

Almost-Tight Bounds on Preserving Cuts in Classes of Submodular Hypergraphs

Sanjeev Khanna  

School of Engineering and Applied Sciences, University of Pennsylvania, Philadelphia, PA, USA

Aaron (Louie) Putterman  

School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA

Madhu Sudan  

School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA

Abstract

Recently, a number of variants of the notion of cut-preserving hypergraph sparsification have been studied in the literature. These variants include directed hypergraph sparsification, submodular hypergraph sparsification, general notions of approximation including spectral approximations, and more general notions like sketching that can answer cut queries using more general data structures than just sparsifiers. In this work, we provide reductions between these different variants of hypergraph sparsification and establish new upper and lower bounds on the space complexity of preserving their cuts. Specifically, we show that:

1. $(1 \pm \epsilon)$ directed hypergraph spectral (respectively cut) sparsification on n vertices efficiently reduces to $(1 \pm \epsilon)$ undirected hypergraph spectral (respectively cut) sparsification on $n^2 + 1$ vertices. Using the work of Lee and Jambulapati, Liu, and Sidford (STOC 2023) this gives us directed hypergraph spectral sparsifiers with $O(n^2 \log^2(n)/\epsilon^2)$ hyperedges and directed hypergraph cut sparsifiers with $O(n^2 \log(n)/\epsilon^2)$ hyperedges by using the work of Chen, Khanna, and Nagda (FOCS 2020), both of which improve upon the work of Oko, Sakaue, and Tanigawa (ICALP 2023).
2. Any cut sketching scheme which preserves all cuts in any directed hypergraph on n vertices to a $(1 \pm \epsilon)$ factor (for $\epsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$) must have worst-case bit complexity $n^{3-o(1)}$. Because directed hypergraphs are a subclass of submodular hypergraphs, this also shows a worst-case sketching lower bound of $n^{3-o(1)}$ bits for sketching cuts in general submodular hypergraphs.
3. $(1 \pm \epsilon)$ monotone submodular hypergraph cut sparsification on n vertices efficiently reduces to $(1 \pm \epsilon)$ symmetric submodular hypergraph sparsification on $n + 1$ vertices. Using the work of Jambulapati et. al. (FOCS 2023) this gives us monotone submodular hypergraph sparsifiers with $\tilde{O}(n/\epsilon^2)$ hyperedges, improving on the $O(n^3/\epsilon^2)$ hyperedge bound of Kenneth and Krauthgamer (arxiv 2023).

At a high level, our results use the same general principle, namely, by showing that cuts in one class of hypergraphs can be simulated by cuts in a simpler class of hypergraphs, we can leverage sparsification results for the simpler class of hypergraphs.

2012 ACM Subject Classification Theory of computation → Sketching and sampling

Keywords and phrases Sparsification, sketching, hypergraphs

Digital Object Identifier 10.4230/LIPIcs.ICALP.2024.98

Category Track A: Algorithms, Complexity and Games

Funding *Sanjeev Khanna*: Supported in part by NSF awards CCF-1934876 and CCF-2008305.

Aaron (Louie) Putterman: Supported in part by the Simons Investigator Awards of Madhu Sudan and Salil Vadhan, NSF Award CCF 2152413 and a Hudson River Trading PhD Research Scholarship.

Madhu Sudan: Supported in part by a Simons Investigator Award and NSF Award CCF 2152413.

 © Sanjeev Khanna, Aaron Putterman, and Madhu Sudan;
licensed under Creative Commons License CC-BY 4.0
51st International Colloquium on Automata, Languages, and Programming (ICALP 2024).
Editors: Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson;
Article No. 98; pp. 98:1–98:17

 Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



1 Introduction

Sparsification deals with the following natural question: given a large object, how much can we compress it while still retaining some of its key properties? In the realm of graphs, this has been a well-studied notion spanning decades of research. Starting with the work of Karger [9], the question of how sparse we can make a graph while still preserving the approximate sizes of every cut has been a central topic of research. Since then, numerous works by many authors have resolved this question (starting with the work of Benczúr and Karger [2]) and pushed the boundaries of this research beyond just graph cuts [1, 21, 12, 3].

More rigorously, for a weighted graph $G = (V, E)$ on n vertices, we can define a cut in the graph corresponding to each set $S \subseteq V$. For such a set S , we define the vector $\mathbf{1}_S \in \{0, 1\}^{|V|}$ as the indicator vector of whether the i th vertex is in S . Using this vector, we say that $\text{cut}_G(S) = \sum_{(u,v) \in E} w_{(u,v)} (\mathbf{1}_{S_u} - \mathbf{1}_{S_v})^2$, i.e., the weight of the edges crossing between S and $V - S$. A cut-sparsifier asks for a reweighted subset of edges $\hat{E} \subseteq E$ such that in the graph $G = (V, \hat{E})$, with the corresponding new weights \hat{w} , for every $S \subseteq V$

$$(1 - \varepsilon)\text{cut}_G(S) \leq \text{cut}_{\hat{G}}(S) \leq (1 + \varepsilon)\text{cut}_G(S).$$

The seminal work of [2] was the first to show the existence of such sparsifiers \hat{G} for any graph G such that $|\hat{E}| = \tilde{O}(n/\varepsilon^2)$. Subsequent work in the *spectral* regime asked whether such sparsifiers still exist when we consider real-valued vectors as opposed to cut-vectors. In this setting, we define a Laplacian L_G for our graph G . We say that for $x \in \mathbb{R}^{|V|}$

$$x^T L_G x = \sum_{(u,v) \in E} w_{(u,v)} (x_u - x_v)^2.$$

The goal in this regime instead becomes finding a reweighted subgraph \hat{G} such that for every $x \in \mathbb{R}^{|V|}$,

$$(1 - \varepsilon)x^T L_{\hat{G}} x \leq x^T L_G x \leq (1 + \varepsilon)x^T L_{\hat{G}} x.$$

Work by Batson, Spielman, and Srivastava, and Spielman and Teng [1, 21] settled the size complexity of spectral sparsifiers for ordinary graphs by showing the existence of such sparsifiers of size $O(n/\varepsilon^2)$.

Recently, starting with the work of Kogan and Krauthgamer [12], a natural extension to the study of graph sparsification has been the study of sparsifying *hypergraphs*. In this setting, one is given a hypergraph $H = (V, E)$, and asked to preserve to a $(1 \pm \varepsilon)$ factor the weight of all hyperedges crossing a particular cut. A cut is given by a bichromatic coloring of the vertices and a hyperedge is considered cut if it is not monochromatic. Work by Chen, Khanna, and Nagda [3] was the first to completely characterize the cut-sparsifiability of hypergraphs, which showed that there exist $(1 \pm \varepsilon)$ -cut-sparsifiers for any hypergraph on n vertices of size $O(n \log(n)/\varepsilon^2)$. As in the graph setting, where the natural next step from cut-sparsifiers was spectral-sparsifiers, Soma and Yoshida [20] later introduced this notion of spectral *hypergraph* sparsification. More explicitly, the “energy function” (also called the Laplacian) of an undirected hypergraph $H = (V, E)$ is as follows:

$$L_H(x) = \text{cut}_H(x) = \sum_{e \in E} w_e \max_{u, v \in e} (x_u - x_v)^2.$$

A $(1 \pm \varepsilon)$ -spectral sparsifier for an undirected hypergraph then corresponds to a reweighted subhypergraph of H , denoted by \hat{H} such that for any $x \in \mathbb{R}^{|V|}$,

$$(1 - \varepsilon)L_H(x) \leq L_{\hat{H}}(x) \leq (1 + \varepsilon)L_H(x).$$

This question of whether one could preserve the Laplacian of undirected hypergraphs with only a near-linear number of hyperedges was then resolved by Kapralov et. al. [7], Jambulapati, Liu, and Sidford [6], and Lee [14] in the affirmative.

