbol{x}$.

We first argue that $|S(\boldsymbol{x})| \leq (i_1 + 1) \cdot d$. Assume towards a contradiction that $S(\boldsymbol{x}) > (i_1 + 1) \cdot d$. Then, we have that

$$\begin{aligned} c_{i_1}(S(\boldsymbol{x})) &\geq L_G \cdot i_1 \cdot d + M(S(\boldsymbol{x}) - i_1 \cdot d) \\ &\geq L_G \cdot i_1 \cdot d + M \cdot d \\ &\geq M \cdot d = 2 \end{aligned}$$

and as a result

$$\begin{aligned} H_{i_1}(\boldsymbol{x}) &= \mathsf{T}_{[-1,1]}\Big(G_{i_1}(\Pi(\boldsymbol{x})) + c_{i_1}\left(S(\boldsymbol{x})\right)\Big) \\ &\geq \mathsf{T}_{[-1,1]}(-1 + 2) = 1 > 1/2 \end{aligned}$$

a contradiction. Next, we argue that $|G_{i_2}(\Pi(\boldsymbol{x}))| \leq 3/4$ and $|G_{i_3}(\Pi(\boldsymbol{x}))| \leq 3/4$. Since $|S(\boldsymbol{x})| \leq (i_1 + 1) \cdot d \leq i_2 \cdot d$, we have that

$$|c_{i_2}(S(\boldsymbol{x}))| = |L_G \cdot S(\boldsymbol{x})| \leq L_G(i_1 + 1)d \leq nL_G d = 1/4.$$

As a result, given that we have $|H_{i_2}(\boldsymbol{x})| \leq 1/2$ and this implies that $|G_{i_2}(\Pi(\boldsymbol{x})) + c_{i_2}(S(\boldsymbol{x})| \leq 1/2$, it follows that $|G_{i_2}(\Pi(\boldsymbol{x}))| \leq 3/4$. A very similar argument shows that $|G_{i_3}(\Pi(\boldsymbol{x}))| \leq 3/4$.

It remains to show that $\Pi(\boldsymbol{x})$ satisfies condition (i) in order to be a solution to G. Since $\boldsymbol{x} \in \partial[-1,1]^{k+1}$, there exists $j \in [k+1]$ such that $|x_j| = 1$. At the same time, we have shown above that $|S(\boldsymbol{x})| \leq (i_1+1)\cdot d \leq nd$. As a result, it follows that

$$|[\Pi(\boldsymbol{x})]_j| = \left| x_j - \frac{S(\boldsymbol{x})}{k+1} \right| \geq 1 - \frac{nd}{k+1} = 1 - \frac{1}{4(k+1)L_G} \geq \frac{1}{2}$$

where we used $L_G = 2L \geq 1$. As a result, $\Pi(\boldsymbol{x})$ is a solution to G, and thus it yields a solution to the original Borsuk-Ulam instance. □

This completes the proof of the Theorem. □

B. Consensus-Halving

In this section we present our results for the Consensus Halving problem. We use an existing reduction from Borsuk-Ulam to Consensus Halving [DFRH22] that we generalize appropriately to the setting with additional cuts. We begin with the definition of the problem.

A valuation function over the continuous resource $I = [0,1]$ is a function $v : \Lambda([0,1]) \to [0,1]$, where $\Lambda([0,1])$ is the set of Lebesgue-measurable subsets of $[0,1]$. We say that a valuation v is L-Lipschitz-continuous, if for all $A, B \in \Lambda([0,1])$, $|v(A) - v(B)| \leq L \cdot \lambda(A \Delta B)$, where $A \Delta B := (A \setminus B) \cup (B \setminus A)$ denotes the symmetric difference, and λ denotes the Lebesgue measure over $[0,1]$. We always assume that valuation functions are normalized, i.e., $v(\emptyset) = 0$. Finally, we say that the valuation v is monotone, if $v(A) \leq v(B)$ whenever $A \subseteq B$.

