Hypergraph Spectral Analysis and Processing in 3D Point Cloud

Songyang Zhang, Shuguang Cui, *Fellow, IEEE*, and Zhi Ding, *Fellow, IEEE*

***Abstract*—Along with increasingly popular virtual reality applications, the three-dimensional (3D) point cloud has become a fundamental data structure to characterize 3D objects and surroundings. To process 3D point clouds efficiently, a suitable model for the underlying structure and outlier noises is always critical. In this work, we propose a hypergraph-based new point cloud model that is amenable to efficient analysis and processing. We introduce tensor-based methods to estimate hypergraph spectrum components and frequency coefficients of point clouds in both ideal and noisy settings. We establish an analytical connection between hypergraph frequencies and structural features. We further evaluate the efficacy of hypergraph spectrum estimation in two common applications of sampling and denoising of point clouds for which we provide specific hypergraph filter design and spectral properties. Experimental results demonstrate the strength of hypergraph signal processing as a tool in characterizing the underlying properties of 3D point clouds.**

***Index Terms*—3D point clouds, hypergraph signal processing, hypergraph construction, denoising, sampling.**

I. Introduction

Recent developments in depth sensors and softwares make it easier to capture the features and create a three-dimensional (3D) model for an object and its surroundings [1]. In particular, with low-cost scanners such as light detection and ranging (LIDAR) and Kinect, a new data structure known as the point cloud has achieved significant successes in many areas, including virtual reality, geographic information system, reconstruction of art document and high-precision 3D maps for self-driving cars [2]. A point cloud consists of 3D coordinates with attributes such as color, temperature, texture, and depth [3]. Owing to the easy access to scanning sensors and the huge need in describing the 3D features, the use of point clouds has attracted significant attentions in areas of computer vision, virtual reality, and medical science. How to process the point clouds efficiently becomes an important topic of research in many 3D imaging and vision systems.

To analyze the features of point cloud, the first step is to construct an analytical model to represent the 3D structures. The literature provides several different models. In [4], the 3D space is partitioned into several boxes or voxels, and the point clouds are then discretized therein. One disadvantage of voxels is that a dense grid is required to achieve fine resolution, leading to spatial inefficiency [3]. A spatially efficient approach [5], [6] is the octree representation of point clouds. An octree is a tree data structure in which each node has exactly eight children. It can partition a 3D space recursively, and represent the point clouds with partitioned boxes. Although efficient, octree suffers from discretization errors [3]. The bd-tree is another spatial decomposition technique and is robust in highly cluttered point cloud dataset. However, compared to octree structures, bd-trees are more difficult to update.

Recently, graphs and graph signal processing (GSP) have found applications in modeling point clouds. For example, the authors of [3] construct a graph based on pairwise point distances. Some other works, such as [8], [9], construct graphs based on the k-nearest neighbors, where each vertex (point) has an edge connection to its k nearest neighbors. There are several clear connections between graph features and point cloud characteristics. For example, the smoothness over a graph can describe the flatness of surfaces in point clouds. GSP-based tools such as filters and graph learning methods can process the point clouds and have shown great success because of the graph model's ability to capture the underlying geometric structures. However, graph-based methods still face some challenges, such as limited orders and measurement inefficiency. In a traditional graph, each edge can only connect two nodes, constraining graph-based models to describe only pairwise relationships. However, a multilateral relationship among multiple nodes is far more informative in a point cloud model. For example, points (i.e., nodes) on the same surface of a point cloud exhibit a strong multilateral relationship, which cannot be easily captured by an edge of a traditional graph. In fact, construction of an efficient graph for a given dataset is always an open question. Thus, studies on point clouds can benefit from more general and efficient models.

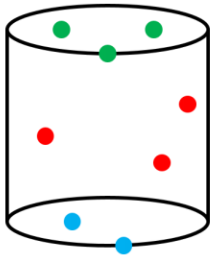
To develop an efficient model for point clouds, we explore a high-dimensional graph model, known as hypergraph [10]. Hypergraph can be a useful model in processing 3D point clouds. A hypergraph $\mathcal{H} = \{\mathcal{V}, \mathcal{E}\}$ consists of a set of nodes $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_K\}$ and a set of hyperedges $\mathcal{E} = \{\mathbf{e}_1, \ldots, \mathbf{e}_K\}$. Each hyperedge in a hypergraph can connect more than two nodes. Obviously, a normal graph is a special case of a hypergraph, where each hyperedge degrades to connect two nodes exactly. The hyperedge in a hypergraph can characterize the multilateral relationship among several related nodes (e.g., on a surface), thereby making hypergraph a natural

This material is based upon work supported by the National Science Foundation under Grant No. 1824553 and Grant No. 2002937.

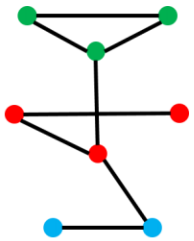
The work was also supported in part by the National Key R&D Program of China with grant No. 2018YFB1800800, by the Key Area R&D Program of Guangdong Province with grant No. 2018B030338001, by Shenzhen Outstanding Talents Training Fund, and by Guangdong Research Project No. 2017ZT07X152.

S. Zhang, and Z. Ding are with Department of Electrical and Computer Engineering, Univerisity of California, Davis, CA, 95616. (E-mail: sydzhang@ucdavis.edu, and zding@ucdavis.edu).

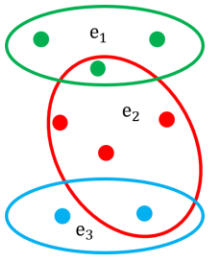
S. Cui is currently with the Shenzhen Research Institute of Big Data and Future Network of Intelligence Institute (FNii), the Chinese University of Hong Kong, Shenzhen, China, 518172 (e-mail: shuguangcui@cuhk.edu.cn).



(a) A 3D Shape with Eight Nodes: Green in the Top, Red in the Side, Blue in the Bottom.



(b) Distance-based Graph Model with Eight Edges.



(c) Hypergraph Model with Three Hyperedges.

Fig. 1. Examples of Geometric Models of Point Clouds.

and intuitive model for point clouds. For example, a 3D shape together with its geometric models are shown as Fig. 1. Since each edge only connects two nodes as nodes are treated equivalently in a traditional graph as Fig. 1(b), it is hard to distinguish from the graph structure which surface a node belongs to. However, from the hypergraph model in Fig. 1(c), we can easily identify surfaces of nodes according to high-dimensional hyperedges. Furthermore, advances in hypergraph signal processing (HGSP) [10] are providing more hypergraph tools to process high-dimensional cross-features and multilateral interactions among nodes, such as HGSP-based filters and spectral analysis, for effective point cloud processing.

However, processing the point clouds based on hypergraphs still poses several challenges. Similar to GSP, the first problem lies in the construction of hypergraph for point clouds. The traditional hypergraph construction method for a general dataset relies on data structure. For example, in [12], a hypergraph model is constructed according to the sentence structure in natural language processing. The k-nearest neighbor model is another method to construct the hypergraph. In [10], a hypergraph can be formed from the feature distances for an animal dataset to achieve clustering. However, such distance-based or structure-based model may be rather lossy in information preservation. For example, the structure-based method may not preserve the correlation of some irregular structures, whereas the k-nearest neighbor method may narrowly emphasize the distance information. In addition to hypergraph construction, another issue in analyzing point cloud with hypergraph tools is the computation complexity of the spectrum space. In the HGSP framework, spectrum-based analysis plays an important role but needs to compute the spectral space. Usually, the computation of hypergraph spectrum is based on orthogonal-CP decomposition [16], which incurs high-complexity when there are many nodes. Another challenge in point cloud processing is the effect of noise and outliers. Since a hypergraph model is constructed from observed data, noise can distort the hypergraph and degrade the performances of HGSP. Thus, mitigating noise effect and robustly estimating the hypergraph model for point clouds pose a significant challenge.

This work addresses the aforementioned problems. We propose novel spectrum-based hypergraph construction methods for both clean and noisy point clouds. For clean point clouds, we first estimate their spectrum components based on the hypergraph stationary process and optimally determine their frequency coefficients based on smoothness to recover the original hypergraph structure. For noisy point clouds, we introduce a method for joint hypergraph structure estimation and data denoising. We shall illustrate the effectiveness of the proposed hypergraph construction and spectrum estimation in two point cloud applications: sampling and denoising. Our experimental results clearly establish a connection between hypergraph frequencies and point cloud features. The performance improvement in both applications demonstrates the strength and power of hypergraph in point cloud processing and the practical value of our estimation methods.

We organize the rest of the paper as follows. In Section II, we lay the foundation with respect to the preliminaries and notations of point clouds, tensor basics and hypergraph signal processing. Next, we propose means in estimating hypergraph spectrum for basic point clouds in Section III and further develop means for hypergraph structure estimation of noisy point clouds in Section IV. With the proposed estimation methods, we study two important application scenarios and establish the effectiveness of hypergraph signal processing in Section V. Finally, we present the conclusion and future directions in Section VI.

II. Preliminaries and Notations

In this section, we cover basic background with respect to point cloud, tensor basics and hypergraph signal processing.

A. Point Clouds

A point cloud is a set of 3D points obtained from sensors or generated synthetically, where each point is attributed with coordinates and other features, such as color [11]. Since the 3D coordinates are basic features of a point cloud, in this work, we primarily focus on point clouds characterized by their coordinates. We consider a matrix representation of the point clouds, where a point cloud with N nodes is denoted by a location matrix

$$\mathbf{s} = [\mathbf{X_1} \quad \mathbf{X_2} \quad \mathbf{X_3}] = \begin{bmatrix} \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \cdots \\ \mathbf{s}_N^T \end{bmatrix} \in \mathbb{R}^{N\times 3}, \tag{1}$$

where $\mathbf{X}_i$ denotes a vector of the ith coordinates of all the points, and $\mathbf{s}_i$ is the three coordinates of ith point. With the information of coordinates, different models, such as graphs [3] and octrees [5], can be constructed to analyze the point clouds, for which we will discuss more in Section V.

B. Tensor Basics

Tensor is a high-dimensional generalization of matrix. A tensor can be interpreted as multi-dimensional arrays. The order of tensor is the number of indices to label the components of arrays [14]. For example, a scalar is a zeroth-order tensor; a vector is a first-order tensor; a matrix is a second-order tensor; and an M-dimensional array is an Mth-order tensor [15]. In this work, an Mth-order tensor is denoted by $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$, whose entry in position $(i_1, i_2, \cdots, i_M)$ is labeled as $a_{i_1 \cdots i_M}$. Here, I_k is the dimension of kth order.