More recently however, work has sought to generalize hypergraph sparsification even further. Indeed, given a hypergraph $H = (V, E)$, instead of viewing edge-cuts in the traditional way (i.e., for a bichromatic coloring of the vertices counting how many hyperedges are not one color), a more general *splitting* function is assigned to each hyperedge $e \subseteq V$. This splitting function is a set function $g_e : 2^e \rightarrow \mathbb{R}^{\geq 0}$. One natural extension to the case of ordinary hypergraphs that has received particular attention is the case in which these splitting functions g_e are also required to be submodular [10, 13] (though there has also been work on the regime where these functions are not submodular, for instance with parity functions in [11]). Such submodular hypergraphs appear in numerous contexts, for instance in clustering data points with complex relationships [15, 16, 22, 23] and summarizing data [17]. For such a submodular hypergraph $H = (V, E)$, the value on any cut $S \subset V$ is

$$\text{cut}_H(S) = \sum_{e \in E} g_e(S \cap e).$$

Recall that a function $g : 2^V \rightarrow \mathbb{R}^{\geq 0}$ is said to be *submodular* if it has the property of diminishing returns. That is, for any $S \subset T \subset V$, and any element $x \in V, x \notin T$,

$$g(S \cup \{x\}) - g(S) \geq g(T \cup \{x\}) - g(T).$$

Under this definition, one type of submodular hypergraph is a *directed hypergraph*. In a directed hypergraph, one can view each directed hyperedge instead as a tuple $(e_{\text{tail}}, e_{\text{head}})$ of subsets of V . The cut function of a directed hyperedge e on cut S is 1 if and only if an element from S is in e_{tail} and an element from $V - S$ is in e_{head} . More explicitly, for a directed hypergraph $G = (V, E, w)$ on n vertices, and a vector $x \in \mathbb{R}^n$, we can define the Laplacian for G as

$$L_G(x) = \sum_{e \in E} \max_{u \in L(e), v \in R(e)} (x_u - x_v)_+^2.$$

In this context, $(x_u - x_v)_+ = \max((x_u - x_v), 0)$, and directed hypergraph cuts are simply the restriction of the vector x to be in $\{0, 1\}^{|V|}$ (seen as the indicator vector for a set $S \subseteq V$). A non-zero contribution from a hyperedge occurs only if a tail vertex of the hyperedge has a larger value than a head vertex of the hyperedge.

One can check that in the cut regime (i.e. $x \in \{0, 1\}^n$), each directed hyperedge cut yields a submodular function $g_e : 2^{e_{\text{head}} \cup e_{\text{tail}}} \rightarrow \mathbb{R}^{\geq 0}$. In what follows, we describe our contributions to various problems in this area.

1.1 Improved Bounds for Directed Hypergraph Sparsification

In the graph case, it is known that directed graph cut-sparsifiers for graphs with n vertices can require as many as $\Omega(n^2)$ edges to preserve cuts to a $(1 \pm \varepsilon)$ factor. In this sense, directed graph cut-sparsification is a trivial task, as any graph has at most $O(n^2)$ edges to begin with. Contrary to this however, directed *hypergraph* sparsification is non-trivial. While the same $\Omega(n^2)$ lower bound exists, a directed hypergraph can have as many as 4^n directed hyperedges to start with, so a sparsifier with $O(n^2)$ directed hyperedges is a vast improvement. This observation has led to a rich line of research studying the feasibility of sparsifying directed hypergraphs. The first work on this front was the work of [20] which

showed the existence of directed hypergraph sparsifiers with $O(n^3/\varepsilon^2)$ directed hyperedges and gave a polynomial time algorithm for computing them. Later work by [7] presented a proof of sparsifiers with $\tilde{O}(nr/\varepsilon^2)$ (where r is the maximum size of any hyperedge) hyperedges for undirected hypergraph spectral sparsification, and with $\tilde{O}(n^2r^3/\varepsilon^2)$ directed hyperedges for directed hypergraph spectral sparsification by tuning their algorithm and performing a different analysis. In particular, this improved upon the result of [20] in the regime where r is constant. Note that as with graphs, spectral sparsification is a *stronger* notion than cut sparsification, so in particular, these proofs imply the existence of cut-sparsifiers of the same complexity.

Ultimately however, the complexity of directed spectral hypergraph sparsification was nearly settled by the work of Oko, Sakaue, and Tanigawa [18], who showed $(1 \pm \varepsilon)$ spectral-sparsifiers with $O(n^2 \log^3(n/\varepsilon)/\varepsilon^2)$ directed hyperedges exist for directed hypergraphs on n vertices.

Continuing this line of research, we show that fundamentally, the task of directed hypergraph sparsification can be reduced in a black-box manner to undirected hypergraph spectral sparsification.

More specifically, we show there is a lifting from a directed hypergraph on n vertices to an undirected hypergraph on $n^2 + 1$ vertices such that the Laplacian of every individual hyperedge is simultaneously preserved. That is, we show the following theorem:

► **Theorem 1.** *For $H = (V, E)$ an a directed hypergraph on n vertices, one can compute an undirected hypergraph $\psi(H)$ on $n^2 + 1$ vertices in time $O(mr^2)$ (where m is the number of hyperedges in H , and r is the maximum size of any hyperedge in H), such that for any $x \in \mathbb{R}^n$, one can also compute $\vartheta(x) \in \mathbb{R}^{n^2+1}$ in time $O(n^2)$ such that*

$$L_H(x) = L_{\psi(H)}(\vartheta(x)).$$

Moreover, for any hyperedge $e \in H$, there is a single corresponding hyperedge $\psi(e)$ in $\psi(H)$ such that

$$L_e(x) = L_{\psi(e)}(\vartheta(x)).$$

The size of $\psi(e)$ is at most $|e|^2$. Further, for $x \in \{0, 1\}^n$, i.e. corresponding to a cut, $\vartheta(x)$ will be in $\{0, 1\}^{n^2+1}$, i.e. also corresponding to a cut.

We can then use the existing state of the art literature of undirected spectral hypergraph sparsification [6, 14] to conclude the existence of directed spectral hypergraph sparsifiers with only $O(n^2 \log(n) \log(r)/\varepsilon^2)$ hyperedges which can be found in time $\tilde{O}(mr^2)$, where m is the original number of hyperedges and r is the maximum size of any hyperedge. Note that this bound on the size of sparsifiers improves on the result of [18], and in particular, makes the dependence on ε exactly $O(1/\varepsilon^2)$, which now matches the literature for undirected sparsification. That is, we show the following:

► **Theorem 2.** *For any directed hypergraph $H = (V, E)$ on n vertices, and any $0 < \varepsilon < 1$ there exists a weighted sub-hypergraph \hat{H} such that for all $x \in \mathbb{R}^n$:*

$$(1 - \varepsilon)L_H(x) \leq L_{\hat{H}}(x) \leq (1 + \varepsilon)L_H(x),$$

and \hat{H} only has $O(n^2 \log(n) \log(r)/\varepsilon^2)$ hyperedges, where r is the maximum size of any hyperedge of H .

As an additional benefit, because the reduction of Theorem 1 preserves cut vectors, we can also invoke the result of [3] to conclude the existence of directed hypergraph *cut*-sparsifiers with $O(n^2 \log(n)/\varepsilon^2)$ hyperedges.

1.2 Lower Bounds for Sketching Cuts in Directed Hypergraphs

We next focus on the bit complexity of creating cut-sparsifiers for directed hypergraphs. This is done in hopes of answering an open question from [10] regarding the bit-complexity of arbitrary sketching schemes for submodular hypergraphs. In prior work [18, 10], a lower bound of size $\Omega(n^3)$ (ignoring ε) was established for the bit complexity of any directed hypergraph cut-*sparsifier*. However, lower bounds for sparsifiers explicitly take advantage of the sparsifier structure by starting with known examples of sparsifiers that require $\Omega(n^2)$ hyperedges, and then padding these hyperedges with random vertices in their tail such that the bit complexity of each hyperedge becomes $\Omega(n)$. One can trivially show that this padding does not change the requirement of preserving $\Omega(n^2)$ hyperedges. Because sparsifiers are limited to storing only hyperedges that were originally present, this then forces a bit complexity lower bound of $\Omega(n^3)$. However, this same technique is not amenable to a sketching lower bound as the padding procedure only adds complexity to each hyperedge, and not necessarily to the cut function as a whole. Thus, the difficulty is in showing that the cut function itself requires a large description size, regardless of how we choose to represent it. This marks a fundamental difference.