Definition 6. Let $n, k, s \in \mathbb{N}$ with $s \leq \min\{k, n\}$, and $\varepsilon \in [0,1]$ be constants. An instance of CONSENSUS-HALVING with n agents consists of valuation functions[13] $v_1, \ldots, v_n$ that are L-Lipschitz-continuous for some $L > 0$ given in the input. The goal is to find a partition (I^+, I^-) of $[0,1]$ that is an ε-approximate solution with at most k cuts that satisfies at least s out of n agents in the instance, i.e.,

1) the partition (I^+, I^-) of $[0,1]$ can be obtained by using at most k cuts, and
2) there exists $S \subseteq [n]$ with $|S| = s$ such that $|v_i(I^+) - v_i(I^-)| \leq \varepsilon$ for all $i \in S$.

If the valuations $v_1, \ldots, v_n$ are monotone, then we say that the CONSENSUS-HALVING instance is monotone.

We are now ready to state our results for Consensus Halving. First, note that since the original problem with $n = k = s$ is known to lie in PPA (see, e.g., [DFRH22, Proposition 3.2]), this weaker version also lies in PPA. The following first PPA-completeness result is tight, since a simple binary search suffices to find a solution that satisfies 1-out-of-n agents.

Theorem IV.3. *For any $k, n \geq 2$, and any $\varepsilon \in [0,1)$, it is* PPA*-complete to find an ε-approximate solution with at most k cuts that satisfies at least two out of n agents in* CONSENSUS-HALVING.

[13] We assume that the valuations are provided as Turing machines that take as input a list of intervals and output the value of the union of these intervals. For a formal definition see [DFRH22, Appendix B]. Furthermore, the valuations are promised to be L-Lipschitz-continuous for some L provided in the input. The hard instances we construct below will always satisfy the promise, including monotonicity, where stated.

Next, we move on to monotone valuations. Here again, the following result is tight, since there is an efficient algorithm to find a Consensus Halving that satisfies two monotone agents using at most two cuts [DFRH22].

Theorem IV.4. *For any $k, n \geq 3$, and any $\varepsilon \in [0,1)$, it is* PPA*-complete to find an ε-approximate solution with at most k cuts that satisfies at least three out of n agents in* monotone CONSENSUS-HALVING.

Both theorems follow from the following proposition, together with the corresponding hardness results for Borsuk-Ulam, proved in the previous section, namely Theorems IV.1 and IV.2, respectively. We note that this proposition also implies that finding a 2-out-of-n solution to monotone kD-TO-nD-BORSUK-ULAM is easy for any $k, n \geq 2$, since, as mentioned above, monotone CONSENSUS-HALVING is easy for those parameters.

Proposition IV.5. *Let $k, n \in \mathbb{N}$. There is a polynomial-time reduction from kD-TO-nD-BORSUK-ULAM to* CONSENSUS-HALVING *with n agents that has the following properties:*

1) *If the BU instance is monotone, then the CH valuations functions are monotone.*
2) *Let $\varepsilon \in [0,1]$ and $s \in \mathbb{N}$ with $s \leq \min\{k, n\}$. Then for any ε-approximate solution to the CH instance that uses at most k cuts and satisfies at least s out of n agents, we obtain an ε-approximate s-out-of-n solution to the BU instance.*

Proof. We generalize the corresponding proof in [DFRH22, Proposition 3.1]. Let $F : [-1,1]^{k+1} \to [-1,1]^n$ be an instance of kD-TO-nD-BORSUK-ULAM. We construct corresponding valuation functions over the $[0,1]$ interval for n agents as follows. First, we partition the interval $[0,1]$ into $k+1$ intervals $R_1, \ldots, R_{k+1}$ of equal length, i.e., $R_j = [(j-1)/(k+1), j/(k+1)]$. Next, for any measurable subset A of $[0,1]$, we define a corresponding vector $\boldsymbol{x}(A) \in [-1,1]^{k+1}$ as follows

$$x_j(A) = 2(k+1) \cdot \lambda(A \cap R_j) - 1 \qquad \forall j \in [k+1].$$

Finally, we let the valuation of agent $i \in [n]$ be given by $v_i : \Lambda([0,1]) \to [0,1]$,