Tensor outer product is a widely used operation to construct a higher-order tensor from lower-order tensors. The tensor outer product between an Mth-order tensor $\mathbf{U} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M}$ with entries $u_{i_1 \ldots i_M}$ and an Nth-order tensor $\mathbf{V} \in \mathbb{R}^{J_1 \times J_2 \times \ldots \times J_N}$ with entries $v_{j_1 \ldots j_N}$ is denoted by

$$\mathbf{W} = \mathbf{U} \circ \mathbf{V}, \tag{2}$$

where the result $\mathbf{W} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M \times J_1 \times J_2 \times \ldots \times J_N}$ is an $(M+N)$th-order tensor with entries

$$w_{i_1 \ldots i_M j_1 \ldots j_N} = u_{i_1 \ldots i_M} \cdot v_{j_1 \ldots j_N}. \tag{3}$$

C. Hypergraph Signal Processing

Hypergraph signal processing (HGSP) is a tensor-based framework [10]. In the HGSP framework, a hypergraph with N nodes and longest hyperedge connecting M nodes, is represented by an M-th order N-dimension representing tensor $\mathbf{A} = (a_{i_1 i_2 \cdots i_M}) \in \mathbb{R}^{N^M}$. Note that, the integer M is the maximal cardinality and each hyperedge can only connect at most M nodes. For those hyperedges with fewer than M nodes, a normalization scheme is applied based on permutation and combination. Interested readers are referred to [13] for more discussions on hyperedge normalization and weight calculation. The representing tensor can be an adjacency tensor or Laplacian tensor in different purposes [13]. In this paper, we refer the adjacency tensor as the representing tensor, in which each entry $a_{i_1 i_2 \cdots i_M}$ indicates whether nodes $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_M\}$ are connected in the normalized hyperedges.

With the orthogonal CANDECOMP/PARAFAC (CP) decomposition [16] - [18], the representing tensor can be decomposed via

$$\mathbf{A} \approx \sum_{r=1}^{N} \lambda_r \cdot \underbrace{\mathbf{f}_r \circ \ldots \circ \mathbf{f}_r}_{M \text{ times}}, \tag{4}$$

where $\mathbf{f}_r$'s are orthonormal basis vectors called spectrum components and λ_r are frequency coefficients related to the hypergraph frequency. All the spectrum components $\{\mathbf{f}_1, \cdots, \mathbf{f}_N\}$ construct the hypergraph spectral space. Each pair $(\mathbf{f}_r, \lambda_r)$ is called the spectral pair of the hypergraph, which is also the E-eigenpair of the representing tensor [10].

Given an original signal $\mathbf{s} = [s_1 \quad s_2 \quad \ldots \quad s_N]^{\mathrm{T}}$, the hypergraph signal is defined as the $(M-1)$ times tensor outer product of $\mathbf{s}$, i.e.,

$$\mathbf{s}^{[M-1]} = \underbrace{\mathbf{s} \circ \ldots \circ \mathbf{s}}_{M-1 \text{ times}}. \tag{5}$$

The hypergraph frequencies are ordered by the total variation of the spectrum component, which is defined as

$$\mathbf{TV}(\mathbf{f_r}) = ||\mathbf{f}_r - \frac{1}{\lambda_{max}} \mathbf{A}\mathbf{f}_r^{[M-1]}||_k, \tag{6}$$

where $\mathbf{A}\mathbf{f}_r^{[M-1]}$ is the contraction between representing tensor $\mathbf{A}$ and the hypergraph signal.

A spectrum component with larger total variation is a higher-frequency component, which indicates a faster propagation over the given hypergraph. Moreover, a supporting matrix

$$\mathbf{P_s} = \frac{1}{\lambda_{\max}} \begin{bmatrix} \mathbf{f}_1 & \cdots & \mathbf{f}_N \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{bmatrix} \begin{bmatrix} \mathbf{f}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{f}_N^{\mathrm{T}} \end{bmatrix}, \tag{7}$$

can be defined to capture the overall spectral information of the hypergraph.

Instead of reviewing many properties of HGSP here, other aspects such as hypergraph Fourier transform, hypergraph filter design and sampling theory can be found in [10].

III. Hypergraph Spectrum Estimation for Point Clouds

To process the 3D point clouds, the first step is to construct an optimal hypergraph to model the point clouds. As we mentioned in the Section I, it is time-comsuming and inefficient to first construct a hypergraph structure before tensor decomposition to obtain the hypergraph spectrum. Instead, we propose to directly estimate the hypergraph spectral pairs based on the observed data, and then recover the original representing tensor with Eq. (4). In this section, we first estimate the hypergraph spectrum components $\mathbf{f}_r$'s based on the hypergraph stationary process, and optimize the frequency coefficients λ_r's based on the smoothness for original point clouds.

A. Estimation of Hypergraph Spectrum Components

In this part, we propose a method to estimate the hypergraph spectral components based on the hypergraph stationary process.

1) Hypergraph Stationary Process: Before providing details of the estimation, let us first introduce some new definitions and properties necessary for spectrum estimation.

Stationarity is a cornerstone property that facilities the analysis of random signals and observations in traditional signal processing [19]. It has equal importance in graph and hypergraph signal processing. Based on graph shifting introduced in [23], a definition of graph stationary process proposed in [19] can analyze the properties of the different observations of nodes, or the random signals over the graphs. Furthermore, [22] introduces a method to estimate the graph spectrum space

and graph diffusion for multiple observations based on the graph stationary process. Similarly, the hypergraph stationary process can be defined to estimate hypergraph spectrum.

Now, let us introduce the definition of the hypergraph stationary process. In [10], a polynomial hypergraph filter based on supporting matrix is defined as

$$\mathbf{s}' = \sum_{k=1}^{a} \alpha_k \mathbf{P}^k \mathbf{s}, \tag{8}$$

where $\mathbf{P} = \lambda_{max}\mathbf{P}_\mathbf{s}$.

Similarly, based on the supporting matrix, a τ-step shifting operation is defined as $\mathbf{P}_\tau = \mathbf{P}^\tau$. Then, similar to the definition of the stationary process in traditional digital signal processing and graph signal processing, a strict-sense stationary process in HGSP can be defined as follows.

Definition 1. *(Strict-Sense Stationary Process) A stochastic signal $\mathbf{x} \in \mathbb{R}^N$ is strict-sense stationary over the hypergraph with $\mathbf{P}_\tau$ if and only if*

$$\mathbf{x} \overset{d}{=} \mathbf{P}_\tau \mathbf{x} \tag{9}$$

holds for any τ.

Since the strict-sense stationary is hard to achieve and analyze in the real datasets, we introduce the weak-sense stationary process similar to traditional digital signal processing.

Definition 2. *(Weak-Sense Stationary Process) A stochastic signal $\mathbf{x} \in \mathbb{R}^N$ is weak-sense stationary over the hypergraph with $\mathbf{P}_\tau$ if and only if*

$$\mathbb{E}[\mathbf{x}] = \mathbb{E}[\mathbf{P}_\tau \mathbf{x}] \tag{10}$$

and

$$\mathbb{E}[(\mathbf{P}_{\tau_1}\mathbf{x})((\mathbf{P}^H)_{\tau_2}\mathbf{x})^H] = \mathbb{E}[(\mathbf{P}_{\tau_1+\tau}\mathbf{x})((\mathbf{P}^H)_{\tau_2-\tau}\mathbf{x})^H] \tag{11}$$

hold for any τ, where $\mathbb{E}(\cdot)$ refers to the mean of observations and $(\cdot)^H$ is the Hermitian transpose.

From the definition of the weak-sense stationary process (WSS), Eq. (10) implies that the mean function of the signal must be constant, which is the same condition as in traditional digital signal processing (DSP) [24]. From the definition of supporting matrix, the (i,j)-th entry of $\mathbf{P}$ is the same as the (j,i)-th entry of $\mathbf{P}^H$, which indicates that $\mathbf{P}^H$ is the shifting in the opposite direction of $\mathbf{P}$. Then, the condition in Eq. (11) indicates that the hypergraph covariance function $K_{\mathbf{xx}}(\tau_1, -\tau_2) = K_{\mathbf{xx}}(\tau_1+\tau, \tau-\tau_2) = K_{\mathbf{xx}}(\tau_1+\tau_2, 0)$, which is also consistent with the definition in traditional DSP.

With the definition of the hypergraph stationary process, we have the following properties regarding the relationship between signals and hypergraph spectrum.

Theorem 1. *A stochastic signal $\mathbf{x}$ is WSS if and only if it has zero-mean and its covariance matrix has the same eigenvectors as the hypergraph spectrum basis, i.e.,*

$$\mathbb{E}[\mathbf{x}] = \mathbf{0} \tag{12}$$

and

$$\mathbb{E}[\mathbf{x}\mathbf{x}^H] = \mathbf{V}\Sigma_\mathbf{x}\mathbf{V}^H, \tag{13}$$

where $\mathbf{V} = [\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_N] \in \mathbb{R}^{N\times N}$ are the hypergraph spectrum.

Proof. Since the hypergraph spectrum basis are orthonormal, we have $\mathbf{V}\mathbf{V}^T = \mathbf{I}$. Then, the τ-step shifting based on supporting matrix can be calculated as

$$\mathbf{P}_\tau = \underbrace{\mathbf{V}\Lambda_\mathbf{P}\mathbf{V}^T\mathbf{V}\Lambda_\mathbf{P}\mathbf{V}^T\cdots\mathbf{V}\Lambda_\mathbf{P}\mathbf{V}^T}_{\tau \quad times} \tag{14}$$

$$= \mathbf{V}\Lambda_\mathbf{P}^\tau\mathbf{V}^T. \tag{15}$$

Now, the Eq. (10) can be written as

$$\mathbb{E}[\mathbf{x}] = \mathbf{V}\Lambda_P^\tau\mathbf{V}^T\mathbb{E}[\mathbf{x}]. \tag{16}$$

Since $\mathbf{V}\Lambda_P^\tau\mathbf{V}^T$ does not always equal to $\mathbf{I}$, Eq. (10) holds for arbitrary supporting matrix and τ if and only if $\mathbb{E}[\mathbf{x}] = \mathbf{0}$.

Next we show the sufficiency and necessity of the condition in Eq. (13). The condition in Eq. (11) can be written as

$$\mathbf{P}_{\tau_1}\mathbb{E}[\mathbf{x}\mathbf{x}^H]((\mathbf{P})^H_{\tau_2})^H = \mathbf{P}_{\tau_1+\tau}\mathbb{E}[\mathbf{x}\mathbf{x}^H]((\mathbf{P})^H_{\tau_2-\tau})^H. \tag{17}$$

Considering Eq. (15) and the fact that hypergraph spectrum is real [10], Eq. (11) is equivalent to

$$\mathbf{V}\Lambda_\mathbf{P}^{\tau_1}\mathbf{V}^H\mathbb{E}[\mathbf{x}\mathbf{x}^H]\mathbf{V}\Lambda_\mathbf{P}^{\tau_2}\mathbf{V}^H = \mathbf{V}\Lambda_\mathbf{P}^{\tau_1+\tau}\mathbf{V}^H\mathbb{E}[\mathbf{x}\mathbf{x}^H]\mathbf{V}\Lambda_\mathbf{P}^{\tau_2-\tau}\mathbf{V}^H, \tag{18}$$

which can be written as

$$(\mathbf{V}^H\mathbb{E}[\mathbf{x}\mathbf{x}^H]\mathbf{V})\Lambda_\mathbf{P}^\tau = \Lambda_\mathbf{P}^\tau(\mathbf{V}^H\mathbb{E}[\mathbf{x}\mathbf{x}^H]\mathbf{V}). \tag{19}$$

If Eq. (19) holds for arbitrary $\mathbf{P}$, $(\mathbf{V}^H\mathbb{E}[\mathbf{x}\mathbf{x}^H]\mathbf{V})$ should be diagonal, which indicates $\mathbb{E}[\mathbf{x}\mathbf{x}^H] = \mathbf{V}\Sigma_\mathbf{x}\mathbf{V}^H$. Thus, the sufficiency of the condition is proved.