Addressing this, we show the following theorem:

► **Theorem 3.** *Any $(1 \pm \varepsilon)$ cut-sketching scheme for directed hypergraphs on n vertices must have worst-case space $\frac{n^3}{2^{O(\sqrt{\log(n)})}}$ bits (for $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$).*

At a high level, our proof takes advantage of a result of Kapralov et. al. [8]. In this work, the authors show that there exists a family of undirected hypergraphs on n vertices, each with at most n hyperedges, such that any sketching scheme which can sketch cuts in any of the hypergraphs in this family to an additive error of εn (for $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$) must have worst-case size at least $\frac{n^2}{2^{O(\sqrt{\log(n)})}}$. We show that by using a specific construction of a directed hypergraph, along with a specific reconstruction procedure, we can actually store an additive cut-approximation to n distinct undirected hypergraphs in a single cut-sketch of a directed hypergraph. That is, we show the following theorem:

► **Theorem 4.** *For any undirected hypergraphs H_1, \dots, H_n , each on vertex set V , with $|V| = n$, there exists a directed hypergraph G on $2n$ vertices, such that given a $(1 \pm \varepsilon)$ cut-sketch for G , for any of the undirected hypergraphs $H_i = (V, E_i)$, one can recover $\text{cut}_{H_i}(S)$ to within additive error $3\varepsilon|E_i|$.*

Now, by sampling these undirected hypergraphs H_1, \dots, H_n from a specific family of hypergraphs, we can argue that simultaneously preserving the cut-values in all of these hypergraphs (even to an additive error) requires a data structure of size $\frac{n^2}{2^{O(\sqrt{\log(n)})}} \cdot n = \frac{n^3}{2^{O(\sqrt{\log(n)})}}$. In particular, by the previous reduction, any general scheme for sketching directed hypergraphs or submodular hypergraphs would be such a scheme, and therefore must have worst-case size at least $\Omega(n^{3-o(1)})$ (for $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$).

Prior to our work, there was no known super-quadratic (in n) lower bound on the sketching complexity of cuts in directed hypergraphs. In conjunction with our positive results on the sparsifiability of directed hypergraphs, this shows that directed hypergraph sparsification is almost-optimal even among all possible sketches for preserving cut values. That is, from the previous section, we know that directed hypergraph sparsifiers approximately preserve the sizes of all cuts in a directed hypergraph to a factor $(1 \pm \varepsilon)$ using $\tilde{O}(n^3/\varepsilon^2)$ bits. In conjunction with our lower bound, we can conclude that this is almost the best possible (among *any* sketching scheme) in the regime where $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$. Thus, we show that for approximately storing cuts in directed hypergraphs using as few bits as possible, using a sparsifier is almost optimal. We view this as an important contribution of our work.

1.3 Cut Sparsifiers for Monotone Submodular Hypergraphs

Finally, we show that one can simulate cuts in monotone submodular hypergraphs with cuts in symmetric submodular hypergraphs. Recall that a set function is monotone if $f(S \cup \{t\}) \geq f(S)$, and we say that a submodular hypergraph is monotone if *every* splitting function is also monotone. This model of hypergraphs was specifically studied in the work of [10], where their sparsifiers ultimately achieved a complexity of $O(n^3/\varepsilon^2)$ hyperedges. In particular, monotone submodular functions capture a wide variety of natural and common functions such as matroid rank and entropy of random variables.

With respect to this, we show the following theorem:

► **Theorem 5.** Suppose $f : 2^V \rightarrow \mathbb{R}^{\geq 0}$ is a monotone, submodular function. Then, $f' : 2^{V \cup \{*\}} \rightarrow \mathbb{R}^{\geq 0}$ defined as $\forall S \subseteq V$

$$f'(S) = f(S) = f'(V - S \cup \{*\})$$

is submodular and symmetric.

Next, we show that given an arbitrary monotone, submodular hypergraph on n vertices, we can lift this to a symmetric submodular hypergraph on $n + 1$ vertices, where the single extra vertex is the $\{*\}$ vertex from the preceding theorem. Next, for each individual splitting function $g_e : 2^e \rightarrow \mathbb{R}^+$ in the monotone, submodular hypergraph, we replace g_e with g'_e , again using the preceding theorem.

Note that for each monotone submodular function, we *re-use* the same $\{*\}$ vertex. Thus, the increase in the size of the vertex set is only 1. Finally, we can then invoke a result from [5], which states that for any submodular hypergraph H where each splitting function is symmetric, one can calculate a sparsifier for H with only $\tilde{O}(n/\varepsilon^2)$ hyperedges.

We then get the following:

► **Theorem 6.** Let $H = (V, E)$ be a hypergraph, such that $\forall e \in E$, the corresponding splitting function $g_e : 2^e \rightarrow \mathbb{R}^{\geq 0}$ is submodular and monotone. Then there exists a $(1 \pm \varepsilon)$ cut-sparsifier for H retaining only $\tilde{O}(n/\varepsilon^2)$ hyperedges.

Prior to this work, the best known upper bound for the size complexity (in hyperedges) for $(1 \pm \varepsilon)$ -sparsifying any monotone submodular hypergraph was $O(n^3/\varepsilon^2)$ hyperedges, proved in the work of [10]. Our result essentially improves this to the best possible, where we now only have a near-linear dependence on the size of the vertex set. We view it as an interesting open question if one can extend our proof method used here to general submodular functions (although this case will necessarily require a blow-up of at least quadratic size).

1.4 Overview

At a high level, all of our results use the same general principle, namely, by showing that cuts in one class of hypergraphs can be simulated by cuts in a simpler class of hypergraphs, we can leverage sparsification results for the simpler class of hypergraphs. This leads to our proofs being quite simple despite the fact that the results improve upon the state-of-the-art knowledge in hypergraph sparsification.

In Section 2 we introduce formal definitions and other preliminaries. In Section 3 we present the algorithms for sparsifying directed hypergraphs by reducing to undirected hypergraph sparsification. Next, in Section 4, we show how to simultaneously simulate cuts in many different undirected graphs thereby leading to new lower bounds for the worst case size of sketching cuts in directed hypergraphs. Finally, in Section 5, we show how to sparsify arbitrary monotone, submodular hypergraphs to near-optimal size.

2 Preliminaries

First, we introduce the definitions of undirected and directed hypergraphs.

► **Definition 7.** An *undirected hypergraph* $G = (V, E)$ is a collection of vertices V , with associated hyperedges $e \in E$, where $e \subseteq V$ can be of arbitrary size.

► **Definition 8.** A *directed hypergraph* $H = (V, E)$ is a collection of vertices V along with directed hyperedges $e \in E$. Each directed hyperedge is of the form $e = (e_{tail}, e_{head})$, where $e_{head}, e_{tail} \subseteq V$. We will use $L(e) = e_{tail}$, $R(e) = e_{head}$. Note that e_{head}, e_{tail} are not necessarily disjoint.

Next, we introduce the definition of spectral sparsifiers for both undirected and directed hypergraphs.

► **Definition 9.** For an undirected hypergraph $G = (V, E, w)$ on n vertices, and a vector $x \in \mathbb{R}^n$, the *quadratic form of the Laplacian of G* is

$$L_G(x) = \sum_{e \in E} \max_{u, v \in e} (x_u - x_v)^2.$$

► **Definition 10.** For a directed hypergraph $G = (V, E, w)$ on n vertices, and a vector $x \in \mathbb{R}^n$, the *directed quadratic form of the Laplacian of G* is

$$L_G(x) = \sum_{e \in E} \max_{u \in L(e), v \in R(e)} (x_u - x_v)_+^2.$$

In this context, $(x_u - x_v)_+ = \max((x_u - x_v), 0)$. A non-zero contribution from a hyperedge occurs only if a tail vertex of the hyperedge has a larger value than a head vertex of the hyperedge. Note that the head set and tail set of a directed hyperedge are not necessarily disjoint.

