FPS-SFT: A MULTI-DIMENSIONAL SPARSE FOURIER TRANSFORM BASED ON THE FOURIER PROJECTION-SLICE THEOREM

Shaogang Wang, Vishal M. Patel and Athina Petropulu

Department of Electrical and Computer Engineering
Rutgers, The State University of New Jersey, Piscataway, NJ 08854, USA

ABSTRACT

We propose a multidimensional sparse Fourier transform inspired by the idea of the Fourier projection-slice theorem, called FPS-SFT. FPS-SFT extracts samples along lines (1-dimensional slices from a multidimensional data cube), which are parameterized by random slopes and offsets. The discrete Fourier transform (DFT) along those lines represents projections of multidimensional DFT of the data onto those lines. The multidimensional frequencies that are contained in the signal can be reconstructed from the DFT along lines with a low sample and computational complexity provided that the signal is sparse in the frequency domain and the lines are appropriately designed. The performance of FPS-SFT is demonstrated both theoretically and numerically. A sparse image reconstruction application is illustrated, which shows the capability of the FPS-SFT in solving less sparse scenarios containing non-uniformly distributed frequencies.

***Index Terms*—** Multidimensional signal processing, sparse Fourier transform, Fourier projection-slice theorem, sparse image reconstruction

1. INTRODUCTION

Conventional signal processing methods in radar, sonar, and medical imaging systems usually involve multidimensional discrete Fourier transforms (DFT), which can be implemented by the fast Fourier transform (FFT). Recently, the sparse Fourier transform (SFT) [1–4] has been proposed, which leverages the sparsity of signals in the frequency domain to reduce the sample and computational cost of the FFT. Different versions of the SFT have been investigated for several applications including a fast Global Positioning System receiver, wide-band spectrum sensing, radar signal processing, etc. [5–9].

In all SFT algorithms the reduction of sample and computational complexity is achieved by reducing the data samples. This is implemented via a well designed, randomized subsampling procedure, which leverages the resulting frequency domain aliasing. The significant frequencies contained in the original signal are then localized and the corresponding DFT coefficients are estimated with low-complexity operations. Such *subsampling-localization-estimation* procedure is carried out in an *iterative* manner in several SFT algorithms [2, 4, 10, 11], while for other SFT algorithms [7–9, 12], the localization and estimation are implemented in *one-shot* after gathering sufficient copies of subsampled signals corresponding to different subsampling parameters, e.g., subsample rate, offset and number of samples. Generally, iterative SFT algorithms exhibit lower complexity as compared to non-iterative SFT algorithms, since in the former, in each iteration, the contribution of the recovered significant frequencies are removed from the signal, which yields a sparser signal (an easier problem) in the next iteration.

Multidimensional signal processing requires multidimensional SFT algorithms. Most of the existing SFT algorithms, however, are designed for 1-dimensional (1-D) signals and their extension to multidimensional signals is typically not straightforward. Multidimensional SFT algorithms are investigated in [2, 8, 9]. The idea behind those SFT algorithms is to reduce a multidimensional DFT into a number of 1-D DFTs along lines extracted from the input multidimensional data. The sample-optimal SFT (SO-SFT) achieves the sample and computational complexity lower bounds of all known SFT algorithms by reducing a 2-dimensional (2-D) DFT into 1-D DFTs along columns and rows of a data matrix; in the frequency domain, this results into projection of 2-D frequencies onto the two axis of the matrix. Under the assumption that the frequencies are sparse and uniformly distributed, the 2-D frequencies are not likely to collide (more than one frequencies are projected into the same bin) in the projection, and thus the 2-D frequencies can be effectively recovered from their 1-D projections (see Section 2.1 for details). The DFT along columns and rows provides two degrees of freedom in terms of the direction of the frequency projection; a frequency collision in the column direction may not happen in the row direction and vise versa. However, when frequencies are less sparse, it often results in unrecoverable frequencies (see Fig. 1).