Similarly, we can apply Eq. (13) on both sides of Eq. (11), we can establish the necessity of the condition in Eq. (13). □

This theorem can be used to estimate the hypergraph spectrum, given multiple observations of several signal points.

2) Estimation of Spectrum Components for Point Clouds: Now, we can use the property of stationary process to estimate the hypergraph spectrum of point clouds. The three coordinates of a point can be interpreted as three observations of the point from different angles, which describe the underlying multilateral relationship. Thus, we can assume that the point cloud signals follow the stationary process over the estimated underlying hypergraph structure. If the point cloud signals $\mathbf{s}$ follow the hypergraph stationarity, it should satisfy Eq. (12) and Eq. (13). Thus, a spectrum estimation method can be based on hypergraph stationarity. The details of the algorithm is described as follows in **Algorithm 1**.

Algorithm 1 Estimation of Hypergraph Spectrum

1: **Input**: Point cloud dataset $\mathbf{s} = [\mathbf{X_1} \quad \mathbf{X_2} \quad \mathbf{X_3}] \in \mathbb{R}^{N\times 3}$.
2: Calculate the mean of each row in $\mathbf{s}$, i.e., $\overline{\mathbf{s}} = (\mathbf{X_1} + \mathbf{X_2} + \mathbf{X_3})/3$;
3: Normalize the original point cloud data as zero-mean in each row, i.e., $\mathbf{s}' = [\mathbf{X_1} - \overline{\mathbf{s}}, \mathbf{X_2} - \overline{\mathbf{s}}, \mathbf{X_3} - \overline{\mathbf{s}}]$;
4: Calculate the eigenvectors $\{\mathbf{f}_1, \cdots, \mathbf{f}_N\}$ for $R_{\mathbf{s}'} = \mathbf{s}'(\mathbf{s}'^T)$ via SVD;
5: **Output**: Hypergraph spectrum $\mathbf{V} = [\mathbf{f}_1, \cdots, \mathbf{f}_N]$.

With *Theorem 1*, we can directly obtain an estimation of the hypergraph spectrum based on the hypergraph stationarity. Note that, here, we assume all the observations are from a clean point cloud without noise. The case of noisy point clouds will be discussed later in Section IV.

Different from the traditional column-wise mean of the coordinates in Eq. (1), we use a row-wise mean of the coordinates to calculate the spectrum. Traditional column-wise mean methods [25] is based on principal component analysis (PCA) in a local region for dimension reduction and visualization of the data structure with a covariance matrix in $\mathbb{R}^{3\times 3}$. Typical applications of such PCA-based methods include points classification and region growing in segmentation [26], [27]. However, our objective is different. Since our goal is to estimate a hypergraph spectrum matrix in $\mathbb{R}^{N\times N}$ based on hypergraph stationary process rather than to reduce the data structure dimensions, the three coordinates represent three observations (or hypergraph signals) of the N nodes. Consequently, we apply the row-wise mean instead, unlike column-wise means used in the PCA-based methods.

B. Estimation of Frequency Coefficients

Next, we discuss how we estimate the hypergraph frequency coefficients with the spectrum components based on the hypergraph smoothness.

In real applications, the large-scale networks are usually sparse, which makes it meaningful to infer that most entries of the hypergraph representing tensor for real datasets are zero [28]. In addition, the smoothness of signals is a widely-used assumption when estimating the underlying structure of graphs and hypergraphs [29]. Thus, the estimation of the hypergraph representing tensor with known spectrum components for a given dataset $\mathbf{s}$ can be generally formulated as

$$\min_{\boldsymbol{\lambda}} \quad \alpha \text{Smooth}(\mathbf{s}, \boldsymbol{\lambda}, \mathbf{f}_r) + \beta ||\mathbf{A}||_T^2 \tag{20}$$

$$s.t. \quad \mathbf{A} = \sum_{r=1}^{N} \lambda_r \cdot \underbrace{\mathbf{f}_r \circ ... \circ \mathbf{f}_r}_{\text{M times}}. \tag{21}$$

$$\mathbf{A} \in \mathcal{A}. \tag{22}$$

$$||\mathbf{A}||_T = \sqrt{\sum_{i_1,i_2,\cdots,i_M=1}^{N} a_{i_1 i_2 \cdots i_M}^2}. \tag{23}$$

The constraint set $\mathcal{A}$ in (22) includes the prior information of the representing tensor. For example, if the representing tensor is the adjacency tensor, its entries should be non-negative. In the constraint of (23), $||\mathbf{A}||_T$ is the tensor norm which controls the sparsity of the hypergraph structure. Here, we consider the Frobenius-like norm [15] since we aim to build a connection between spectra and hypergraph structures. The use of other tensor norms hypergraph applications can be exploited in future works. The smoothness function $\text{Smooth}(\mathbf{s}, \boldsymbol{\lambda}, \mathbf{f}_r)$ can be designed for specific problems. Typical functions can be hypergraph Laplacian regularization, label ranking, and total variation [10].

Here, the HGSP-based total variation is used to measure the smoothness of signals over estimated hypergraph. In Eq. (6), we have

$$\mathbf{A}\mathbf{f}_r^{[M-1]} = \sum_{i=1}^{N} \lambda_i \mathbf{f}_i (\mathbf{f}_i^{\mathrm{T}} \mathbf{f}_r)^{M-1} = \lambda_r \mathbf{f}_r. \tag{24}$$

With the supporting matrix defined in Eq. (7), we also have

$$\mathbf{P}_{\mathbf{s}} \mathbf{f}_r = \frac{1}{\lambda_{max}} \sum_{i=1}^{N} \lambda_i \mathbf{f}_i (\mathbf{f}_i^{\mathrm{T}} \mathbf{f}_r) = \frac{\lambda_r}{\lambda_{max}} \mathbf{f}_r. \tag{25}$$

Consequently, the total variation can be written with the supporting matrix as follows:

$$\mathbf{TV}(\mathbf{s}) = ||\mathbf{s} - \mathbf{P}_{\mathbf{s}}\mathbf{s}||_k, \tag{26}$$

where k ls application dependent.

For convenience, we use the quadratic-form total variation based on the supporting matrix to describe the hypergraph smoothness, i.e.,

$$\mathbf{TV}(\mathbf{s}) = ||\mathbf{s} - (1/\lambda_{max})\mathbf{P}\mathbf{s}||_2^2. \tag{27}$$

This form of smoothness function suggested in [10] can capture the differences between one node and its neighbors over hypergraph. Since the signals are smooth over the estimated hypergraph, observations are also smooth. Thus, the final smoothness function for point cloud $\mathbf{s} = [\mathbf{X_1} \quad \mathbf{X_2} \quad \mathbf{X_3}]$ is

$$\begin{aligned} \text{Smooth}(\mathbf{s}, \boldsymbol{\lambda}, \mathbf{f}_r) &= \sum_{i=1}^{3} ||\mathbf{X}_i - \mathbf{P}_{\mathbf{s}} \mathbf{X}_i||_2^2 \\ &= \sum_{i=1}^{3} ||\mathbf{X}_i - \sum_{r} \sigma_r (\mathbf{f}_r^T \mathbf{X}_i) \mathbf{f}_r||_2^2 \\ &= \sum_{i=1}^{3} ||\mathbf{X}_i - \mathbf{W}_i \boldsymbol{\sigma}||_2^2, \end{aligned} \tag{28}$$

where $\mathbf{W}_i = [(\mathbf{f}_1^T \mathbf{X}_i)\mathbf{f}_1 \quad (\mathbf{f}_2^T \mathbf{X}_i)\mathbf{f}_2 \quad \cdots \quad (\mathbf{f}_N^T \mathbf{X}_i)\mathbf{f}_N]$, $\sigma_r = \lambda_r / \lambda_{max}$ and $\boldsymbol{\sigma} = [\sigma_1 \cdots \sigma_N]^T$.

Moreover, the tensor norm of a given hypergraph has the following property with the frequency coefficients.

Theorem 2. *Given a representing tensor* $\mathbf{A} = \sum_{r=1}^{N} \lambda_r \cdot \underbrace{\mathbf{f}_r \circ ... \circ \mathbf{f}_r}_{M \text{ times}}$, *the tensor norm* $||\mathbf{A}||_T^2 = \sum_{i_1,i_2,\cdots,i_M=1}^{N} a_{i_1 i_2 \cdots i_M}^2$ *can be written in the form of frequency coefficients as*

$$||\mathbf{A}||_T^2 = \sum_{r=1}^{N} \lambda_r^2 = \boldsymbol{\lambda}^T \boldsymbol{\lambda}, \tag{29}$$

where $\boldsymbol{\lambda} = [\lambda_1 \quad \lambda_2 \quad \cdots \quad \lambda_N]^T$.

Proof. Since $\mathbf{A} = \sum_{r=1}^{N} \lambda_r \cdot \underbrace{\mathbf{f}_r \circ ... \circ \mathbf{f}_r}_{\text{M times}}$, we have

$$a_{i_1 i_2 \cdots i_M} = \sum_{r=1}^{N} \lambda_r f_{r,i_1} f_{r,i_2} \cdots f_{r,i_M}, \tag{30}$$

Algorithm 2 Estimation of Frequency Coefficient

1: **Input**: Point cloud dataset $\mathbf{s} = [\mathbf{X_1}, \mathbf{X_2}, \mathbf{X_3}] \in \mathbb{R}^{N\times 3}$, hypergraph spectrum $\mathbf{V} = [\mathbf{f}_1, \cdots, \mathbf{f}_N]$.
2: **for** i=1,2,...,iter **do**:
3: Set $\sigma_i = 1$ as the maximal normalized eigenvalue.
4: Solve the optimization problem in Eq. (32).
5: **end for**
6: Find the optimal i to minimize the target function.
7: The optimal coefficients $\boldsymbol{\sigma}$ is the solution of Eq. (32) correlated to the optimal i.
8: **Output**: Frequency coefficients $\boldsymbol{\sigma}$.

where $f_{r,i}$ is the ith element of $\mathbf{f}_r$. Then, the tensor norm is

$$\begin{aligned}
||\mathbf{A}||_T^2 &= \sum_{i_1,i_2,\cdots,i_M} (\sum_{r=1}^{N} \lambda_r f_{r,i_1} f_{r,i_2} \cdots f_{r,i_M})^2 \\
&= \sum_{i_1,i_2,\cdots,i_M} (\sum_{r=1}^{N} \lambda_r f_{r,i_1} \cdots f_{r,i_M})(\sum_{t=1}^{N} \lambda_t f_{t,i_1} \cdots f_{t,i_M}) \\
&= \sum_{i_1,i_2,\cdots,i_M} \sum_{r,t} \lambda_r \lambda_t f_{r,i_1} \cdots f_{r,i_M} f_{t,i_1} \cdots f_{t,i_M} \\
&= \sum_{r,t} \lambda_r \lambda_t \sum_{i_1,i_2,\cdots,i_M=1}^{N} (f_{r,i_1} f_{t,i_1}) \cdots (f_{r,i_M} f_{t,i_M}) \\
&= \sum_{r,t} \lambda_r \lambda_t (\mathbf{f}_r^T \mathbf{f}_t)^M. \qquad (31)
\end{aligned}$$

Since $\mathbf{f}_r$ is orthonormal, $\mathbf{f}_r^T \mathbf{f}_t = 1$ holds if $r = t$; otherwise, $\mathbf{f}_r^T \mathbf{f}_t = 0$. Thus, we obtain $||\mathbf{A}||_T^2 = \sum_{r=1}^{N} \lambda_r^2$. □

This property can help us build a connection from the tensor norm to the frequency coefficients directly.