► **Definition 11.** For a (directed or undirected) hypergraph $G = (V, E)$ on n vertices, a $(1 \pm \varepsilon)$ -spectral sparsifier for G is a weighted (directed or undirected) sub-hypergraph H such that for every $x \in \mathbb{R}^n$,

$$(1 - \varepsilon)L_G(x) \leq L_H(x) \leq (1 + \varepsilon)L_G(x).$$

Further, we require that the hyperedges of H are a subset of the hyperedges of G .

► **Remark 12.** For all the above definitions, if a reweighted sub-hypergraph H of G preserves the quadratic form for vectors $x \in \{0, 1\}^n$ to $(1 \pm \varepsilon)$ multiplicative error, we say that H is a **cut-sparsifiers**. Note that all spectral sparsifiers are cut-sparsifiers, while the converse is not necessarily true.

We also refer to cut-sizes in hypergraphs. A cut is specified by a set $S \subseteq V$, and we say the size of the cut S in G (denoted $|\text{cut}_G(S)|$) is $L_G(\mathbf{1}_S)^T$, where $\mathbf{1}_S$ is the indicator vector in $\{0, 1\}^n$ for the set S . Combinatorially, this refers to the weight of the hyperedges that are “leaving” the set S .

Next we define submodular functions and submodular hypergraphs.

► **Definition 13.** A function $g : 2^V \rightarrow \mathbb{R}^{\geq 0}$ is said to be submodular if for any $S \subset T \subset V$, and any $x \in V - T$,

$$g(S \cup \{x\}) - g(S) \geq g(T \cup \{x\}) - g(T).$$

Using this, we can define a submodular hypergraph.

► **Definition 14.** A submodular hypergraph $H = (V, E)$ is a set of n vertices along with a set of hyperedges E . For each hyperedge $e \in E$, there is a corresponding submodular splitting function $g_e : 2^e \rightarrow \mathbb{R}^{\geq 0}$. For any subset $S \subseteq V$, the corresponding cut of the submodular hypergraph is

$$\text{cut}_H(S) = \sum_{e \in E} g_e(S \cap e).$$

► **Definition 15.** We say that a data structure G is a $(1 \pm \varepsilon)$ -cut sketch of a submodular hypergraph $H = (V, E)$, if for any $S \subseteq V$ one can deterministically recover $\text{cut}_H(S)$ to within a $(1 \pm \varepsilon)$ factor using only the data structure G , and the set S .

We will use the following result from [5] regarding the sparsifiability of symmetric, submodular hypergraphs. Note that a submodular function $f : 2^V \rightarrow \mathbb{R}^+$ is said to be symmetric if $\forall S \subseteq V, f(S) = f(V - S)$.

► **Theorem 16** (Corollary 1.2 of [5]). For any symmetric submodular hypergraph H on n vertices, there is a $(1 \pm \varepsilon)$ -sparsifier for H with $\tilde{O}(n/\varepsilon^2)$ hyperedges.

3 Directed to Undirected Hypergraph Sparsification

In this section, we will show that any algorithm that produces an undirected spectral hypergraph sparsifier with $f(n, r)$ hyperedges (for a vertex set of size n , and maximum hyperedge size r), can be used in a *black-box manner* to create a spectral sparsifier with $f(n^2 + 1, r^2)$ hyperedges for any n -vertex directed hypergraph.

To this end, we first have to define the “lifting” operation from a directed hypergraph on n vertices to an undirected hypergraph on $n^2 + 1$ vertices.

► **Definition 17.** For a directed hypergraph $H = (V, E)$ on n vertices, let $\psi(H)$ be an undirected hypergraph on $n^2 + 1$ vertices defined as follows. For the first n^2 vertices of $\psi(H)$, associate these vertices with tuples of vertices from H , that is, each of these vertices is associated with an element from the set $V \times V$. The final vertex in $\psi(H)$ will be a special vertex we denote by $*$. Now, for each hyperedge $e \in E$ of H , define a corresponding hyperedge $\varphi(e)$ in $\psi(H)$ as follows: let the vertices in $L(e)$ be u_1, \dots, u_ℓ , and let the vertices in $R(e)$ be v_1, \dots, v_r . Let $\varphi(e)$ contain

$$L(e) \times R(e) \cup \{*\} = \{(u_1, v_1), (u_1, v_2), \dots, (u_1, v_r), (u_2, v_1), \dots, (u_2, v_r), \dots, (u_\ell, v_1), \dots, (u_\ell, v_r), *\}.$$

Note that this transformation is invertible. If we are given an undirected hyperedge of the form $\varphi(e) = L(e) \times R(e) \cup \{*\}$, we can invert this transformation to recover the directed hyperedge $e = (L(e), R(e))$. Additionally, note that this transformation and its inverse are efficiently computable (running in time $O(r^2)$, where r is the size of the undirected hyperedge).

Next, we define the lifting of a test vector.

► **Definition 18.** For a vector $x \in \mathbb{R}^n$, we define the lifting of x denoted as $\vartheta(x)$. $\vartheta(x)$ is in \mathbb{R}^{n^2+1} , and in particular, for the first n^2 entries, we associate these with the set $[n] \times [n]$. We say that $(\vartheta(x))_{u,v} = \max(x_u - x_v, 0)$. For the final entry, which we associate with the special vertex $*$ in the lifted H , we let $\vartheta(x)_* = 0$.

Note again that $\vartheta(x)$ is efficiently computable in time $O(n^2)$ where n is the dimension of x .

► **Theorem 19.** Let $H = (V, E)$ be a directed hypergraph on n vertices. Then, for any $x \in \mathbb{R}^n$,

$$L_H(x) = L_{\psi(H)}(\vartheta(x)).$$

Proof. It suffices to show that for a single hyperedge $e \in E$,

$$\max_{u \in L(e), v \in R(e)} (x_u - x_v)_+^2 = \max_{(y, z) \in \varphi(e)} (\vartheta(x)_y - \vartheta(x)_z)^2.$$

The reason this suffices is that there is one $\varphi(e)$ for each corresponding hyperedge $e \in E$. So, we are in effect showing that every term in the sum of the quadratic form of the Laplacians is the same.

To see why this equality is true, let some $\hat{u} \in L(e), \hat{v} \in R(e)$ be the maximizers for the expression on the left. Then, note that the corresponding entry $\vartheta(x)_{\hat{u}, \hat{v}}$ is exactly $(x_{\hat{u}} - x_{\hat{v}})_+$. Now, because $\hat{u} \in L(e)$ and $\hat{v} \in R(e)$, it follows that $(\hat{u}, \hat{v}) \in \varphi(e)$. Because the special vertex $* \in \varphi(e)$, it follows that in the above expression

$$\max_{(y, z) \in \varphi(e)} (\vartheta(x)_y - \vartheta(x)_z)^2 \geq (\vartheta(x)_{(\hat{u}, \hat{v})} - \vartheta(x)_*)^2 = (x_{\hat{u}} - x_{\hat{v}})_+^2 = \max_{u \in L(e), v \in R(e)} (x_u - x_v)_+^2.$$

Now, we will show the opposite direction. Indeed, suppose that some elements \hat{y}, \hat{z} are maximizers for $\max_{(y, z) \in \varphi(e)} (\vartheta(x)_y - \vartheta(x)_z)^2$. Note that by construction, every entry in $\vartheta(x)$ is ≥ 0 . This means that without loss of generality, we can assume that $\hat{z} = *$ (the special vertex), as this vertex attains the smallest possible value 0. This means that the maximizing value of the expression is exactly $\vartheta(x)_{\hat{y}}^2$, where \hat{y} is one of the first n^2 vertices in $\psi(H)$. So, let us write $\hat{y} = (\hat{a}, \hat{b})$, where \hat{a}, \hat{b} are both vertices in G . By construction, because $\hat{y} \in \hat{e}$, it follows that $\hat{a} \in L(e)$, and $\hat{b} \in R(e)$. As such it follows that

$$\max_{u \in L(e), v \in R(e)} (x_u - x_v)_+^2 \geq (x_{\hat{a}} - x_{\hat{b}})_+^2 = \vartheta(x)_{\hat{y}}^2 = \max_{(y, z) \in \hat{e}} (\vartheta(x)_y - \vartheta(x)_z)^2.$$

Thus, it follows that

$$\max_{u \in L(e), v \in R(e)} (x_u - x_v)_+^2 = \max_{(y, z) \in \varphi(e)} (\vartheta(x)_y - \vartheta(x)_z)^2,$$

as claimed. ◀

► **Corollary 20.** Let H be a directed hypergraph on n vertices. Suppose that $\widehat{\psi(H)}$ is a $(1 \pm \varepsilon)$ undirected hypergraph spectral sparsifier to $\psi(H)$. Then, it follows that the unlifted graph \widehat{H} which is calculated by applying φ^{-1} to each hyperedge in $\widehat{\psi(H)}$, is a $(1 \pm \varepsilon)$ directed hypergraph spectral sparsifier to H .