$$v_i(A) = \frac{F_i(\boldsymbol{x}(A)) + 1}{2}.$$

Given that F is L-Lipschitz-continuous, it follows that v_i is $(k+1)L$-Lipschitz-continuous. Furthermore, we have that $v_i(\emptyset) = (F_i(-1,\ldots,-1)+1)/2 = 0$, since $F_i(-1,\ldots,-1) = -1$, which follows from $F_i(1,\ldots,1) = 1$ and the fact that F is odd. If F is monotone, then the valuations v_i are also monotone. Indeed, if $A \subseteq B$, then $\boldsymbol{x}(A) \leq \boldsymbol{x}(B)$, and thus $F_i(\boldsymbol{x}(A)) \leq F_i(\boldsymbol{x}(B))$.

Now consider any $\varepsilon \in [0,1]$ and any $s \in \mathbb{N}$ with $s \leq \min\{k, n\}$, and let I^+, I^- be a partition of the interval $I = [0,1]$ using at most k cuts that ε-satisfies at least s out of n agents. Note that $\boldsymbol{x}(I^+) = -\boldsymbol{x}(I^-)$, because $\lambda(I^+ \cap R_j) +$

$\lambda(I^- \cap R_j) = 1/(k+1)$ for all $j \in [k+1]$. In particular, since F is odd,

$$\begin{aligned} F_i(\boldsymbol{x}(I^+)) - F_i(\boldsymbol{x}(I^-)) &= F_i(\boldsymbol{x}(I^+)) - F_i(-\boldsymbol{x}(I^+)) \\ &= 2F_i(\boldsymbol{x}(I^+)). \end{aligned} \quad (2)$$

Furthermore, since the partition uses at most k cuts, there exists $j \in [k+1]$ such that R_j does not contain any cut. This implies that $|x_j(I^+)| = 1$, and thus $\boldsymbol{x}(I^+)$ lies on the boundary of $[-1,1]^{k+1}$. Since I^+, I^- constitute a solution, there exists $S \subseteq [n]$, $|S| = s$, such that $|v_i(I^+) - v_i(I^-)| \leq \varepsilon$ for all $i \in S$. By definition of v_i, this implies that $|F_i(\boldsymbol{x}(I^+)) - F_i(\boldsymbol{x}(I^-))| \leq 2\varepsilon$. Finally, by (2) we obtain that $|F_i(\boldsymbol{x}(I^+))| \leq \varepsilon$ for all $i \in S$, as desired. □

C. Generalized Ham-Sandwich

In this section we present our results for a generalized Ham Sandwich problem. We begin with a statement and proof of the more general theorem. In this generalization, we consider general set functions $\mu : \Lambda([0,1]^d) \to [0,1]$ defined over the set of Lebesgue measurable subsets of $[0,1]^d$, denoted by $\Lambda([0,1]^d)$. We say that μ is Lipschitz-continuous with some constant $L > 0$, if for all $A, B \in \Lambda([0,1]^d)$ we have $|\mu(A) - \mu(B)| \leq L \cdot \lambda(A \Delta B)$, where λ is the Lebesgue measure over $[0,1]^d$ and $A \Delta B$ denotes the symmetric difference of A and B as before. Finally, we say that μ is monotone, if $\mu(A) \leq \mu(B)$, whenever $A \subseteq B$.

Theorem IV.6 (Generalized Ham Sandwich)**.** *For any Lipschitz-continuous*[14] *set functions $\mu_1, \dots, \mu_d : \Lambda([0,1]^d) \to [0,1]$, there exists a hyperplane H partitioning $[0,1]^d$ into (H^+, H^-) such that $\mu_i(H^+) = \mu_i(H^-)$ for all $i \in [d]$.*

Proof. The proof is adapted from the work of Stone and Tukey [ST42] to the more general set functions we consider here. For $a \in \mathcal{S}^d$ define the hyperplane $H_a = \{x \in [0,1]^d : a_1x_1 + \dots + a_dx_d + a_{d+1} = 0\}$, and the halfspaces $H_a^+ = \{x \in [0,1]^d : a_1x_1 + \dots + a_dx_d + a_{d+1} \geq 0\}$ and $H_a^- = \{x \in [0,1]^d : a_1x_1 + \dots + a_dx_d + a_{d+1} \leq 0\}$.