Now, if we consider the representing tensor as the adjacency tensor and each hyperedge consists of three nodes since at least three nodes are required to construct a surface, we optimize the normalized frequency coefficients $\boldsymbol{\sigma} = \frac{1}{\lambda_{max}} \boldsymbol{\lambda} = [\sigma_1 \quad \sigma_2 \quad \cdots \quad \sigma_N]^T$ via

$$\min_{\boldsymbol{\sigma}} \quad \alpha \sum_{i=1}^{3} ||\mathbf{X}_i - \mathbf{W}_i \boldsymbol{\sigma}||_2^2 + \beta \boldsymbol{\sigma}^T \boldsymbol{\sigma} \qquad (32)$$

$$s.\,t. \quad 0 \le \sigma_r \le \max_i \sigma_i = 1, \qquad (33)$$

$$\sum_{r=1}^{N} \sigma_r f_{r,i_1} f_{r,i_2} f_{r,i_3} \ge 0, \quad i_1, i_2, i_3 = 1, 2, \cdots, N. \qquad (34)$$

The constraint (34) limits the estimated representing tensor as the adjacency tensor. The constraint (33) is the nonnegative constraint on weight and the factor [16]. Clearly, the optimization is non-convex with the constraint $\max_i \sigma_i = 1$. However, if the position of the maximal frequency is known, the optimization problem can be solved by tools such as cvx [30], [31]. Thus, we can develop the following algorithm to estimate the frequency coefficients.

Note that, since we consider clean point cloud without noise, we usually set parameter $\alpha \le \beta$. Then, from the estimated spectrum pair $(\mathbf{f}_r, \sigma_r)$ under normalization, we can recover the original adjacency tensor as Eq. (21). Hence, the hypergraph construction process for a clean point cloud can be summarized as Fig. 2. The recovery of original adjacency tensor is not always necessary in practical applications since storing the representing tensor is less efficient than storing the spectrum pairs.

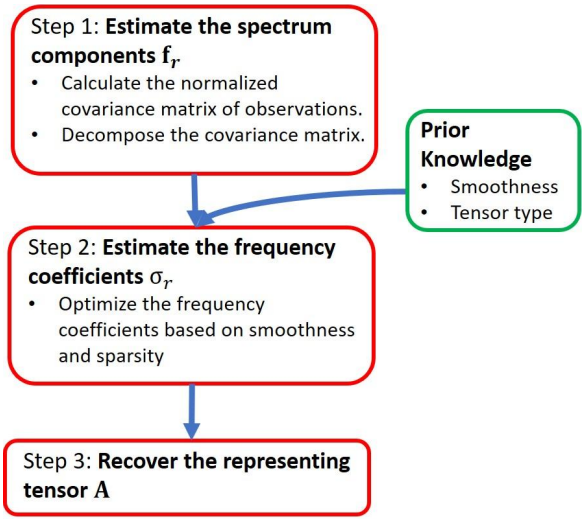


Fig. 2. Estimation of Hypergraph Spectral Pairs for Original Point Clouds

IV. Joint Spectrum Estimation and Denoising

In practical 3D imaging, perturbations such as noises and outliers often exist when generating a point cloud of an unknown object. These noises may significantly affect the performance of point cloud processing since many existing algorithms require quality datasets [30]. Thus, denoising remains a vital issue in practical point cloud applications.

Usually, to denoise point sets with sharp features is difficult, especially when the noise is large, as such features are hard to distinguish from noise effect. Generally, smoothness-based methods are common. In [32], a method based on L_0 norm of differences between k-nearest neighbors is introduced. In [33], Laplacian regularization is used to describe smoothness and to denoise noisy point sets. Other works, such as [8], [34], minimize the total variation over graphs to denoise the point sets. Although smoothness-based methods have achieved notable successes, how to interpret and define an effective smoothness function for a general point set remains open. Furthermore, for graph-based smoothness methods, the construction of graph model remains a critical problem, since traditional methods based on distance suffers from the imprecise location measurement. To this end, a more general definition of smoothness and a more efficient denoising method for arbitrary point clouds are highly desirable.

In this section, we introduce a joint method to simultaneously estimate the hypergraph structure and denoise noisy point clouds. In Section III, we already introduce an estimation method of spectral pair $(\mathbf{f}_r, \sigma_r)$ for clean point clouds. A similar construction process can be developed for the noisy point clouds. As the estimation of spectrum components only depends on the observed data, we need to denoise the noisy observations while optimizing the frequency coefficients. As

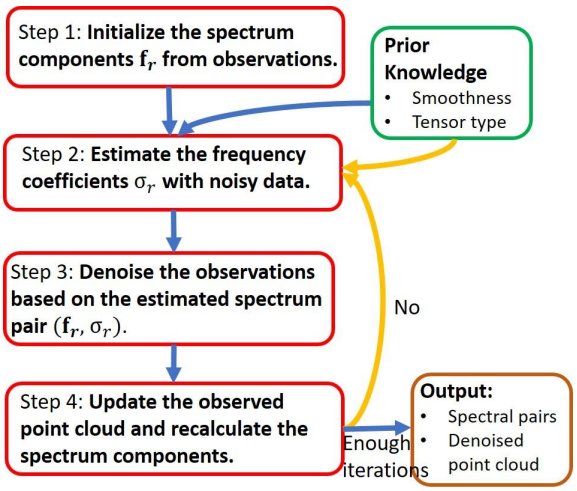


Fig. 3. Joint Hypergraph Estimation and Denoising for Noisy Point Cloud.

already discussed, the problem of denoising a signal on a hypergraph can be written as a convex minimization problem with the constraints that denoised signals should be smooth over the hypergraph. Accordingly, the general process of hypergraph denoising and estimation can be summarized as the following steps:

- Step 1: Estimate the approximated hypergraph spectrum components from the observed noisy point clouds;
- Step 2: Jointly estimate frequency coefficients and denoise the noisy observations;
- Step 3: Update the noisy observations as denoised data and repeat Step 1 until enough iterations.

To estimate hypergraph spectral components of noisy data, the process is the same as **Algorithm 1** based on hypergraph stationary process. To jointly estimate the frequency coefficients to recover the original underlying structure and to denoise the noisy point clouds, we propose the following objective. Given N noisy points $\mathbf{s} = [\mathbf{X_1} \quad \mathbf{X_2} \quad \mathbf{X_3}]$, the joint estimation task can be formulated as

$$\min_{\boldsymbol{\sigma},\mathbf{Y}} \quad \sum_{i=1}^{3}[||\mathbf{X}_i - \mathbf{Y}_i||_2^2 + \alpha||\mathbf{X}_i - \mathbf{W}_i\boldsymbol{\sigma}||_2^2] + \beta||\mathbf{A}||_T^2 \tag{35}$$

$$\begin{aligned} s.t. \quad & \mathbf{A} = \sum_{r=1}^{N} \lambda_r \cdot \underbrace{\mathbf{f}_r \circ \ldots \circ \mathbf{f}_r}_{\text{M times}} \in \mathcal{A}, \\ & 0 \le \sigma_r \le \max_i \sigma_i = 1, \\ & \mathbf{W}_i = [(\mathbf{f}_1^T\mathbf{X}_i)\mathbf{f}_1 \quad (\mathbf{f}_2^T\mathbf{X}_i)\mathbf{f}_2 \quad \cdots \quad (\mathbf{f}_N^T\mathbf{X}_i)\mathbf{f}_N]. \end{aligned}$$

The resulting $\mathbf{Y} = [\mathbf{Y_1} \quad \mathbf{Y_2} \quad \mathbf{Y_3}]$ is the denoised point clouds, and (α, β) are two positive regularization parameters. The first part in Eq. (35) lets the denoised point cloud maintain the observed structural features. The second part is the smoothness function derived from Eq. (28) which adjusts positions of noisy points. The third part is the tensor norm regularization to control hypergraph sparsity.

Algorithm 3 Joint Hypergraph Estimation and Point Cloud Denoising

1: **Input**: Noisy observations of point clouds $\mathbf{s} = [\mathbf{X_1}, \mathbf{X_2}, \mathbf{X_3}] \in \mathbb{R}^{N\times 3}$.
2: **Initialization**: Calculate the spectrum components $\mathbf{f}_r$'s from the observed point cloud $\mathbf{s}$ as **Algorithm 1**.
3: **for** i=1,2,...,iter **do**:
4: Find the optimal $\boldsymbol{\sigma}$ for the first subproblem in Eq. (36) with **Algorithm 2**.
5: Solve the optimization problem in Eq. (37) with $\mathbf{Y}$ in Eq. (38).
6: Update the observed signals as $\mathbf{Y}$ and recalculate the spectrum components $\mathbf{f}_r$'s.
7: **end for**
8: **Output**: Spectral pairs $(\mathbf{f}_r, \sigma_r)$'s, denoised point clouds $\mathbf{Y}$.

The optimization problem of Eq. (35) is not convex in $\mathbf{Y}$ and $\boldsymbol{\sigma}$. Therefore, similar to [29], we split the problem into two subproblems. For each subproblem, we fix one variable set to solve the other one. Upon convergence, the solution corresponds to a local minimum and not necessarily a global minimum.

We first initialize $\mathbf{Y}$ as the observed signals $\mathbf{X}$ and solve the following problem similar to that in Section III.

$$\min_{\boldsymbol{\sigma}} \alpha \sum_{i=1}^{3} ||\mathbf{X}_i - \mathbf{W}_i\boldsymbol{\sigma}||_2^2 + \beta\boldsymbol{\sigma}^T\boldsymbol{\sigma} \tag{36}$$

$$\begin{aligned} s.t. \quad & 0 \le \sigma_r \le \max_i \sigma_i = 1, \\ & \sum_{r=1}^{N} \sigma_r f_{r,i_1} f_{r,i_2} f_{r,i_3} \ge 0, \quad i_1, i_2, i_3 = 1, 2, \cdots, N. \end{aligned}$$

This problem can be solved similarly to the solution of clean point cloud with **Algorithm 2**.