Proof. Indeed, suppose $H, \psi(H), \widehat{H}, \widehat{\psi(H)}$ are as specified above, and let $x \in \mathbb{R}^n$. It follows that

$$\begin{aligned} (1 - \varepsilon)L_H(x) &= (1 - \varepsilon)L_{\psi(H)}(\vartheta(x)) \leq L_{\widehat{\psi(H)}}(\vartheta(x)) = L_{\widehat{H}}(x) \\ &= L_{\widehat{\psi(H)}}(\vartheta(x)) \leq (1 + \varepsilon)L_{\psi(H)}(\vartheta(x)) = (1 + \varepsilon)L_H(x). \end{aligned}$$

To conclude, this implies that for \widehat{H}, H as above,

$$(1 - \varepsilon)L_H(x) \leq L_{\widehat{H}}(x) \leq (1 + \varepsilon)L_H(x).$$

► **Theorem 21.** *For a directed hypergraph H on n vertices, one can find a directed hypergraph spectral sparsifier \widehat{H} of H , with $O(n^2 \log(n) \log(r)/\varepsilon^2)$ hyperedges in time $\tilde{O}(mr^2)$, where m is the number of directed hyperedges in H and r is the maximum size of a hyperedge in H .*

Proof. If the number of hyperedges in H is less than n^2 , simply return H . Otherwise, lift H to $\psi(H)$, and spectrally sparsify $\psi(H)$ using [6]. This will result in a $(1 \pm \varepsilon)$ spectral sparsifier $\widehat{\psi(H)}$ to $\psi(H)$, with at most $O(n^2 \log(n^2) \log(r^2)/\varepsilon^2)$ hyperedges, as we desire. Here, we have used that the maximum rank of a hyperedge in \widehat{G} is at most the squared rank of a hyperedge in G . Further, the running time of this algorithm is $\tilde{O}(mr^2)$, as the number of hyperedges in \widehat{G} is the same as the number of hyperedges in G , and the rank, again, is at most r^2 . Now, we can unlift $\widehat{\psi(H)}$ to \widehat{H} by applying φ^{-1} to each hyperedge, and use the previous corollary to conclude our theorem. ◀

► **Remark 22.** Note that if we restrict our original vector x to be in $\{0, 1\}^n$, it follows that $\vartheta(x) \in \{0, 1\}^{n^2+1}$. By repeating the exact same steps above, this means that we can use the same reduction from above to get directed hypergraph cut sparsifiers, by only using algorithms from undirected hypergraph cut sparsifiers.

► **Corollary 23.** *For a directed hypergraph H on n vertices, one can find a directed hypergraph cut sparsifier \widehat{H} of H , with $O(n^2 \log(n)/\varepsilon^2)$ hyperedges in time $\tilde{O}(mr^2/\varepsilon^2)$, where m is the number of directed hyperedges in H , and r the maximum size of any hyperedge.*

Proof. Simply perform the reduction from above, and invoke the algorithm for undirected hypergraph cut-sparsification from Theorem 1.3 of [19]. Specifically, for an undirected hypergraph G , Quanrud presents an algorithm running in time $\tilde{O}(p(G))$ which constructs $(1 \pm \varepsilon)$ k -cut sparsifiers (a stronger requirement than the cut-sparsifiers required here), where $p(G) = \sum_{e \in G} |e|$. In our setting, for a directed hypergraph H , when we construct the undirected lifting $\psi(H)$, it will be the case that $p(\psi(H)) = \sum_{e \in \psi(H)} |e| \leq \sum_{e \in H} |e|^2 \leq mr^2$, yielding the desired runtime. ◀

4 Space Lower-bounds for Sketching Cuts in Directed Hypergraphs

In this section, we will establish an $\Omega(n^{3-o(1)})$ lower-bound for worst-case sketching of the cuts in a directed hypergraph on n vertices to a $(1 \pm \varepsilon)$ factor for ε being $\frac{1}{2^{O(\sqrt{\log(n)})}}$. As mentioned in the introduction, this improves upon a result of [10] who showed a lower bound of size $\Omega(n^3)$ for the bit complexity of any *sparsifier*. However, their lower bound explicitly takes advantage of the sparsifier structure by starting with known examples of sparsifiers that require $\Omega(n^2)$ hyperedges, and then padding these hyperedges with random vertices in their tail such that the bit complexity is $\Omega(n)$. One can trivially show that this padding does not change the requirement of preserving $\Omega(n^2)$ hyperedges. Because sparsifiers are limited to storing only hyperedges that were originally present, this then forces a bit complexity lower bound of $\Omega(n^3)$. However, this same technique is not amenable to a sketching lower bound as the padding procedure only adds complexity to each hyperedge, and not necessarily to the cut function as a whole.

To overcome this, we take advantage of a result of [8] who showed that, in general, any $(1 \pm \varepsilon)$ cut-sketching scheme for undirected hypergraphs on n vertices, with $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$ must have worst case bit complexity $\frac{n^2}{2^{O(\sqrt{\log(n)})}}$. This result uses encodings of Rusza-Szemerédi graphs into undirected hypergraphs, along with a reconstruction argument to show that general $(1 \pm \varepsilon)$ cut-sketching schemes in undirected hypergraphs give very non-trivial string

compression schemes. Then, by invoking known results on size lower bounds for string compression schemes, they are able to conclude worst-case lower bounds of $\frac{n^2}{2^{O(\sqrt{\log(n)})}}$ for the bit complexity of sketching cuts in undirected hypergraphs. To this end, we first reintroduce their notion of a string compression scheme:

► **Definition 24 ([4]).** Let ℓ, k be positive integers, and let $\varepsilon, g > 0$. We say that a pair of functions $\text{Encode} : \{0, 1\}^\ell \rightarrow \{0, 1\}^k$ and $\text{Decode} : \{0, 1\}^k \times 2^{[\ell]} \rightarrow \mathbb{N}$ is an $(\ell, k, \varepsilon, g)$ string compression scheme (SCS) if there exists a set of strings $\mathcal{G} \subseteq \{0, 1\}^\ell$ such that:

1. $|\mathcal{G}| \geq g \cdot 2^\ell$.
2. For every string $s \in \mathcal{G}$, and every query $q \in 2^{[\ell]}$,

$$|\text{Decode}(\text{Encode}(s), q) - |s \cap q|| \leq \varepsilon \ell / 2.$$

The work of [8] takes advantage of the following theorem, which is proved in [4]:

► **Theorem 25 ([4]).** Suppose $(\text{Encode}, \text{Decode})$ is an $(\ell, k, \varepsilon, g)$ -SCS, where $\varepsilon \leq 1/10$. Then,

$$k \geq \frac{\log(g) + 3\ell/50}{\log 2} - 1.$$

Qualitatively, [8] shows that for a specific family of *undirected* hypergraphs with n vertices, for some $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$ any $(1 \pm \varepsilon)$ cut-sketching scheme for these hypergraphs using $\leq k$ bits implicitly gives an $\left(\frac{n^2}{2^{O(\sqrt{\log(n)})}}, k, 1/10, 1/2\right)$ -SCS. Thus, by invoking the previous theorem, these sparsifiers must have bit complexity $\frac{n^2}{2^{O(\sqrt{\log(n)})}}$. However, their proof actually provides a stronger result than stated. Although the sparsifiers they use give $(1 \pm \varepsilon)$ multiplicative approximations to cut-sizes, their argument makes uses of an *additive* error bound of $\varepsilon \cdot (\# \text{ of hyperedges})$. We take advantage of this in our method by showing that a $(1 \pm \varepsilon)$ cut-sketch for a directed hypergraph can be used to retrieve cut sizes in n distinct undirected hypergraphs with only additive error ε (with respect to each of these undirected hypergraphs). We first state the result of [8] more succinctly, and then describe our construction in more detail.