In order to reduce the chance of deadlock, the SFT of [8, 9] extends the degrees of freedom of frequency projection by applying 1-D DFT not only along the axis of the input multidimensional data cube, but also along a few lines of predefined and deterministic slopes extracted form the data cube. Although employing lines with various slopes leads to more degrees of freedom in frequency projection, the limited choice of line slopes in [8, 9] is still an obstacle in addressing less sparse signals. Moreover, the sample and computational complexity of SFT of [8, 9] are higher than that of SO-SFT, as the former method applies the one-shot based architecture while the latter follows the iterative approach.

In this work we propose FPS-SFT, a multidimensional, Fourier projection-slice based SFT, which enjoys low complexity while avoiding the limitations of the aforementioned algorithms, i.e., it can handle less sparse data in the frequency domain, and accommodate frequencies that are non-uniformly distributed. FPS-SFT uses the low-complexity framework of SO-SFT, and extends the multiple slopes idea of [8, 9] by using lines of randomly generated slopes. The abundance of randomness of line slopes enables a large number of degrees of freedom in frequency projection. Thus, less sparse, non-uniformly distributed frequencies can be effectively recovered.

FPS-SFT can be viewed as a low-complexity, Fourier projection-slice approach for signals that are sparse in the frequency domain. In FPS-SFT, the DFT along a 1-D slice (discrete line) of the multidi-

The work was supported by ARO grant W911NF-16-1-0126, NSF Grant ECCS-1408437, China Scholarship Council and Shanghai Institute of Spaceflight Electronics Technology.

mensional data is the projection of the multidimensional DFT of the data to such line. While the classical Fourier projection-slice based method either reconstructs the frequency domain of the signal via interpolation of frequency-domain slices, or reconstructs the time-domain samples via solving linear equation systems that relate the DFT along projections and the time-domain samples, the proposed FPS-SFT aims to reconstruct the signal directly based on frequency domain projections; this is achieved by leveraging the sparsity of the signal in the frequency domain.

Notation: We use lower-case bold letters to denote vectors. $[\cdot]^T$ denotes the transpose of a vector. The N-modulo operation is denoted by $[\cdot]_N$. $[S]$ refers to the integer set of $\{0, ..., S-1\}$. The cardinality of set $\mathbb{S}$ is denoted as $|\mathbb{S}|$. The DFT of signal x is denoted by $\hat{x}$.

2. THE FPS-SFT ALGORITHM

Let us consider the following 2-D signal model, which is a superposition of K 2-D complex sinusoids, i.e.,

$$x(\mathbf{n}) \triangleq \sum_{(a,\boldsymbol{\omega})\in\mathbb{S}} a e^{j\mathbf{n}^T\boldsymbol{\omega}},\ \mathbf{n} \triangleq [n_0, n_1]^T \in \mathcal{X} \triangleq [N_0] \times [N_1], \quad (1)$$

where N_0, N_1 denote the sample length of the two dimensions, respectively. $(a, \boldsymbol{\omega})$ represents a 2-D frequency whose amplitude is a with $a \in \mathbb{C}, a \neq 0$ and frequency is $\boldsymbol{\omega} \triangleq [\omega_0, \omega_1]^T$ with $\omega_k = \frac{2\pi}{N_k} m_k, m_k \in [N_k], k \in \{0, 1\}$. The set $\mathbb{S}$ with $|\mathbb{S}| = K$ includes all the 2-D frequencies. We assume that the signal is sparse in the frequency domain, i.e., $K << N \triangleq N_0 N_1$. We are interested in the recovery of all the frequencies from samples of $x(\mathbf{n})$. While we consider 2-D signals here, the generalization to higher dimensions, i.e., D-D cases with $D > 2$, is straightforward.