Once the estimated frequency coefficients are found, we solve the subproblem of point cloud denoising

$$\min_{\mathbf{Y}} \sum_{i=1}^{3}[||\mathbf{X}_i - \mathbf{Y}_i||_2^2 + \alpha||\mathbf{X}_i - \mathbf{W}_i\boldsymbol{\sigma}||_2^2], \tag{37}$$

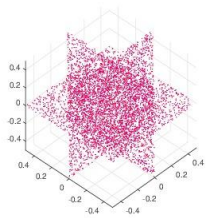
whose close-form solution for each coordinate is

$$\mathbf{Y}_i = [\mathbf{I} + \alpha(\mathbf{I} - \mathbf{P_s})^T(\mathbf{I} - \mathbf{P_s})]^{-1}\mathbf{X}_i. \tag{38}$$

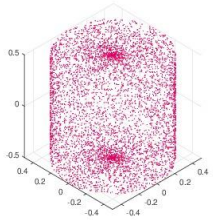
Note that $\mathbf{P_s}$ is the supporting matrix. We then update the frequency components based on the denoised point clouds, and repeatedly carry out Step 1 to Step 3 until getting the final solution. In practice, we generally observe the convergence within only a few iterations. The complete algorithm is summarized in **Algorithm 3** as shown in Fig. 3. Unlike for clear point clouds, we emphasize more on the smoothness of signals over the hypergraph. The parameter α can be set larger than used when dealing with clean point clouds.

V. Application Examples

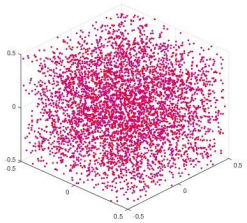
In this section, we examine two application examples to test the efficacy of the proposed method in estimating hypergraph structure for both clear and noisy point clouds.



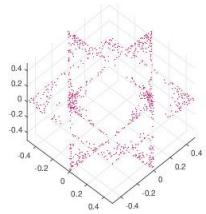
(a) Original Surfaces with 6000 Points.



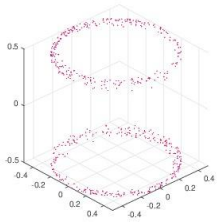
(b) Original Cylinder with 6000 Points.



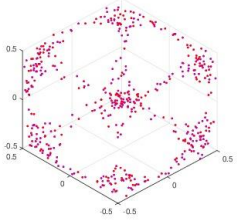
(c) Original Cube with 5000 Points.



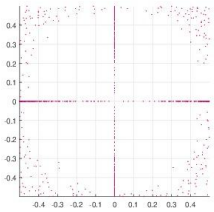
(d) Sampled Surfaces with 800 Points.



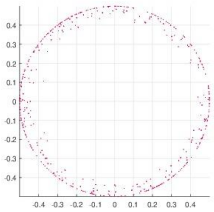
(e) Sampled Cylinder with 600 Points.



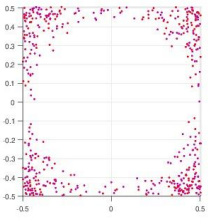
(f) Sampled Cube with 500 Points.



(g) Sampled Surfaces with 800 Points.



(h) Sampled Cylinder with 600 Points.



(i) Sampled Cube with 500 Points.

Fig. 4. View from the Top of Sampled Point Clouds.

A. *Sampling*

Sampling is an important operation to facilitate analysis of very large point clouds. In this part, we consider different sampling strategies depending on different kinds of applications. Some interesting connections are found from the hypergraph frequency and point cloud features.

1) Resampling using Harr-like Highpass Filtering: Filtering helps extract select features of a given dataset. In some applications such as boundary detection, accurate extraction of shape features of point clouds is important. Thus, an efficient sampling should retain the features of the original point cloud. In our estimation of hypergraph structure, smoothness is a significant feature to model point clouds. Ideally, smoothness over the original surface of a point cloud should correspond to smoothness over its hypergraph model. Therefore, we can also design a Harr-like high-pass filter to extract sharp features over the surfaces.

Let $\mathbf{I}$ be an identity matrix of appropriate size. Similar to that in GSP [3], a Haar-like high-pass filter is designed as

$$\mathbf{H} = \mathbf{I} - \mathbf{P_s} \tag{39}$$

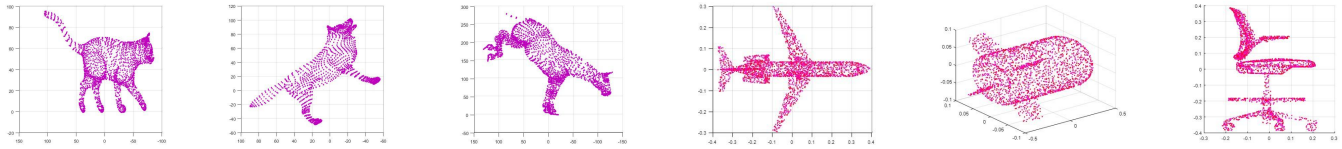
$$= \mathbf{V} \begin{bmatrix} 1-\sigma_1 & 0 & \cdots & 0 \\ 0 & 1-\sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1-\sigma_N \end{bmatrix} \mathbf{V}^T. \tag{40}$$

The filtered signal is

$$(\mathbf{Hs})_i = \mathbf{s}_i - \sum_j P_{s(ij)} \mathbf{s}_j, \tag{41}$$

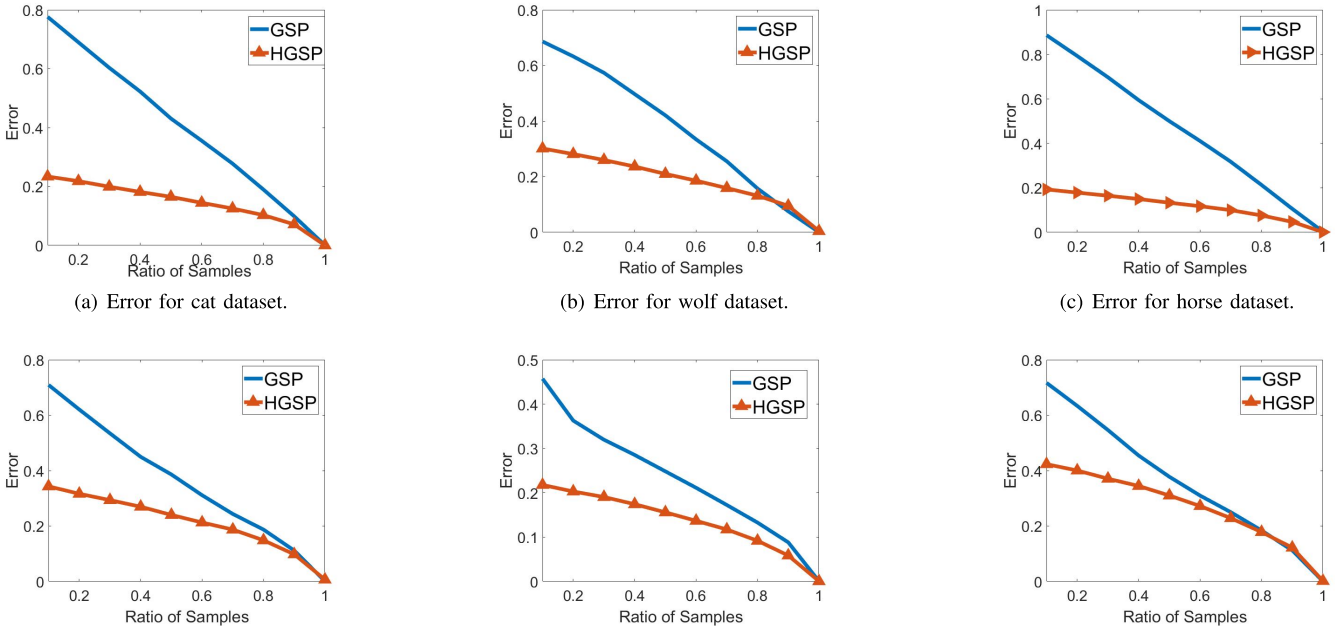
which reflects the differences between nodes and their neighbors over the hypergraph. Note that, the frequency coefficients together with their corresponding spectral components are ordered decreasingly here, i.e., $\sigma_i \geq \sigma_{i+1}$. From the definition of total variation, more smoothness corresponds to larger total variation. Thus, we can extract the sharp features over the point clouds by sampling the nodes with large value of $||\mathbf{s}_i - \sum_j P_{s(ij)} \mathbf{s}_j||_2^2$.

To test this application, we estimate the spectral pairs for clean point clouds and filter the signals over several synthetic datasets. We randomly generate multiple points over the surfaces of basic graphics shown as Fig. 4(a) - 4(c), and sample the point clouds using the high-pass filter (HPF) given in Fig. 4(d) - 4(f). From the test results, we can see that the sampled points of the surfaces in Fig. 4(d) mainly congregate near the corners and edges, which are the sharp parts of the point clouds. In addition, the sampled nodes for a cube shape are also crowded near edges and corners. On the other hand, the sampled nodes of a cylinder are mostly at the boundaries of the cylinder. Our test results show that the Harr-like HPF can extract sharp features from point cloud surfaces, which correspond to the least smooth parts of



(a) Cat. (b) Wolf. (c) Horse. (d) Plane. (e) Rocket. (f) Chair.

Fig. 5. Test Datasets of Sampling.



(a) Error for cat dataset. (b) Error for wolf dataset. (c) Error for horse dataset.

(d) Error for plane dataset. (e) Error for rocket dataset. (f) Error for chair dataset.

Fig. 6. Error between Recovered Data and Original Data.

the estimated hypergraph. Moreover, since the total variation measures the order of frequency, sharp features over the point cloud correspond to high frequency components. Thus, the hypergraph model and the estimated spectral pairs are efficient when extracting features of 3D point clouds.

2) Down-Sampling with Hypergraph Fourier Transform: Projecting signals into a suitable orthonormal basis is a widely-used sampling method [35]. The work of [10] develops a sampling theory based on hypergraph signal processing as follows:

- Step 1: Order the spectrum components from low frequency to high frequency based on their total variations.
- Step 2: Implement hypergraph Fourier tranform as

$$\mathcal{F}(\mathbf{s}) = [(\mathbf{f}_1^T\mathbf{s})^{M-1} \quad (\mathbf{f}_2^T\mathbf{s})^{M-1} \quad \cdots \quad (\mathbf{f}_N^T\mathbf{s})^{M-1}]^T. \tag{42}$$

- Step 3: Use C transformed signal components in the hypergraph frequency domain to represent N signals in the original vertex domain.