► **Theorem 26 ([8]).** For any n , and some $\ell = \frac{n^2}{2^{O(\sqrt{\log(n)})}}$, for at least $2^\ell/2$ strings $s \in \{0, 1\}^\ell$, there exists an undirected hypergraph $H_s = (\tilde{V}, E_s)$ on n vertices, with $\leq n$ hyperedges, such that any data structure which can approximate cuts in H_s to within additive error $|E_s|/2^{O(\sqrt{\log(n)})}$ can for any query $q \subseteq [\ell]$, answer the subset sum $|q \cap s|$ to within additive error $\ell/20$.

Now, we will prove our theorem regarding capability of directed hypergraphs to simulate cuts in undirected hypergraphs with only additive error.

► **Theorem 27.** Given any undirected hypergraphs H_1, \dots, H_n , each on vertex set V , with $|V| = n$, there exists a directed hypergraph G on $2n$ vertices, such that given a $(1 \pm \varepsilon)$ cut-sketch for G , for any of the undirected hypergraphs $H_i = (V, E_i)$ and any set $S \subseteq V$, one can recover $|\text{cut}_{H_i}(S)|$ to within additive error $3\varepsilon|E_i|$.

Proof. As stated, each of the undirected hypergraphs H_1, \dots, H_n are on a vertex set of size n , which we denote by V . We also create a vertex set W of size n , which we associate with w_1, \dots, w_n . Now, we create the directed hypergraph G , which lives on the vertex set $V \cup W$ as follows: for each undirected hypergraph H_i for $i = 1, \dots, n$, and for each undirected hyperedge e in H_i , we add the corresponding directed hyperedge (e, w_i) . That is, the head of the directed hyperedge has the vertices from V corresponding to e , and the tail of the directed hyperedge has only vertex w_i .

Clearly, G has $2n$ vertices, so now it suffices to argue that for any $H_i = (V, E_i)$, and for any cut $S \subseteq V$, we can recover $\text{cut}_{H_i}(S)$ within additive error $\varepsilon|E_i|$. Indeed, let any such H_i be given, and let $S \subseteq V$ be given as well. Then, suppose we have a $(1 \pm \varepsilon)$ cut-sketch for G , which we denote by \tilde{G} . Let us consider the query to \tilde{G} with the set $S \cup W - \{w_i\}$. A directed hyperedge $e \in G$ is crossing this cut if and only if $e_{\text{head}} \cap (S \cup W - \{w_i\}) \neq \emptyset$ and $e_{\text{tail}} \cap ((V \cup W) - (S \cup W - \{w_i\})) \neq \emptyset$. In particular, note that by construction, e_{head} is a subset of V and e_{tail} is a subset of W . This means that a directed hyperedge e is crossing the cut if and only if $e_{\text{head}} \cap S \neq \emptyset$ and $e_{\text{tail}} \cap \{w_i\} \neq \emptyset$. The only directed hyperedges in G which satisfy this second condition are exactly those directed hyperedges in G which correspond to H_i . By construction, this means that the number of directed hyperedges crossing this cut $S \cup W - \{w_i\}$ in G is exactly the number of undirected hyperedges $e \in E_i$ such that $e \cap S \neq \emptyset$. Thus this query to \tilde{G} returns a $(1 \pm \varepsilon)$ approximation to $|\{e \in E_i | S \cap e \neq \emptyset\}|$. Note that the actual size of the cut S in H_i is $|\{e \in E_i | S \cap e \neq \emptyset \wedge S \cap e \neq e\}|$.

However, note that by symmetry, we can also query \tilde{G} with $(V - S) \cup (W - \{w_i\})$. By symmetry, this query to \tilde{G} returns a $(1 \pm \varepsilon)$ approximation to $|\{e \in E_i | (V - S) \cap e \neq \emptyset\}|$, which is the same as $|\{e \in E_i | S \cap e \neq e\}|$. Lastly, we can query \tilde{G} with $V \cup (W - \{w_i\})$. This query to \tilde{G} returns a $(1 \pm \varepsilon)$ approximation to $|\{e \in E_i | V \cap e \neq \emptyset\}|$, which is exactly $|\{e \in E_i\}|$.

Now, we operate by the principle of inclusion-exclusion (PIE). Let A be the event that a hyperedge $e \in E_i$ satisfies $e \cap S \neq \emptyset$, and let B be the event that e satisfies $e \cap S \neq e$. By PIE,

$$|\{e \in E_i | e \text{ satisfies } A \wedge \text{satisfies } B\}| = |\{e \in E_i | e \text{ satisfies } A\}| + |\{e \in E_i | e \text{ satisfies } B\}| - |\{e \in E_i | e \text{ satisfies } A \vee \text{satisfies } B\}|.$$

Note that this final expression is trivially satisfied, i.e. $|\{e \in E_i | e \text{ satisfies } A \vee \text{satisfies } B\}| = |\{e \in E_i\}|$ as a hyperedge cannot simultaneously have an empty and a non-trivial intersection. Thus, we get that

$$\begin{aligned} \text{cut}_{H_i}(S) &= |\{e \in E_i | e \text{ satisfies } A \wedge \text{satisfies } B\}| \\ &= |\{e \in E_i | e \text{ satisfies } A\}| + |\{e \in E_i | e \text{ satisfies } B\}| - |\{e \in E_i\}|. \end{aligned}$$

Now, note that our query to \tilde{G} with the set $S \cup W - \{w_i\}$ gave us a $(1 \pm \varepsilon)$ approximation to $|\{e \in E_i | e \text{ satisfies } A\}|$, our query with $(V - S) \cup W - \{w_i\}$ gave us a $(1 \pm \varepsilon)$ approximation to $|\{e \in E_i | e \text{ satisfies } B\}|$, and our query with $V \cup W - \{w_i\}$ gave us a $(1 \pm \varepsilon)$ approximation to $|\{e \in E_i\}|$. Because each of these has additive error at most $\varepsilon|E_i|$ (as the error from \tilde{G} is a multiplicative guarantee), in total, the expression

$$\text{cut}_{\tilde{G}}(S \cup W - \{w_i\}) + \text{cut}_{\tilde{G}}((V - S) \cup W - \{w_i\}) - \text{cut}_{\tilde{G}}(V \cup W - \{w_i\})$$

gives us a $(3\varepsilon|E_i|)$ -additive approximation to $\text{cut}_{H_i}(S)$, as we desire. \blacktriangleleft

Now, we will show how we can use the above construction to argue a lower bound of size $\frac{n^3}{2^{O(\sqrt{\log(n)})}}$ on the bit complexity of directed hypergraph cut-sketching. We will do this by showing that we can use a directed hypergraph cut-sketch of size k to create a $(\ell, k, 1/10, 2^{-n})$ -SCS, for $\ell = \Omega\left(\frac{n^3}{2^{O(\sqrt{\log(n)})}}\right)$.