2.1. SO-SFT

In SO-SFT, 1-D DFTs are applied on a subset of columns and rows of the data. The N_0-point DFT of the i-th, $i \in [N_1]$ column of the data equals

$$\begin{aligned}\hat{c}_i(m) &\triangleq \frac{1}{N_0} \sum_{l\in[N_0]} x(l, i) e^{-j\frac{2\pi}{N_0}ml} \\ &= \frac{1}{N_0} \sum_{(a,\boldsymbol{\omega})\in\mathbb{S}} \sum_{l\in[N_0]} a e^{j\frac{2\pi}{N_1}m_1 i} e^{j\frac{2\pi}{N_0}l(m_0-m)} \\ &= \sum_{(a,\boldsymbol{\omega})\in\mathbb{S}} a e^{-j\frac{2\pi}{N_0}m_1 i} \delta(m - m_0),\ m \in [N_0], \\ \boldsymbol{\omega} &= [2\pi m_0/N_0, 2\pi m_1/N_1]^T, [m_0, m_1]^T \in \mathcal{X},\end{aligned} \quad (2)$$

where $\delta(\cdot)$ is the Dirac delta function. Hence $\hat{c}_i(m_0)$ can be viewed as the summation of modulated amplitudes of the 2-D sinusoids, whose row frequency indices equal to m_0. Hence $\hat{c}_i(m), m \in [N_1]$ is a projection of the 2-D spectrum, $\hat{x}(m_0, m_1), [m_0, m_1]^T \in \mathcal{X}$, onto the column. Similarly, the N_1-point DFT applied on a row of (1) is a projection of the 2-D spectrum on the row.

Since the signal is sparse in the frequency domain, if $|\hat{c}_i(m)| \neq 0$, with high probability, there will be only one significant frequency projected to the frequency bin of m; in other words, the frequency bin is a '1-sparse', and $\hat{c}_i(m)$ is reduced to $\hat{c}_i(m) = \hat{c}_i(m_0) = a e^{j\frac{2\pi}{N_1}m_1 i}$. The amplitude, a, can be determined by the m_0-th entry of the DFT of the 0-th column, i.e., $a = \hat{c}_0(m_0)$, and the other frequency index, m_1, is 'coded' in the phased difference between the m_0-th entries of the DFTs of the 0-th and the 1-st columns, which can be decoded by $m_1 = \phi\left(\hat{c}_1(m_0)/\hat{c}_0(m_0)\right)\frac{N_1}{2\pi}$, where $\phi(x)$ is the phase of x. Note that the 1-sparsity of the m-th bin can be effectively tested by comparing $|\hat{c}_0(m)|$ and $|\hat{c}_1(m)|$; $\hat{c}_i(m)$ is 1-sparse when $|\hat{c}_0(m)| = |\hat{c}_1(m)|$. Such frequency decoding technique is referred to as OFDM-trick [1]. The contribution of the recovered sinusoids is removed from the signal, so that the following processing can be applied on a sparser signal. A frequency bin that is not 1-sparse based on column processing might be 1-sparse based on row processing. Also, the removal of sinusoids related to the recovered frequencies in the previous column (row) processing may cause bins in the row (column) processing to be 1-sparse. SO-SFT runs iteratively and alternatively between columns and rows, and stops after a finite number of iterations.

SO-SFT succeeds with high probability only when the frequencies are very sparse, and requires that either a row or a column of the DFT contains a 1-sparse bin. However, in many applications, the signal frequency exhibits a block sparsity pattern [13], i.e., the significant frequencies are clustered. In those cases, even when the signal is very sparse, 1-sparse bin may not exist; this is referred to as a 'deadlock' case [2].

2.2. FPS-SFT

SO-SFT reduces a 2-D DFT into 1-D DFTs of the columns and rows of the input data matrix. The columns and the rows can be viewed as discrete lines of the input data matrix with slopes ∞ and 0, respectively. In this section, by proposing FPS-SFT, we reduce the 2-D DFT into 1-D DFTs of the data along discrete lines with random slopes and offsets; this results into random directions of frequency projection, and thus offers a high probability of creating more 1-sparse bins, which resolves the deadlocks encountered by SO-SFT. This is illustrated in Fig. 1, where the 4 2-D frequencies in the 8×8-point DFT domain form a deadlock, as neither a row DFT nor a column DFT creates a 1-sparse bin. However, the DFT along the diagonal, corresponding to the projection of the 2-D DFT of data onto the diagonal, produces 4 1-sparse bins, avoiding a deadlock.