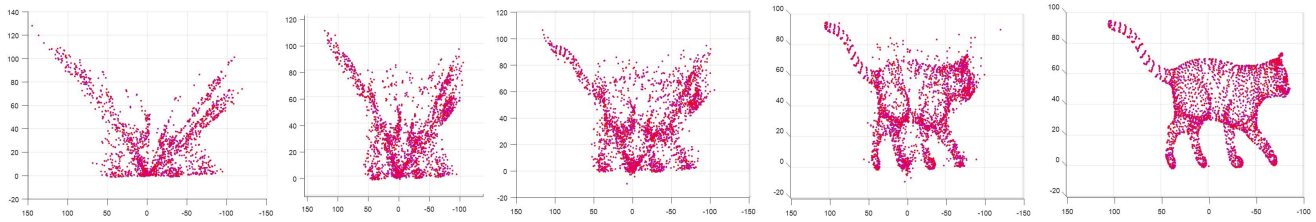
More specifically, for a K-bandlimitted hypergraph signal, a perfect recovery is available with K samples in hypergraph frequency domain. Similarly, we can sample the point clouds based on the hypergraph Fourier transform. To test the performance of the sampled signals, we implement hypergraph Fourier transform (HGFT) on each coordinates of the point clouds, i.e., $\mathcal{F}(\mathbf{X}_i)$ for all i. Then, we take the first C transformed signals in all coordinates. Finally, we implement the inverse hypergraph Fourier transform (iHGFT) to obtain the sampled shapes of the original point clouds. Note that, perfect recovery happens with C samples, if $(\mathcal{F}(\mathbf{X}_i))_{j+C} = 0$ for $i, j \in \mathcal{Z}^+$.

We test the recovered point clouds for animal point datasets [36]–[39] and the ShapeNet datasets of objects [44], [45] with the GSP-based methods. For the GSP-based method, we construct a graph adjacency matrix $\mathbf{W}$ with Gaussian model, i.e.,

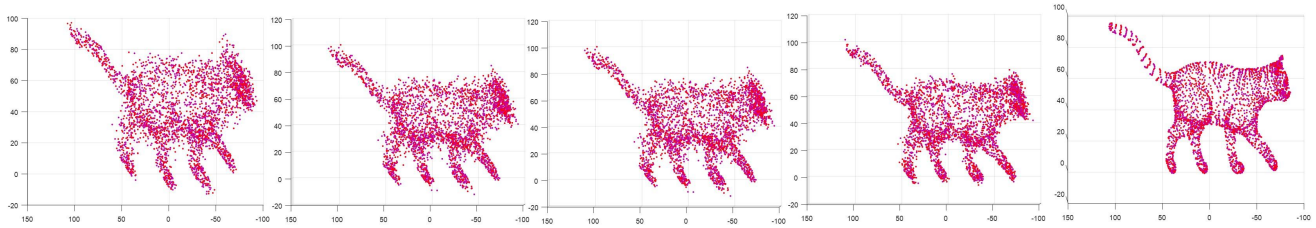
$$W_{ij} = \begin{cases} \exp\left(-\dfrac{||\mathbf{s}_i - \mathbf{s}_j||_2^2}{\delta^2}\right), & ||\mathbf{s}_i - \mathbf{s}_j||_2^2 \le t; \\ 0, & \text{otherwise}, \end{cases} \tag{43}$$

where $\mathbf{s}_i$ is the coordinates of the ith node. Then, we sample the point clouds using the signals after the graph Fourier transform (GFT). The test point cloud is shown as Fig. 5. We first compare the error defined as

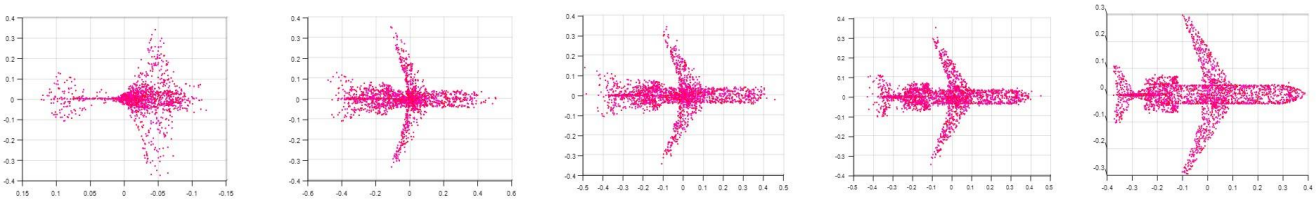
$$Error_1 = \frac{\sum_i |\mathbf{X}_i - \mathbf{X}_i^{rec}|}{\sum_i |\mathbf{X}_i|} \tag{44}$$



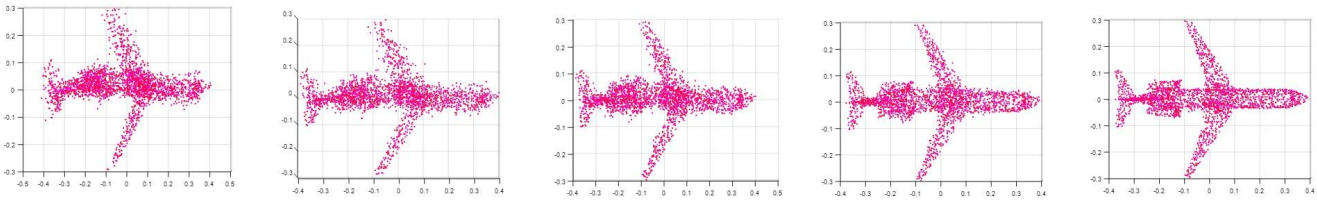
(a) Recovery from GSP-based Sampling with 30%, 50%, 70%, 90%, 98% Ratio of Samples for Cat Datasets.



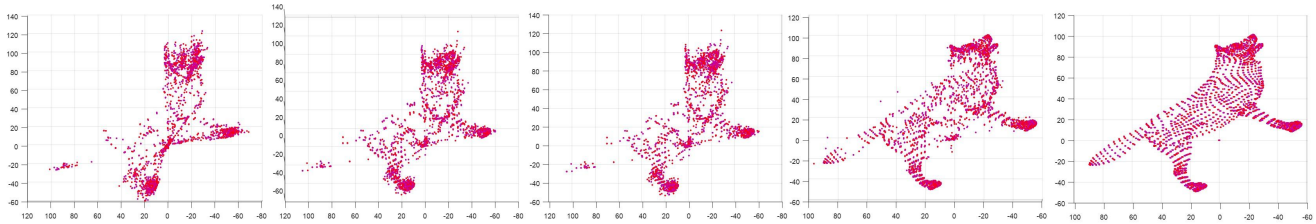
(b) Recovery from HGSP-based Sampling with 30%, 50%, 70%, 90%, 98% Ratio of Samples for Cat Datasets.



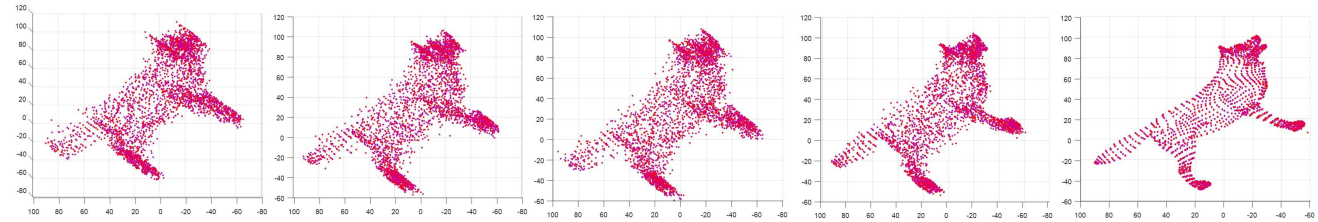
(c) Recovery from GSP-based Sampling with 30%, 50%, 70%, 90%, 98% Ratio of Samples for Plane Datasets.



(d) Recovery from HGSP-based Sampling with 30%, 50%, 70%, 90%, 98% Ratio of Samples for Plane Datasets.

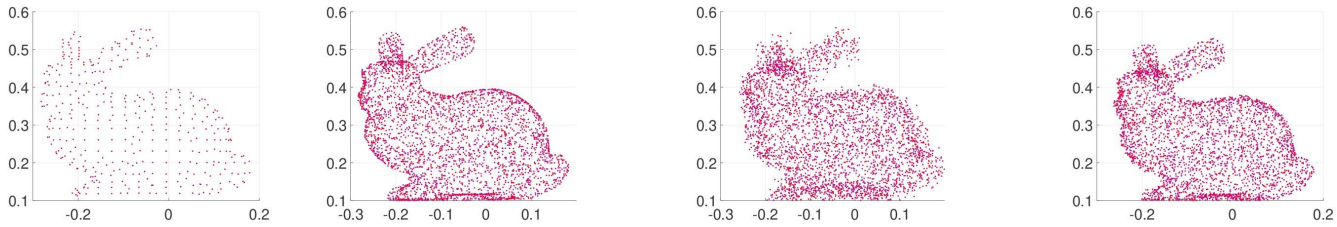


(e) Recovery from GSP-based Sampling with 30%, 50%, 70%, 90%, 98% Ratio of Samples for Wolf Datasets.



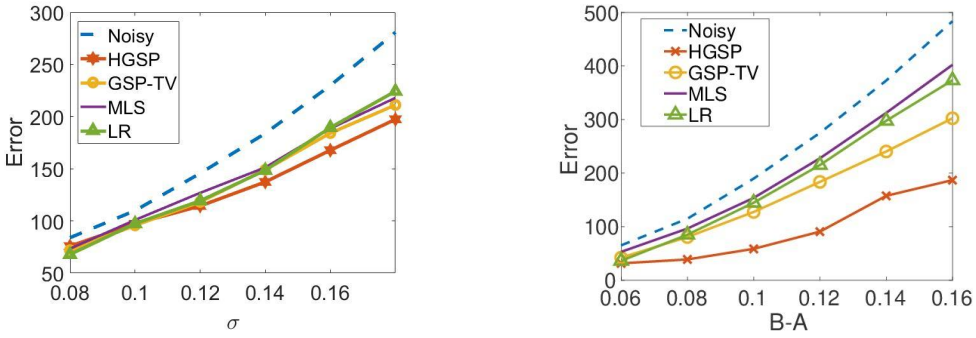
(f) Recovery from HGSP-based Sampling with 30%, 50%, 70%, 90%, 98% Ratio of Samples for Wolf Datasets.

Fig. 7. Recovered Point Clouds from Sampled Transformed Signals.



(a) Bunny with 397 Samples. (b) Bunny with 3595 Samples. (c) Noisy Bunny Dataset. (d) Denoised Bunny Dataset.

Fig. 8. Visualization of the Original and Denoised Bunny Point Clouds.



(a) Comparison in Gaussian Distribution. (b) Comparison in Uniform Distribution.

Fig. 9. Comparison between Different Methods.

between the recovered point clouds and original point clouds with sampling ratio $0.1 \sim 1$ shown as Fig. 6. From the experimental results, we can see that the HGSP-based method has smaller error than the GSP downsampling method, clearly indicating hypergraph to be a better model. However, sometimes, the recovery error alone cannot tell the true story in terms of the performance for the recovered point clouds. To explore more, we compare the recovered point clouds directly in Fig. 7. From the experimental results, we can see that HGSP-based method captures the overall structure of the point clouds with very few samples, whereas the GSP-based method requires more samples to get sufficient details. The error of GSP mainly stems from some outliers when taking more than 90 percent of the samples. The experiments show that HGSP-based method is a better tool for applications which need to recover an overall shape of point clouds from limited data storage. Our test shows hypergraph to be a suitable model for point clouds, and the estimated hypergraph spectral pairs capture the point cloud characteristics very well.