Now define the function $f : \mathcal{S}^d \to \mathbb{R}^d$, $f_i(a) = \mu_i(H_a^+) - \mu_i(H_a^-)$ for all $i \in [d]$. Since $H_{-a}^+ = H_a^-$, it immediately follows that f is odd, i.e., $f(-a) = -f(a)$. If f is also continuous, then we can apply the Borsuk-Ulam theorem to deduce that there exists a such that $f(a) = 0$. Then the hyperplane H_a will bisect all d measures.

It remains to prove that f is continuous. Consider any sequence $(a^n)_n$ in $\mathcal{S}^d$ that converges to some $a^* \in \mathcal{S}^d$. We want to show that $f_i(a^n)$ converges to $f_i(a^*)$ for all i. By the Lipschitz-continuity of μ_i we can write

$$\begin{aligned} |\mu_i(H_{a^n}^+) - \mu_i(H_{a^*}^+)| &\leq L \cdot \lambda(H_{a^n}^+ \Delta H_{a^*}^+) \\ &\leq L(\lambda(H_{a^n}^+ \setminus H_{a^*}^+) + \lambda(H_{a^*}^+ \setminus H_{a^n}^+)). \end{aligned}$$

Now, given that the halfspaces are intersected with $[0,1]^d$, it is not too hard to see that for any $\varepsilon > 0$, there exists $N \in \mathbb{N}$ such that for all $n \geq N$, $\lambda(H_{a^n}^+ \setminus H_{a^*}^+) \leq \varepsilon$. This implies that $\lim_{n\to\infty} \lambda(H_{a^n}^+ \setminus H_{a^*}^+) = 0$, and a similar argument shows $\lim_{n\to\infty} \lambda(H_{a^*}^+ \setminus H_{a^n}^+) = 0$. As a result, we obtain that $\mu_i(H_{a^n}^+)$ converges to $\mu_i(H_{a^*}^+)$. An analogous argument also shows that $\mu_i(H_{a^n}^-)$ converges to $\mu_i(H_{a^*}^-)$, and thus $f_i(a^n)$ converges to $f_i(a^*)$, as desired. □

Next, we define our computational Ham Sandwich problem. Here we will allow the partition to be obtained by using multiple hyperplanes, instead of just one, so we have to explain what we mean by that. Given hyperplanes $H_1, \dots, H_k$, they partition $[0,1]^d$ into a certain number of cells. A partition (A^+, A^-) of $[0,1]^d$ is obtained by k hyperplanes $H_1, \dots, H_k$, if there is a way to label each cell by "+" or "−" such that the union of all positive cells yields A^+, and the union of all negative cells yields A^-. Computationally, the partition will be represented by a list of hyperplanes, and by a list of each cell and corresponding sign. Since k is constant, the number of cells is also constant.

Definition 7. Let $d, n, s, k \in \mathbb{N}$ with $s \leq \min\{d, n\}$, and $\varepsilon \in [0,1]$ be constants. An instance of HAM-SANDWICH with those parameters consists of n set functions[15] $\mu_1, \dots, \mu_n : \Lambda([0,1]^d) \to [0,1]$ that are L-Lipschitz-continuous for some given $L > 0$. The goal is to find a partition (A^+, A^-) of $[0,1]^d$ obtained by at most k hyperplanes that is an ε-approximate s-out-of-n solution, i.e., such that there exists $S \subseteq [n]$ with $|S| = s$ such that $|\mu_i(A^+) - \mu_i(A^-)| \leq \varepsilon$ for all $i \in S$.

If the set functions are monotone, then we call the problem *monotone* HAM-SANDWICH.

For $d = n = s$ and $k = 1$ the problem can be shown to lie in PPA by observing that the proof of Theorem IV.6 yields a polynomial-time reduction to a Borsuk-Ulam problem. For any other valid setting of the parameters, the problem can always be reduced to the setting $d = n = s$ and $k = 1$, so the problem always lies in PPA. Furthermore, it is clear that binary search can be used to bisect a single set function, so the problem can be solved in polynomial time for $s = 1$ and any $d, n, k \geq 1$. The reduction to Borsuk-Ulam can also be used to show that the *monotone* problem is easy for $s = 2$ and any $d, n \geq 2$, $k \geq 1$, since the corresponding monotone version of Borsuk-Ulam is polynomial-time solvable.