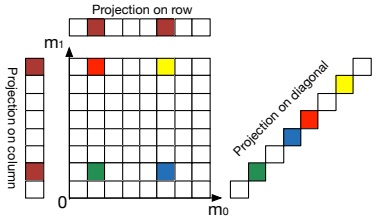


Fig. 1. Demonstration of projection of 2-D frequencies onto 1-D. The colored blocks mark significant frequencies.

FPS-SFT is an iterative algorithm; each iteration returns a subset of recovered 2-D frequencies. After T iterations, the FPS-SFT returns a set, $\hat{\mathbb{S}}$, which is an estimate of $\mathbb{S}$ of (1). The frequencies recovered in previous iterations are passed to the next iteration, and their contributions are removed from the signal.

Within each iteration of FPS-SFT, the signal of (1) is sampled along a line with slope α_1/α_0 and offset $\boldsymbol{\tau}$, with $\boldsymbol{\alpha}, \boldsymbol{\tau} \in \mathcal{X}$, where $\boldsymbol{\alpha} \triangleq [\alpha_0, \alpha_1]^T, \boldsymbol{\tau} \triangleq [\tau_0, \tau_1]^T$. The sampled signal can be expressed as

$$\begin{aligned}s(\boldsymbol{\alpha}, \boldsymbol{\tau}, l) &\triangleq x([\alpha_0 l + \tau_0]_{N_0}, [\alpha_1 l + \tau_1]_{N_1}) \\ &= \sum_{(a,\boldsymbol{\omega})\in\mathbb{S}} a e^{j2\pi\left(\frac{m_0[\alpha_0 l+\tau_0]_{N_0}}{N_0} + \frac{m_1[\alpha_1 l+\tau_1]_{N_1}}{N_1}\right)}, l \in [L].\end{aligned} \quad (3)$$

Taking an L-point DFT on (3), for $m \in [L]$, we get

$$\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m) \triangleq \frac{1}{L} \sum_{l \in [L]} s(\boldsymbol{\alpha}, \boldsymbol{\tau}, l) e^{-j2\pi \frac{lm}{L}}$$
$$= \frac{1}{L} \sum_{(a, \boldsymbol{\omega}) \in \mathbb{S}} a e^{j2\pi \left(\frac{m_0 \tau_0}{N_0} + \frac{m_1 \tau_1}{N_1} \right)} \sum_{l \in [L]} e^{j2\pi l \left(\frac{m_0 \alpha_0}{N_0} + \frac{m_1 \alpha_1}{N_1} - \frac{m}{L} \right)}. \tag{4}$$

Let us assume that for all $m \in [L]$, $[m_0, m_1]^T \in \mathcal{X}$,

$$\hat{f}(m) \triangleq \frac{1}{L} \sum_{l \in [L]} e^{j2\pi l \left(\frac{m_0 \alpha_0}{N_0} + \frac{m_1 \alpha_1}{N_1} - \frac{m}{L} \right)} \in \{0, 1\}. \tag{5}$$

This holds when $\frac{m_0 \alpha_0}{N_0} + \frac{m_1 \alpha_1}{N_1} - \frac{m}{L}$ is multiple of $1/L$, which requires the line length L and slope $\boldsymbol{\alpha}$ satisfying Lemma 1.

When $\hat{f}(m) = 1$, i.e.,

$$\left[\frac{m_0 \alpha_0}{N_0} + \frac{m_1 \alpha_1}{N_1} - \frac{m}{L} \right]_1 = 0, [m_0, m_1]^T \in \mathcal{X}, \tag{6}$$

where $[\cdot]_1$ is modulo of 1, (4) can be simplified as $\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m) = \sum_{(a, \boldsymbol{\omega}) \in \mathbb{S}} a e^{j2\pi \left(\frac{m_0 \tau_0}{N_0} + \frac{m_1 \tau_1}{N_1} \right)}$.