B. Denoising

From estimated hypergraph spectral pairs from noisy point clouds, the performance of denoising is an intuitive metric of how good the estimates are. There are multiple methods developed to denoise noisy point clouds. The authors of [8] proposed a graph-based method to denoise based on total variation (GSP-TV). This method constructs a graph based on observed coordinates first before solving the denoising optimization

$$\min_{\mathbf{Y}} ||\mathbf{X}-\mathbf{Y}||_2^2 + \alpha \mathbf{TV}(\mathbf{Y}, \mathbf{W}), \tag{45}$$

where $\mathbf{X}$ is the observed coordinates, and $\mathbf{W}$ is the adjacency matrix. Here, the graph total variation $\mathbf{TV}(\mathbf{Y}, \mathbf{W})$ is applied in describing the smoothness over the graphs. In addition to total variation, Laplacian regularization (LR) [40] has also been used in denoising with a basic formulation

$$\min_{\mathbf{Y}} ||\mathbf{X}-\mathbf{Y}||_2^2 + \alpha ||\mathbf{Y}^T \mathbf{L} \mathbf{Y}||_2^2, \tag{46}$$

where $\mathbf{L}$ is the Laplacian matrix. Developed from traditional Laplacian regularization methods, a mesh Laplacian smooth (MLS) method is given in [41]. Other graph learning based methods include graph Laplacian regularization (GLR) [42] in a low-dimension manifold and feature graph learning (GL) [43].

1) Overall Performance of Denoising: To validate the performance of our denoising method, we first compare with the aforementioned traditional methods using the Standford bunny dataset with 397 points and sampled bunny with 3595 points, shown as Fig. 8(a) and Fig. 8(b), respectively. For GSP-based methods, a graph is constructed according to the Gaussian distance model to encode the local geometry information through an adjacency matrix in Eq. (43) [3]. For the Laplacian-based method, we establish the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{S}$, where $\mathbf{S}$ is the unweighted adjacency matrix and $\mathbf{D}$ is the diagonal matrix of node degree. We compare different methods in the sampled bunny dataset adding zero-mean Gaussian noise with variance σ^2, and zero-mean Uniform noise with the interval $B - A$, respectively. We use the error denoted by

$$Error_2 = \sum_{i=1}^{N} \sum_{j=1}^{3} |X_{ji} - Y_{ji}|, \tag{47}$$

TABLE I
COMPARISON IN DIFFERENT DATASETS OF DIFFERENT NOISE.

	Noisy	GSP-TV [8]	LR [39]	MLS [40]	GLR [41]	GL [42]	**HGSP**
Uniform∼U(-3, 3)							
Cat	2.9819	2.3744	2.4649	2.5312	2.2576	2.2412	**2.2209**
Horse	3.0377	2.7638	2.8280	2.8615	2.6777	2.6415	**2.6216**
Wolf	2.9857	2.6009	2.5650	2.6312	2.4391	2.4281	**2.4164**
Rocket	3.0587	2.6179	2.5391	2.6346	2.4512	2.3645	**2.3052**
Plane	2.9638	2.5369	2.4118	2.5110	2.4374	2.3827	**2.3690**
Chair	3.0233	2.5532	2.3718	2.5988	2.2855	**2.1809**	2.2048
Gaussian∼N(0, 2)							
Cat	4.0218	3.4952	3.2597	3.6202	**2.9778**	3.0112	3.0026
Horse	4.1020	3.6088	3.8241	3.7268	3.4370	3.3805	**3.2996**
Wolf	4.0402	3.4154	3.2637	3.4756	3.1616	3.0800	**3.0463**
Rocket	4.1527	3.2323	3.4771	3.5967	3.1415	**3.1063**	3.2471
Plane	3.9544	3.3188	2.9365	3.4101	2.9677	2.9246	**2.8868**
Chair	4.0119	3.4372	3.0770	3.5044	3.1987	3.0212	**3.0208**
Impulse (p=0.08)							
Cat	8.4066	**7.3728**	7.6285	7.8024	7.4001	7.3802	7.3844
Horse	35.7202	30.0853	29.8711	33.2011	28.9012	28.1285	**27.9143**
Wolf	9.4107	8.5086	8.5757	8.6105	8.2024	8.1684	**8.1395**
Rocket	1.8312	1.6718	1.6312	1.7354	1.6010	**1.5896**	1.6619
Plane	0.7759	0.6413	0.6559	0.6678	0.6752	0.6588	**0.6386**
Chair	1.1175	0.9374	0.9081	0.9661	0.8815	0.8622	**0.8598**
Average							
	5.5335	4.7318	4.6827	5.0047	4.5058	4.4195	**4.3657**

where X_{ij} and Y_{ji} are the jth coordinates of observed and denoised point i, respectively, to measure the performance. We repeat the test on 1000 randomly generated noisy data. The error between the original dataset and the denoised dataset is shown in Fig. 9. The error of the noisy point clouds before denoising is also given as a reference in Fig. 9. From the test results, we can see that the HGSP-based method can achieve the lowest error, which demonstrates the effectiveness of the proposed denoising methods and estimated spectral pairs. In addition, the denoised bunny with 3595 samples is shown in Fig. 8(d), using our proposed method to denoise the noisy bunny in Fig. 8(c). The successful recovery of the bunny point cloud presents strong evidence that our estimated spectral pairs and denoising method are powerful tools in processing noisy point clouds.

To test the performance in a more general experiment setup, we compare the proposed method with traditional methods together with graph learning based methods with other types of noise in both complex animal datasets and the ShapeNet object datasets in Fig. 5. We normalize the coordinates of each point cloud to maintain a similar noise level. Using mean squared error (MSE) to measure the performances of different methods, the results are shown in Table. I. We can see that the HGSP-based denoising achieves the best overall performances in most datasets with different noise types. Generally, GLR is better than traditional Laplacian regularization owing to its utilization of self-similarity among surface patches, and GL exhibits a slight improvement over GLR. The GSP-based total variation is worse than the HGSP-based total variation since the graph is constructed from noisy coordinates. Moreover, these methods perform better under Gaussian and uniform noises, since impulsive noise has a rescaling effect on the coordinates according to the coefficient p.

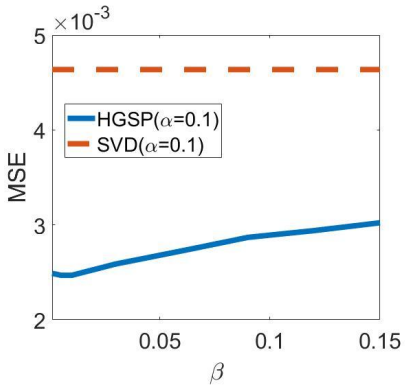
Discussion: Generally speaking, the total variation and the Laplacian regularization are both effective descriptions for signal smoothness. Hypergraphs and graphs can be efficient tools in processing point clouds for specific datasets. For example, hypergraphs are more efficient in processing complex shape features and tend to capture the overall shapes well with fewer samples. It would be interesting to explore a joint framework including both hypergraphs and graphs in processing different kinds of features in point clouds. Additionally, future works should also evaluate the effects of different tensor norms.

2) Impact of Sparsity and Smoothness: In addition to the overall performance, it is interesting to consider the ablation effect on smoothness and sparsity. To set up such additional experiments, we compare the impact of optimization of frequency coefficients with the simple approach of recovering eigenvalues from SVD of the covariance matrix.

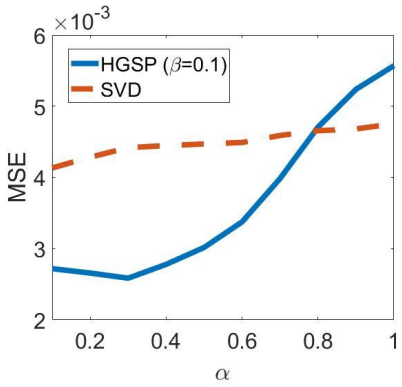
We apply the same strategy to denoise the bunny datasets with uniform noise U(-0.1, 0.1) for the SVD-based method and the HGSP optimization based method in Eq. (35). For the SVD-based method, eigenvalues are directly from step 4 in **Algorithm 1** and only α is used in denoising (smoothness) as Eq. (38). For the HGSP-based method, we first fix the parameter $\alpha = 0.1$ (smoothness) and measure the impact of β (sparsity) in Fig. 10(a). Next, we freeze the parameter $\beta = 0.1$ and test on different α's to measure the impact of smoothness in Fig. 10(b).

From Fig. (10), we observe that the optimal HGSP-based results achieve lower MSE than the SVD-based method in general, except for very large α. With properly chosen parameters for optimization, these results show that the HGSP-based method is better than the SVD-based method, and demonstrate the effectiveness of the proposed optimization-based method.

In Fig. 10(a), we can see that the appropriate β lies in $[0.01, 0.03]$, which is smaller than $\alpha = 0.1$. In Fig. 10(b), the optimum α lies near 0.3, larger than $\beta = 0.1$. Generally, α larger than β generates reasonably desirable results. Moreover, we can see from the curves of MSE that denoising is more



(a) Impact of Sparsity.



(b) Impact of Smoothness.

Fig. 10. Impact of Sparsity and Smoothness.

sensitive to smoothness than sparsity.

VI. Conclusions and Future Directions

In this work, we develop HGSP tools for effectively processing 3D point clouds. We first introduce a novel method to estimate hypergraph spectral components and present an optimization formulation to optimally select frequency coefficients to recover the optimal hypergraph structure. We develop a HGSP algorithm to jointly estimate hypergraph spectrum pairs and denoise noisy point clouds. To test the practicality and efficacy of our proposed hypergraph tools, we study two point cloud application examples. Our results illustrate significant performance improvements for both sampling and denoising applications. Moreover, we establish a clear connection between hypergraph frequency components and features on point-cloud surface that can be exploited in future studies.

Our work establish hypergraph signal processing as an efficient tool in tackling high-dimensional interactions among multiple nodes. In addition to sampling and denoising, HGSP can find good applications in many other aspects of point clouds through estimation of spectral components and frequency coefficients. One direction is the design of filters to analyze the spectral properties and surface features of 3D point clouds. Another interesting problem is the recovery of point clouds from low dimensional samples. Beyond point clouds, HGSP can also effectively handle datasets with other complex underlying structure.

References

[1] A. Aldoma, Z. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, "Tutorial: point cloud library: three-dimensional object recognition and 6 dof pose estimation," *IEEE Robotics and Automation Magazine*, vol. 19, no. 3, pp. 80-91, Sep. 2012.

[2] R. B. Rusu and S. Cousins, "3D is here: point cloud library (pcl)," in *2011 IEEE International Conference onRobotics and Automation*, Shanghai, China, May 2011, pp. 1–4.