We obtain the following two matching hardness results.

Theorem IV.7. *For any $d, n, k \in \mathbb{N}$ with $d, n \geq 2$ and any $\varepsilon \in [0,1)$, it is* PPA*-complete to find an ε-approximate 2-out-of-n solution to* HAM-SANDWICH *that uses at most k hyperplanes.*

Theorem IV.8. *For any $d, n, k \in \mathbb{N}$ with $d, n \geq 3$ and any $\varepsilon \in [0,1)$, it is* PPA*-complete to find an ε-approximate 3-out-*

[14]We use Lipschitz-continuity here, because it is more convenient for computational purposes below. The theorem also holds for weaker notions of continuity.

[15]We assume that the set functions are provided as Turing machines that take as input a list of hyperplanes and a subset of the resulting cells, and output the value of the union of these cells. This is analogous to the formal definition of Consensus Halving. Furthermore, the set functions are promised to be L-Lipschitz-continuous with the provided constant L. The hard instances we construct below will always satisfy the promise, including monotonicity, where stated.

of-n solution to monotone HAM-SANDWICH *that uses at most k hyperplanes.*

Both theorems follow from the following proposition, together with the corresponding hardness results for Borsuk-Ulam, proved in the previous section, namely Theorems IV.1 and IV.2, respectively.

Proposition IV.9. *Let $d, n, k \in \mathbb{N}$. There is a polynomial-time reduction from* (kd)D-TO-nD-BORSUK-ULAM *to d-dimensional* HAM-SANDWICH *with n set functions that has the following properties:*

1) *If the BU instance is monotone, then the* HAM-SANDWICH *set functions are monotone.*
2) *Let $\varepsilon \in [0,1]$ and $s \in \mathbb{N}$ with $s \leq \min\{d, n\}$. Then for any ε-approximate solution to the* HAM-SANDWICH *instance that uses at most k hyperplanes and bisects at least s out of n set functions, we obtain an ε-approximate s-out-of-n solution to the BU instance.*

Proof. Our proof technique is similar to the reduction from Borsuk-Ulam to Consensus Halving presented in the previous section. The main difference is that we embed along the moment curve, a curve in $\mathbb{R}^d$ such that any hyperplane intersects it in at most d points. The idea of embedding along the moment curve has been used before, in particular to embed a necklace splitting instance into a ham sandwich problem; see [FRG23] and references therein.

Let $F : [-1,1]^{kd+1} \to [-1,1]^n$ be an instance of (kd)D-TO-nD-BORSUK-ULAM. We construct the HAM-SANDWICH set functions $\mu_1, \ldots, \mu_n$ as follows.

The moment curve is the curve defined by $c : [0,1] \to [0,1]^d$, $t \mapsto (t, t^2, \ldots, t^d)$. It has the property that any hyperplane intersects it in at most d points, which we will crucially use below. Consider the points $\boldsymbol{z}^i := c(t_i)$, where $t_i = i/(kd+2)$ for all $i \in [kd+1]$. Now, define for each $i \in [kd+1]$, the hypercube C_i centered at $\boldsymbol{z}_i$ and with sidelength ℓ. We pick $\ell > 0$ sufficiently small such that (i) $C_i \subseteq [0,1]^d$, and (ii) any hyperplane intersects at most d of the $kd+1$ different C_i's. For (i) it suffices to make sure that ℓ satisfies $\ell/2 \leq 1/(kd+2)^d$. For (ii), a compactness argument shows that there must exist $\ell > 0$ such that this holds (otherwise it would also not hold at $\ell = 0$). Note that the existence of such an $\ell > 0$ is enough for us here, since its value only depends on d, n, and k which are fixed constants.