As it can be shown by the proof of Lemma 2, the points (m_0, m_1) satisfying (6) lie on a line with slope $-\alpha_0 N_1 / (\alpha_1 N_0)$ in the $N_0 \times N_1$-point DFT domain, i.e.,

$$m_0 = [m_0' + k \alpha_1 L / N_1]_{N_0}, m_1 = [m_1' - k \alpha_0 L / N_0]_{N_1}, k \in \mathbb{Z}, \tag{7}$$

where $[m_0', m_1']^T \in \mathcal{X}$ is one of the solutions of (6). Hence, the time domain line and the corresponding frequency domain line are orthogonal to each other.

Each entry of the L-point DFT of samples along a time-domain line with slope α_1/α_0 represents a projection of the 2-D DFT along the line with slope $-\alpha_0 N_1/(\alpha_1 N_0)$ in the $N_0 \times N_1$-point DFT domain, which is orthogonal to the time-domain line. This is closely related to the Fourier projection-slice theorem, which states that the Fourier transform of a projection is a slice of the Fourier transform of the projected object. While the classical projection is in the time domain and the corresponding slice is in the frequency domain, in the FPS-SFT case, the projection is in the DFT domain and the corresponding slice is in the sample (discrete-time) domain. The important difference between the Fourier projection-slice theorem and FPS-SFT is that while the former reconstructs the frequency domain of the signal via interpolation of frequency domain slices, the latter efficiently recovers the significant frequencies of the signal directly based on the DFT of time-domain 1-D slices, i.e., samples along random lines, which involves lower complexity. The efficiency of FPS-SFT is achieved by exploring the sparsity nature of the signal in the frequency domain, which is explained in the following.

We apply the assumption that the signal is sparse in the frequency domain; specifically, we assume that $|\mathbb{S}| = O(L)$. Then, if $|\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m)| \neq 0$, with high probability, the m-th bin is 1-sparse, and it holds that $\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m) = a e^{j2\pi \left(\frac{m_0 \tau_0}{N_0} + \frac{m_1 \tau_1}{N_1} \right)}, (a, \boldsymbol{\omega}) \in \mathbb{S}$. In such case, the 2-D frequency, $(a, \boldsymbol{\omega})$, can be 'decoded' using three lines of the same slope but different offsets, i.e., $\boldsymbol{\tau}, \boldsymbol{\tau}_0 \triangleq [[\tau_0 + 1]_{N_0}, \tau_1]^T, \boldsymbol{\tau}_1 \triangleq [\tau_0, [\tau_1 + 1]_{N_1}]^T$, respectively. Such design allows the frequencies to be decoded independently in each dimension. The frequency corresponding to the 1-sparse bin, m, can be decoded as

$$m_0 = \left[\frac{N_0}{2\pi} \phi \left(\frac{\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}_0, m)}{\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m)} \right) \right]_{N_0},$$
$$m_1 = \left[\frac{N_1}{2\pi} \phi \left(\frac{\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}_1, m)}{\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m)} \right) \right]_{N_1}, \tag{8}$$
$$a = \hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m) e^{-j2\pi (m_0 \tau_0 / N_0 + m_1 \tau_1 / N_1)}.$$