[3] S. Chen, D. Tian, C. Feng, A. Vetro and J. Kovačević, "Fast resampling of three-dimensional point clouds via graphs," *IEEE Transactions on Signal Processing*, vol. 66, no. 3, pp. 666-681, Feb, 2018.

[4] J. Ryde, and H. Hu, "3D mapping with multi-resolution occupied voxel lists" *Autonomous Robots*, vol. 28, no. 2, pp. 169-185. Apr. 2010.

[5] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: an efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, pp. 189–206, Apr. 2013.

[6] J. Peng and C. C. Jay Kuo, "Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition," *ACM Trans. Graph. Proceedings of ACM SIGGRAPH*, vol. 24, no. 3, pp. 609–616, Jul. 2005.

[7] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM*, vol. 45, np. 6, pp. 891–923, Nov. 1998.

[8] Y. Schoenenberger, J. Paratte, and P. Vandergheynst, "Graph-based denoising for time-varying point clouds," in *2015 3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*, Lisbon, Portugal, Aug. 2015, pp. 1-4.

[9] J. Qi, W. Hu and Z. Guo, "Feature preserving and uniformity-controllable point cloud simplification on graph," *2019 IEEE International Conference on Multimedia and Expo (ICME)*, Shanghai, China, 2019, pp. 284-289.

[10] S. Zhang, Z. Ding, and S. Cui, "Introducing hypergraph signal processing: theoretical foundation and practical applications," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 639-660, Jan. 2020.

[11] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D point cloud based object maps for household environments," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 927-941, Nov. 2008.

[12] I. Konstas, and M. Lapata, "Unsupervised concept-to-text generation with hypergraphs," in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montreal, Canada, Jun. 2012, pp.752-761.

[13] A. Banerjee, C. Arnab, and M. Bibhash, "Spectra of general hypergraphs," *Linear Algebra and its Applications*, vol. 518, pp. 14-30, Dec. 2016.

[14] S. Zhang, H. Zhang, H. Li and S. Cui, "Tensor-based spectral analysis of cascading failures over multilayer complex systems," in *Proceedings of 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Monticello, USA, Oct. 2018, pp. 997-1004.

[15] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455-500, Aug. 2009.

[16] A. Afshar, J. C. Ho, B. Dilkina, I. Perros, E. B. Khalil, L. Xiong, and V. Sunderam, "Cp-ortho: an orthogonal tensor factorization framework for spatio-temporal data," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, Redondo Beach, CA, USA, Jan. 2017, p. 1-4.

[17] J. Pan, and M. K. Ng, "Symmetric orthogonal approximation to symmetric tensors with applications to image reconstruction," *Numerical Linear Algebra with Applications*, vol. 25, no. 5, e2180, Apr. 2018.

[18] T. G. Kolda, "Orthogonal tensor decompositions," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 1, pp. 243-255, Jul. 2006.

[19] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *IEEE Transactions on Signal Processing*, vol. 65, no. 22, pp. 5911-5926, Aug. 2017.

[20] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, "Characterization and inference of graph diffusion processes from observations of stationary signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 3, pp. 481-496, Aug. 2017.

[21] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, "Connecting the dots: Identifying network structure via graph signal processing," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 16-43, May 2019.

[22] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, "Characterization and inference of graph diffusion processes from observations of stationary signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 3, pp. 481-496, Aug. 2017.

[23] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808-828, Apr. 2018.

[24] K. I. Park, *Fundamentals of probability and stochastic processes with applications to communications*. Springer International Publishing, 2018.

[25] A. Nurunnabi, D. Belton and G. West, "Robust segmentation in laser scanning 3D point cloud data," *2012 International Conference on Digital Image Computing Techniques and Applications (DICTA)*, Fremantle, WA, 2012, pp. 1-8.

[26] H. Hoppe, T. De Rose, and T. Duchamp, "Surface reconstruction from unorganized points," in *Proceedings of ACM SIGGRAPH*, vol. 26, no. 2, pp. 71–78, 1992.

[27] M. Pauly, R. Keiser, and M. Gross, "Multi-scale feature extraction on point-sampled surfaces," *Eurographics*, vol. 22, no. 3, 2003.

[28] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 3, pp. 467-483, Sep. 2017.

[29] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160-6173, Dec. 2016.

[30] M. Grant and S. Boyd. "Graph implementations for nonsmooth convex programs," *Recent Advances in Learning and Control (a tribute to M. Vidyasagar)*.

[31] V. Blondel, S. Boyd, and H. Kimura, editors, pages 95-110, *Lecture Notes in Control and Information Sciences*, Springer, 2008. http://stanford.edu/ boyd/graph_dcp.html.

[32] Y. Sun, S. Schaefer, and W. Wang, "Denoising point sets via l0 minimization," *Computer Aided Geometric Design*, vol. 35, pp. 2–15, May 2015.

[33] J. Zeng, G. Cheung, M. Ng, J. Pang, and Yang, C. "3d point cloud denoising using graph laplacian regularization of a low dimensional manifold model," 2018, arXiv:1803.07252.

[34] C. Dinesh, G. Cheung, I. V. Bajic, and C. Yang, "Fast 3d point cloud denoising via bipartite graph approximation and total variation," 2018, arXiv: 1804.10831.

[35] A. Sandryhaila, and J. M. F. Moura, "Discrete signal processing on graphs: frequency analysis," *IEEE Transactions on Signal Processing,* vol. 62, no. 12, pp. 3042-3054, Apr. 2014.

[36] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, "Numerical geometry of non-rigid shapes", Springer, 2008. ISBN: 978-0-387-73300-5.

[37] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, "Efficient computation of isometry-invariant distances between surfaces", *SIAM J. Scientific Computing*, vol. 28, no. 5, pp. 1812-1836, 2006.

[38] A. M. Bronstein *et al.*, "SHREC 2010: robust large-scale shape retrieval benchmark", *Proc. EUROGRAPHICS Workshop on 3D Object Retrieval (3DOR)*, May 2010, pp. 71–78..

[39] A. M. Bronstein *et al.*, "SHREC 2010: robust feature detection and description benchmark", *Proc. EUROGRAPHICS Workshop on 3D Object Retrieval (3DOR)*, vol. 2, no. 5, May 2010, p. 6.

[40] J. Pang, and G. Cheung, "Graph Laplacian regularization for image denoising: analysis in the continuous domain," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1770-1785, Apr. 2017.

[41] S. W. Cheng, and M. K. Lau, "Denoising a point cloud for surface reconstruction," 2017, arXiv: 1704.04038.

[42] J. Zeng, G. Cheung, M. Ng, J. Pang, and C. Yang, "3D point cloud denoising using graph Laplacian regularization of a low dimensional manifold model," *IEEE Transactions on Image Processing*, vol. 29, pp. 3474-3489, Dec. 2019.

[43] W. Hu, X. Gao, G. Cheung, and Z. Guo, "Feature graph learning for 3D point cloud denoising," *IEEE Transactions on Signal Processing*, vol.68, pp. 2841-2856, Mar. 2020.

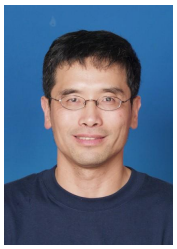
[44] L.Yi *et al.*, "Large-scale 3d shape reconstruction and segmentation from shapenet core55," 2017, arXiv: 1710.06104.

[45] A. X. Chang *et al.*, "Shapenet: an information-rich 3D model repository," 2015, arXiv: 1512.03012.

Songyang Zhang received the Bachelor of Science degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA, USA.

His current research interests include hypergraph signal processing, behavior analysis over high dimensional graphs, and learning over graphs.

Shuguang Cui (S'99-M'05-SM'12-F'14) received his Ph.D in Electrical Engineering from Stanford University, California, USA, in 2005. Afterwards, he has been working as assistant, associate, full, Chair Professor in Electrical and Computer Engineering at the Univ. of Arizona, Texas A&M University, UC Davis, and CUHK at Shenzhen respectively. He has also been the Executive Vice Director at Shenzhen Research Institute of Big Data. His current research interests focus on data driven large-scale system control and resource management, large data set analysis, IoT system design, energy harvesting based communication system design, and cognitive network optimization. He was selected as the Thomson Reuters Highly Cited Researcher and listed in the Worlds' Most Influential Scientific Minds by ScienceWatch in 2014. He was the recipient of the IEEE Signal Processing Society 2012 Best Paper Award. He has served as the general co-chair and TPC co-chairs for many IEEE conferences. He has also been serving as the area editor for IEEE Signal Processing Magazine, and associate editors for IEEE Transactions on Big Data, IEEE Transactions on Signal Processing, IEEE JSAC Series on Green Communications and Networking, and IEEE Transactions on Wireless Communications. He has been the elected member for IEEE Signal Processing Society SPCOM Technical Committee (2009 2014) and the elected Chair for IEEE ComSoc Wireless Technical Committee (2017 2018). He is a member of the Steering Committee for IEEE Transactions on Big Data and the Chair of the Steering Committee for IEEE Transactions on Cognitive Communications and Networking. He was also a member of the IEEE ComSoc Emerging Technology Committee. He was elected as an IEEE Fellow in 2013, an IEEE ComSoc Distinguished Lecturer in 2014, and IEEE VT Society Distinguished Lecturer in 2019.



Zhi Ding (S'88-M'90-SM'95-F'03) is with the Department of Electrical and Computer Engineering at the University of California, Davis, where he currently holds the position of distinguished professor. He received his Ph.D. degree in Electrical Engineering from Cornell University in 1990. From 1990 to 2000, he was a faculty member of Auburn University and later, University of Iowa. Prof. Ding has held visiting positions in Australian National University, Hong Kong University of Science and Technology, NASA Lewis Research Center and USAF Wright Laboratory. Prof. Ding has active collaboration with researchers from various universities in Australia, Canada, China, Finland, Hong Kong, Japan, Korea, Singapore, Taiwan, and USA.

Dr. Ding is a Fellow of IEEE and has been an active member of IEEE, serving on technical programs of several workshops and conferences. He was associate editor for IEEE Transactions on Signal Processing from 1994-1997, 2001-2004, and associate editor of IEEE Signal Processing Letters 2002-2005. He was a member of technical committee on Statistical Signal and Array Processing and member of technical committee on Signal Processing for Communications (1994-2003). Dr. Ding was the General Chair of the 2016 IEEE International Conference on Acoustics, Speech, and Signal Processing and the Technical Program Chair of the 2006 IEEE Globecom. He was also an IEEE Distinguished Lecturer (Circuits and Systems Society, 2004-06, Communications Society, 2008-09). He served on as IEEE Transactions on Wireless Communications Steering Committee Member (2007-2009) and its Chair (2009-2010). Dr. Ding is a coauthor of the text: Modern Digital and Analog Communication Systems, 5th edition, Oxford University Press, 2019. Prof. Ding received the IEEE Communication Society's WTC Award in 2012 and the IEEE Communication Society's Education Award in 2020.