Now, for any measurable subset A of $[0,1]^d$, we define a vector $\boldsymbol{x}(A) \in [-1,1]^{kd+1}$ as follows

$$x_j(A) = \frac{2}{\ell^d}\lambda(A \cap C_j) - 1 \qquad \forall j \in [kd+1].$$

The HAM-SANDWICH set functions $\mu_1, \ldots, \mu_n : \Lambda([0,1]^d) \to [0,1]$ are then defined as

$$\mu_i(A) = \frac{F_i(\boldsymbol{x}(A)) + 1}{2}.$$

Since F is Lipschitz-continuous with some Lipschitz constant L, it follows that μ_i is Lipschitz-continuous with Lipschitz constant L/ℓ^d. In addition, we have $\mu_i(\emptyset) = 0$, since $\boldsymbol{x}(\emptyset) = (-1, \ldots, -1)$ and $F_i(-1, \ldots, -1) = -1$. Similarly, we also have $\mu_i([0,1]^d) = 1$. Furthermore, if $A \subseteq B$, then $\boldsymbol{x}(A) \leq \boldsymbol{x}(B)$, and, if F is monotone, this implies that the set functions μ_i are also monotone.

Now consider any $\varepsilon \in [0,1]$ and any $s \in \mathbb{N}$ with $s \leq \min\{d, n\}$, and let A^+, A^- be a partition of $[0,1]^d$ that is obtained by using at most k hyperplanes and which ε-bisects s out of the n set functions. Observe that by construction we have $\boldsymbol{x}(A^+) = -\boldsymbol{x}(A^-)$, because $\lambda(A^+ \cap C_j) + \lambda(A^- \cap C_j) = \lambda(C_j) = \ell^d$ for all $j \in [kd+1]$. In particular, since F is odd,

$$\begin{aligned} F_i(\boldsymbol{x}(A^+)) - F_i(\boldsymbol{x}(A^-)) &= F_i(\boldsymbol{x}(A^+)) - F_i(-\boldsymbol{x}(A^+)) \\ &= 2F_i(\boldsymbol{x}(A^+)). \end{aligned} \tag{3}$$

Moreover, since any hyperplane intersects at most d of the sets C_j, it follows that k hyperplanes can intersect at most kd of the C_j's. Since there are $kd+1$ such sets, there exists $j \in [kd+1]$ such that C_j is not intersected by any of the k hyperplanes defining the partition A^+, A^-. This implies that $|x_j(A^+)| = 1$, and thus $\boldsymbol{x}(A^+)$ lies on the boundary of $[-1,1]^{kd+1}$. Now, recall that A^+, A^- are an s-out-of-n solution to the HAM-SANDWICH instance, so there exists $S \subseteq [n]$, $|S| = s$, such that $|\mu_i(A^+) - \mu_i(A^-)| \leq \varepsilon$ for all $i \in S$. By definition of μ_i, this implies that $|F_i(\boldsymbol{x}(A^+)) - F_i(\boldsymbol{x}(A^-))| \leq 2\varepsilon$. Finally, by (3) we obtain that $|F_i(\boldsymbol{x}(A^+))| \leq \varepsilon$ for all $i \in S$, and thus $\boldsymbol{x}(A^+)$ is a solution to the original BU instance. ☐

ACKNOWLEDGMENT

We would like to thank the reviewers for useful comments and suggestions.

REFERENCES

[ABB20] J. Aisenberg, M. L. Bonet, and S. Buss, "2-d tucker is PPA complete," *J. Comput. Syst. Sci.*, vol. 108, pp. 92–103, 2020. [Online]. Available: https://doi.org/10.1016/j.jcss.2019.09.002

[AEPT24] N. Alon, D. Elboim, J. Pach, and G. Tardos, "Random necklaces require fewer cuts," *SIAM Journal on Discrete Mathematics*, vol. 38, no. 2, pp. 1381–1408, 2024.

[AG21] N. Alon and A. Graur, "Efficient Splitting of Necklaces," in *Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2021, pp. 14:1–14:17.

[Alo87] N. Alon, "Splitting necklaces," *Advances in Mathematics*, vol. 63, no. 3, pp. 247–253, 1987.

[Alo88] ——, *Some Recent Combinatorial Applications of Borsuk-Type Theorems*, ser. London Mathematical Society Lecture Note Series. Cambridge University Press, 1988, pp. 1–12.