► **Theorem 28.** *A general unweighted directed hypergraph $(1 \pm \frac{1}{2^{O(\sqrt{\log(n)})}})$ cut-sketching scheme on n vertices with maximum sketch size of k bits yields an $(n \cdot \ell, k, 1/10, 2^{-n})$ -SCS for $\ell = \frac{n^2}{2^{O(\sqrt{\log(n)})}}$.*

Proof. First, we will define the set \mathcal{G} of size $\frac{2^{n \cdot \ell}}{2^n}$. Indeed, from Theorem 26, let L be the strings of length ℓ which are able to be compressed and still allow for estimating subset sum queries. Now, let $\mathcal{G} = L \circ L \circ L \circ \dots \circ L$ (n times), where the $S_1 \circ S_2$ operation takes every string in S_1 and prepends it to every string in S_2 (resulting in a new set of size $|S_1| \cdot |S_2|$). Note that this means that strings in \mathcal{G} will be of length $n \cdot \frac{n^2}{2^{O(\sqrt{\log(n)})}} = \frac{n^3}{2^{O(\sqrt{\log(n)})}}$ and \mathcal{G} obtains the stated size bound.

Now we describe our string compression scheme. Indeed, for any string $s \in \mathcal{G}$, decompose s into s_1, \dots, s_n such that each $s_i \in L$. Now, because each $s_i \in L$, we know there exists a corresponding undirected hypergraph $H_{s_i} = (V, E_{s_i})$ on n vertices such that preserving cuts in H_{s_i} to within additive error $|E_{s_i}|/2^{O(\sqrt{\log(n)})}$ allows us to answer subset sum queries in H_{s_i} to within additive error $\ell/20$. Now let G be the directed hypergraph on $2n$ vertices, built with hypergraphs $H_{s_1}, H_{s_2}, \dots, H_{s_n}$ as guaranteed by Theorem 27. It follows that G is an unweighted directed hypergraph on $2n$ vertices.

Now, suppose there exists a general, unweighted, directed hypergraph cut-sketching scheme on n vertices with maximum sketch size of k bits which preserves cuts to a $(1 \pm \frac{1}{2^{O(\sqrt{\log(n)})}})$ multiplicative factor. Then, we can invoke such a scheme on the directed hypergraph G as specified by Theorem 27 to conclude that such a scheme allows us to recover $\text{cut}_{H_{s_i}}(S)$ for any $S \subseteq V$ to within additive error $|E_{s_i}|/2^{O(\sqrt{\log(n)})}$. As a result, this means that for any s_i , and any query to s_i , denoted by $q_i \in [\ell]$, we can recover $|q_i \cap s_i|$ to within additive error $\ell/20$.

Finally, suppose we are given any subset query $q \subseteq [n \cdot \ell]$. We want to show that we can compute the size of $|s \cap q|$ (i.e. the sum of the bits of s on the positions indicated by q) to within additive error $\frac{n\ell}{20}$. For convenience, we view q as a bit string of length $n \cdot \ell$, where a bit is 1 if and only if the corresponding element of $[n \cdot \ell]$ was in the subset. Then, we break q into q_1, \dots, q_n such that each q_i is of length ℓ . Now, we use the aforementioned sketch to compute $|s_i \cap q_i|$ to within additive error $\ell/20$ for every i . Adding these together, we get an estimate to $|s \cap q|$ with additive error at most $n\ell/20$. Thus, a general directed hypergraph cut-sketching scheme of size k bits to multiplicative error $(1 \pm \frac{1}{2^{O(\sqrt{\log(n)})}})$ yields a $(n \cdot \ell, k, 1/10, 2^{-n})$ -SCS. ◀

► **Theorem 29.** Any cut-sketching scheme for directed hypergraphs on $2n$ vertices which preserves cuts to a $(1 \pm \varepsilon)$ factor, for $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$ must have worst case bit complexity $\frac{n^3}{2^{O(\sqrt{\log(n)})}}$.

Proof. Indeed, by the preceding theorem (Theorem 28), any such scheme for $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$, with bit complexity k implies a $(n \cdot \ell, k, 1/10, 2^{-n})$ -SCS, for $\ell = \frac{n^2}{2^{O(\sqrt{\log(n)})}}$. By Theorem 25 [4], this means that

$$k \geq \frac{\log(2^{-n}) + 3n \cdot \ell}{\log 2} - 1 \geq \frac{n^3}{2^{O(\sqrt{\log(n)})}}.$$
 ◀

► **Corollary 30.** Any cut-sketching scheme for submodular hypergraphs on $2n$ vertices which preserves cuts to a $(1 \pm \varepsilon)$ factor, for $\varepsilon = \frac{1}{2^{O(\sqrt{\log(n)})}}$ must have bit complexity $\frac{n^3}{2^{O(\sqrt{\log(n)})}}$.

Proof. This follows simply by noting that directed hypergraphs are a subclass of submodular hypergraphs, so in particular the lower bound from Theorem 29 must extend to this case. ◀

5 Monotone Hypergraph Sparsifiers

In this section, we show how to reduce sparsifying monotone submodular hypergraphs to sparsifying symmetric submodular hypergraphs. At this point, we then invoke the result of [5] to conclude. First, we detail the reduction:

► **Theorem 31.** Suppose $f : 2^V \rightarrow \mathbb{R}^{\geq 0}$ is a monotone, submodular function. Then, $f' : 2^{V \cup \{*\}} \rightarrow \mathbb{R}^{\geq 0}$ defined as $\forall S \subseteq V$

$$f'(S) = f(S) = f'(V - S \cup \{*\})$$

is submodular and symmetric.

Proof. First, the symmetry of f' is easy to see. Indeed, for any set $S \subseteq V \cup \{*\}$, it follows that $f'(S) = f'(V \cup \{*\} - S)$. So, all that remains to be shown is that f' is submodular. To do this, we will show that f' has decreasing marginals. So consider any $T \subset U \subset V \cup \{*\}$. We will show that for any $x \notin U$ that

$$f'(T \cup \{x\}) - f'(T) \geq f'(U \cup \{x\}) - f'(U).$$

We will do this by cases.

1. Suppose that $x = *$. Then, it must be the case that $x \notin U, T$. So, $f'(U) = f(U), f'(T) = f(T)$. Because $T \subset U$, it must also therefore be the case that $f'(T) \leq f'(U)$ (by the monotonicity of f). Next, we note that because $x = *$, $f'(T \cup \{x\}) = f(V - T), f'(U \cup \{x\}) = f(V - U)$. Because $T \subset U$ and f is monotone, it must be the case that $f'(T \cup \{x\}) = f(V - T) \geq f'(U \cup \{x\}) = f(V - U)$. Putting this together, we get that it must be the case that

$$f'(T \cup \{x\}) - f'(T) \geq f'(U \cup \{x\}) - f'(U),$$

as we desire.

2. Suppose that $x \neq *$, and that neither U, T contain $*$. Then the submodularity of f' follows by the submodularity of f .
3. Suppose that $x \neq *$, and that both U, T contain $*$. Then, let \hat{U}, \hat{T} be $U - \{*\}, T - \{*\}$ respectively. It follows that $\hat{T} \subset \hat{U}$. Further, $f'(T \cup \{x\}) = f(V - (\hat{T} \cup \{x\}))$, $f'(U \cup \{x\}) = f(V - (\hat{U} \cup \{x\}))$, and likewise $f'(T) = f(V - \hat{T}), f'(U) = f(V - \hat{U})$. It follows that

$$\begin{aligned} f'(T \cup \{x\}) - f'(T) &= f(V - (\hat{T} \cup \{x\})) - f(V - \hat{T}) \\ &= f(V - (\hat{T} \cup \{x\})) - f(V - (\hat{T} \cup \{x\}) \cup \{x\}) \\ &\geq f(V - (\hat{U} \cup \{x\})) - f(V - (\hat{U} \cup \{x\}) \cup \{x\}) \\ &= f'(U \cup \{x\}) - f'(U). \end{aligned}$$

The inequality in the middle holds because $V - (\hat{U} \cup \{x\}) \subset V - (\hat{T} \cup \{x\})$. Thus, the marginal gain from adding x to $V - (\hat{U} \cup \{x\})$ is larger than the marginal gain from adding x to $V - (\hat{T} \cup \{x\})$ by the submodularity of f .