[AW86] N. Alon and D. B. West, "The Borsuk-Ulam Theorem and Bisection of Necklaces," *Proceedings of the American Mathematical Society*, vol. 98, no. 4, pp. 623–628, 1986.

[BCE+98] P. Beame, S. Cook, J. Edmonds, R. Impagliazzo, and T. Pitassi, "The Relative Complexity of NP Search Problems," *Journal of Computer and System Sciences*, vol. 57, no. 1, pp. 3–19, 1998.

[Bor33] K. Borsuk, "Drei Sätze über die n-dimensionale euklidische Sphäre," *Fundamenta Mathematicae*, vol. 20, pp. 177–190, 1933.

[BPR15] N. Bitansky, O. Paneth, and A. Rosen, "On the cryptographic hardness of finding a Nash equilibrium," in *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, 2015, pp. 1480–1498.

[BZ04] W. A. Beyer and A. Zardecki, "The early history of the ham sandwich theorem," *The American Mathematical Monthly*, vol. 111, no. 1, pp. 58–61, 2004.

[CDT09] X. Chen, X. Deng, and S.-H. Teng, "Settling the complexity of computing two-player Nash equilibria," *Journal of the ACM (JACM)*, vol. 56, no. 3, p. 14, 2009.

[CHK+19] A. R. Choudhuri, P. Hubáček, C. Kamath, K. Pietrzak, A. Rosen, and G. N. Rothblum, "Finding a Nash equilibrium is no easier than breaking Fiat-Shamir," in *Proceedings of the 51st ACM Symposium on Theory of Computing (STOC)*, 2019, pp. 1103–1114.

[DFHM25] A. Deligkas, J. Fearnley, A. Hollender, and T. Melissourgos, "Constant inapproximability for PPA," *SIAM Journal on Computing*, vol. 54, no. 1, pp. 163–192, 2025.

[DFM22] A. Deligkas, J. Fearnley, and T. Melissourgos, "Pizza sharing is PPA-hard," in *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, 2022, pp. 4957–4965.

[DFRH22] A. Deligkas, A. Filos-Ratsikas, and A. Hollender, "Two's company, three's a crowd: Consensus-halving for a constant number of agents," *Artificial Intelligence*, vol. 313, 2022, 103784.

[DGP09] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou, "The complexity of computing a Nash equilibrium," *SIAM Journal on Computing*, vol. 39, no. 1, pp. 195–259, 2009.

[FGHS22] J. Fearnley, P. Goldberg, A. Hollender, and R. Savani, "The complexity of gradient descent: CLS = PPAD ∩ PLS," *Journal of the ACM*, vol. 70, no. 1, pp. 7:1–7:74, 2022.

[FRG23] A. Filos-Ratsikas and P. W. Goldberg, "The complexity of Necklace Splitting, Consensus-Halving, and Discrete Ham Sandwich," *SIAM Journal on Computing*, vol. 52, no. 2, pp. STOC19–200–STOC19–268, 2023.

[FRHSZ23] A. Filos-Ratsikas, A. Hollender, K. Sotiraki, and M. Zampetakis, "Consensus-Halving: Does it ever get easier?" *SIAM Journal on Computing*, vol. 52, no. 2, pp. 412–451, 2023.

[GKSZ20] M. Göös, P. Kamath, K. Sotiraki, and M. Zampetakis, "On the complexity of modulo-q arguments and the Chevalley-Warning theorem," in *Proceedings of the 35th Computational Complexity Conference (CCC)*, 2020, pp. 19:1–19:42.

[GL23] P. Goldberg and J. Li, "Consensus division in an arbitrary ratio," in *Proceedings of the 14th Innovations in Theoretical Computer Science Conference (ITCS)*, 2023, pp. 57:1–57:18.

[GRRS24] R. Gao, M. Roghani, A. Rubinstein, and A. Saberi, "Hardness of approximate sperner and applications to envy-free cake cutting," in *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2024, pp. 1294–1331.

[GW85] C. H. Goldberg and D. B. West, "Bisection of Circle Colorings," *SIAM Journal on Algebraic Discrete Methods*, vol. 6, no. 1, pp. 93–106, 1985.

[Hav22] I. Haviv, "The complexity of finding fair independent sets in cycles," *computational complexity*, vol. 31, no. 2, 2022.

[Hol21] A. Hollender, "The classes PPA-k: Existence from arguments modulo k," *Theoretical Computer Science*, vol. 885, pp. 15–29, 2021.

[HR65] C. R. Hobby and J. R. Rice, "A moment problem in L1 approximation," *Proceedings of the American Mathematical Society*, vol. 16, no. 4, pp. 665–670, 1965.

[Jeř16] E. Jeřábek, "Integer factoring and modular square roots," *Journal of Computer and System Sciences*, vol. 82, no. 2, pp. 380–394, 2016.

[JKKZ21] R. Jawale, Y. T. Kalai, D. Khurana, and R. Zhang, "SNARGs for bounded depth computations and PPAD hardness from sub-exponential LWE," in *Proceedings of the 53rd ACM Symposium on Theory of Computing (STOC)*, 2021, pp. 708–721.

[JPv21] D. Jojić, G. Panina, and R. Živaljević, "Splitting necklaces, with constraints," *SIAM Journal on Discrete Mathematics*, vol. 35, no. 2, pp. 1268–1286, 2021.

[Kne55] M. Kneser, "Aufgabe 360," *Jahresbericht der Deutschen Mathematiker-Vereinigung*, vol. 2, no. 27, pp. 3–16, 1955.

[Lov78] L. Lovász, "Kneser's conjecture, chromatic number, and homotopy," *Journal of Combinatorial Theory, Series A*, vol. 25, no. 3, pp. 319–324, 1978.

[Mat08] J. Matoušek, *Using the Borsuk-Ulam theorem: lectures on topological methods in combinatorics and geometry*. Springer Science & Business Media, 2008.

[MP91] N. Megiddo and C. H. Papadimitriou, "On total functions, existence theorems and computational complexity," *Theoretical Computer Science*, vol. 81, no. 2, pp. 317–324, 1991.

[Ney46] J. Neyman, "Un théorème d'existence," *CR Acad. Sci. Paris*, vol. 222, pp. 843–845, 1946.

[Pap94] C. H. Papadimitriou, "On the complexity of the parity argument and other inefficient proofs of existence," *J. Comput. Syst. Sci.*, vol. 48, no. 3, pp. 498–532, 1994. [Online]. Available: https://doi.org/10.1016/S0022-0000(05)80063-7

[RPS21] E. Roldán-Pensado and P. Soberón, "A survey of mass partitions," *Bulletin of the American Mathematical Society*, vol. 59, no. 2, pp. 227–267, 2021.

[Sch22] P. Schnider, "The complexity of sharing a pizza," *Computing in Geometry and Topology*, vol. 1, no. 1, pp. 4:1–4:19, 2022.

[SS03] F. W. Simmons and F. E. Su, "Consensus-halving via theorems of Borsuk-Ulam and Tucker," *Mathematical social sciences*, vol. 45, no. 1, pp. 15–25, 2003.

[ST42] A. H. Stone and J. W. Tukey, "Generalized "sandwich" theorems," *Duke Mathematical Journal*, vol. 9, no. 2, 1942.

[Ste38] H. Steinhaus, "A note on the ham sandwich theorem," *Mathesis Polska*, vol. 9, pp. 26–28, 1938.

[Ste85] H. Steinlein, "Borsuk's antipodal theorem and its generalizations and applications: a survey," in *Topological methods in nonlinear analysis*, ser. Sém. Math. Sup. Presses Univ. Montréal, Montreal, QC, 1985, vol. 95, pp. 166–235.

[Ste93] ——, "Spheres and symmetry: Borsuk's antipodal theorem," *Topological Methods in Nonlinear Analysis*, vol. 1, no. 1, pp. 15–33, 1993. [Online]. Available: https://apcz.umk.pl/TMNA/article/view/TMNA.1993.004

[Tuc46] A. W. Tucker, "Some topological properties of disk and sphere," in *Proc. First Canadian Math. Congress, Montreal, 1945*. Univ. Toronto Press, Toronto, ON, 1946, pp. 285–309.