4. Suppose that $x \neq *$, but that $* \notin T, * \in U$. Then, by the monotonicity of f , $f'(T \cup \{x\}) - f'(T) = f(T \cup \{x\}) - f(T) \geq 0$. Likewise,

$$f'(U \cup \{x\}) - f'(U) = f(V - (\hat{U} \cup \{x\})) - f(V - \hat{U}) \leq 0,$$

again using the monotonicity of f . Therefore, it must be the case that

$$f'(T \cup \{x\}) - f'(T) \geq f'(U \cup \{x\}) - f'(U),$$

as we desire. ◀

Next, we show how to use this reduction to create sparsifiers.

► **Corollary 32.** *Let $H = (V, E)$ be a hypergraph, such that $\forall e \in E$, the corresponding splitting function $g_e : 2^e \rightarrow \mathbb{R}^{\geq 0}$ is submodular and monotone. Then there exists a $(1 \pm \varepsilon)$ cut-sparsifier for H with $\tilde{O}(|V|/\varepsilon^2)$ hyperedges.*

Proof. We first define the lifting of a monotone, submodular hypergraph into a symmetric submodular hypergraph.

► **Definition 33.** *Let $H = (V, E)$ be a monotone submodular hypergraph. Then, define H' to be the corresponding hypergraph defined on vertex set $V \cup \{*\}$, where for each edge $e \in E$, we replace it with a hyperedge $e' = e \cup \{*\}$, and replace the function g_e with the symmetric, submodular splitting function $g'_e : 2^{e'} \rightarrow \mathbb{R}^{\geq 0}$ defined in accordance with Theorem 31.*

Now, we construct this hypergraph H' . Because each g'_e is symmetric and submodular, we can invoke Theorem 16 to conclude the existence of a hypergraph \hat{H}' such that $\forall S \subseteq V \cup \{*\}$

$$(1 - \varepsilon)\text{cut}_{H'}(S) \leq \text{cut}_{\hat{H}'}(S) \leq (1 + \varepsilon)\text{cut}_{H'}(S),$$

and \hat{H}' only has $\tilde{O}(|V|/\varepsilon^2)$ hyperedges remaining.

It follows that because $\forall S \subseteq V$, $g'_e(S) = g_e(S)$, the corresponding hyperedges chosen to create a $(1 \pm \varepsilon)$ cut-sparsifier for H' also create a $(1 \pm \varepsilon)$ cut-sparsifier for H . That is, if we create the hypergraph \hat{H} by replacing $e' \in \hat{H}'$ with $e \in H$ (but keeping the same corresponding weights that \hat{H}' assigns), it will be the case that $\forall S \subseteq V$

$$(1 - \varepsilon)\text{cut}_{H'}(S) = (1 - \varepsilon)\text{cut}_H(S) \leq \text{cut}_{\hat{H}'}(S) = \text{cut}_{\hat{H}}(S) \leq (1 + \varepsilon)\text{cut}_{H'}(S) = (1 + \varepsilon)\text{cut}_H(S).$$

Thus, \hat{H} will be a $(1 \pm \varepsilon)$ -sparsifier for H , and \hat{H} will only keep $\tilde{O}(|V|/\varepsilon^2)$ hyperedges. ◀

References

- 1 Joshua D. Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-ramanujan sparsifiers. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 255–262. ACM, 2009. doi:10.1145/1536414.1536451.
- 2 András A. Benczúr and David R. Karger. Approximating s - t minimum cuts in $\tilde{O}(n^2)$ time. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 47–55. ACM, 1996. doi:10.1145/237814.237827.
- 3 Yu Chen, Sanjeev Khanna, and Ansh Nagda. Near-linear size hypergraph cut sparsifiers. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 61–72. IEEE, 2020. doi:10.1109/FOCS46700.2020.00015.
- 4 Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In Frank Neven, Catriel Beeri, and Tova Milo, editors, *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 9-12, 2003, San Diego, CA, USA*, pages 202–210. ACM, 2003. doi:10.1145/773153.773173.
- 5 Arun Jambulapati, James R. Lee, Yang P. Liu, and Aaron Sidford. Sparsifying sums of norms. *CoRR*, abs/2305.09049, 2023. doi:10.48550/arXiv.2305.09049.
- 6 Arun Jambulapati, Yang P. Liu, and Aaron Sidford. Chaining, group leverage score overestimates, and fast spectral hypergraph sparsification. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 196–206. ACM, 2023. doi:10.1145/3564246.3585136.

- 7 Michael Kapralov, Robert Krauthgamer, Jakab Tardos, and Yuichi Yoshida. Spectral hypergraph sparsifiers of nearly linear size. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1159–1170. IEEE, 2021. doi:10.1109/FOCS52979.2021.00114.
- 8 Michael Kapralov, Robert Krauthgamer, Jakab Tardos, and Yuichi Yoshida. Towards tight bounds for spectral sparsification of hypergraphs. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 598–611. ACM, 2021. doi:10.1145/3406325.3451061.
- 9 David R. Karger. Global min-cuts in rnc, and other ramifications of a simple min-cut algorithm. In Vijaya Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 21–30. ACM/SIAM, 1993. URL: <http://dl.acm.org/citation.cfm?id=313559.313605>.
- 10 Yotam Kenneth and Robert Krauthgamer. Cut sparsification and succinct representation of submodular hypergraphs. *CoRR*, abs/2307.09110, 2023. arXiv:2307.09110.
- 11 Sanjeev Khanna, Aaron Puterman, and Madhu Sudan. Code sparsification and its applications. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5145–5168. SIAM, 2024.
- 12 Dmitry Kogan and Robert Krauthgamer. Sketching cuts in graphs and hypergraphs. In Tim Roughgarden, editor, *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, ITCS 2015, Rehovot, Israel, January 11-13, 2015*, pages 367–376. ACM, 2015. doi:10.1145/2688073.2688093.
- 13 Jannik Kudla and Stanislav Zivný. Sparsification of monotone k-submodular functions of low curvature. *CoRR*, abs/2302.03143, 2023. doi:10.48550/arXiv.2302.03143.
- 14 James R. Lee. Spectral hypergraph sparsification via chaining. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 207–218. ACM, 2023. doi:10.1145/3564246.3585165.
- 15 Pan Li and Olgica Milenkovic. Inhomogeneous hypergraph clustering with applications. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2308–2318, 2017. URL: <https://proceedings.neurips.cc/paper/2017/hash/a50abba8132a77191791390c3eb19fe7-Abstract.html>.
- 16 Pan Li and Olgica Milenkovic. Submodular hypergraphs: p-laplacians, cheeger inequalities and spectral clustering. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 3020–3029. PMLR, 2018. URL: <http://proceedings.mlr.press/v80/li18e.html>.
- 17 Hui Lin and Jeff A. Bilmes. A class of submodular functions for document summarization. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 510–520. The Association for Computer Linguistics, 2011. URL: <https://aclanthology.org/P11-1052/>.
- 18 Kazusato Oko, Shinsaku Sakaue, and Shin-ichi Tanigawa. Nearly tight spectral sparsification of directed hypergraphs. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPICS*, pages 94:1–94:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.94.
- 19 Kent Quanrud. *Quotient sparsification for submodular functions*, pages 5209–5248. SIAM, 2024. doi:10.1137/1.978161977912.187.

- 20 Tasuku Soma and Yuichi Yoshida. Spectral sparsification of hypergraphs. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2570–2581. SIAM, 2019. doi:10.1137/1.9781611975482.159.
- 21 Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM J. Comput.*, 40(4):981–1025, 2011. doi:10.1137/08074489X.
- 22 Nate Veldt, Austin R. Benson, and Jon M. Kleinberg. Minimizing localized ratio cut objectives in hypergraphs. In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1708–1718. ACM, 2020. doi:10.1145/3394486.3403222.
- 23 Nate Veldt, Austin R. Benson, and Jon M. Kleinberg. Approximate decomposable submodular function minimization for cardinality-based components. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 3744–3756, 2021. URL: <https://proceedings.neurips.cc/paper/2021/hash/1e8a19426224ca89e83cef47f1e7f53b-Abstract.html>.