In order to recover all the frequencies in $\mathbb{S}$ efficiently, each iteration of FPS-SFT adopts a random choice of line slope (see Lemma 2) and offset. Furthermore, the contribution of the recovered sinusoids in the previous iterations is removed via a *construction-subtraction* approach so that the signal becomes sparser in future iterations. Specifically, assuming that for the current iteration the line slope and offset parameters are $\boldsymbol{\alpha}, \boldsymbol{\tau}$, respectively, the recovered 2-D frequencies are projected into L frequency bins to construct the DFT along the corresponding line, i.e., $\hat{s}_r(\boldsymbol{\alpha}, \boldsymbol{\tau}, m) \triangleq \sum_{(a, \boldsymbol{\omega}) \in \mathcal{I}_m} a e^{j2\pi \left(\frac{m_0 \tau_0}{N_0} + \frac{m_1 \tau_1}{N_1} \right)}$, $m \in [L]$, where $\mathcal{I}_m, m \in [L]$ represent the subsets of the recovered frequencies, i.e., $\mathcal{I}_m \triangleq \{(a, \boldsymbol{\omega}) : [\frac{m_0 \alpha_0}{N_0} + \frac{m_1 \alpha_1}{N_1} - \frac{m}{L}]_1 = 0, [m_0, m_1]^T \in \mathcal{X}\}, m \in [L]$. Next, the L-point inverse DFT (IDFT) is applied on $\hat{s}_r(\boldsymbol{\alpha}, \boldsymbol{\tau}, m), m \in [L]$, from which the line, $s_r(\boldsymbol{\alpha}, \boldsymbol{\tau}, l), l \in [L]$ due to the previously recovered frequencies is constructed. Subsequently, the line points are subtracted from the signal samples of the current iteration.

2.3. Analysis of FPS-SFT

In this section, we provide the design details of the critical parameters in FPS-SFT, i.e., line length and slope; those parameters are designed such that the projections in the frequency domain are orthogonal and uniform. The orthogonality is necessary for the projected multidimensional frequencies and the corresponding DFT coefficients to be exactly recoverable. The uniformity of the projection means that the DFT coefficients of N grid locations of the $N_0 \times N_1$-point DFT are uniformly projected to the L entries of the L-point DFT along a line. Compared with a non-uniform projection, the uniform projection creates more 1-sparse bins, which allows for fewer iterations of FPS-SFT to exactly reconstruct the signal.

Lemma 1. ***(Line Length):*** *Let $s(\boldsymbol{\alpha}, \boldsymbol{\tau}, l) = x([\alpha_0 l + \tau_0]_{N_0}, [\alpha_1 l + \tau_1]_{N_1})$ with $l \in [L], \boldsymbol{\alpha} \triangleq [\alpha_0, \alpha_1]^T, \boldsymbol{\tau} \triangleq [\tau_0, \tau_1]^T \in \mathcal{X}$ be a discrete line extracted from the signal model of (1). Then $\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m), m \in [L]$ is the orthogonal projection of $\hat{x}(\mathbf{m}), \mathbf{m} \in \mathcal{X}$ onto such line when L is the least common multiple of N_0, N_1, i.e., $L = \mathrm{LCM}(N_0, N_1)$. Moreover, L is the minimum length of a line to allow orthogonal projections in the DFT domain with arbitrary choice of $\boldsymbol{\alpha} \in \mathcal{X}$.*

Lemma 2. ***(Line Slope):*** *Let $s(\boldsymbol{\alpha}, \boldsymbol{\tau}, l) = x([\alpha_0 l + \tau_0]_{N_0}, [\alpha_1 l + \tau_1]_{N_1})$ with $l \in [L], L = \mathrm{LCM}(N_0, N_1), \boldsymbol{\alpha} \triangleq [\alpha_0, \alpha_1]^T \in \mathcal{A} \subset \mathcal{X}, \boldsymbol{\tau} \triangleq [\tau_0, \tau_1]^T \in \mathcal{X}$ be a discrete line extracted from the signal model of (1), where $\mathcal{A} \triangleq \{\boldsymbol{\alpha} : \boldsymbol{\alpha} \in \mathcal{X}, (\alpha_0, \alpha_1), (\alpha_0, c_1), (\alpha_1, c_0)$ are co-prime$\}$, where $c_0 = L/N_0, c_1 = L/N_1$. Then each entry of $\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m), m \in [L]$ is the projection of N/L distinct samples of $\hat{x}(\mathbf{m}), \mathbf{m} \in \mathcal{X}$, which locate in $\mathcal{P}_m \triangleq \{[m_0, m_1]^T : [\frac{m_0}{N_0} \alpha_0 + \frac{m_1}{N_1} \alpha_1 - \frac{m}{L}]_1 = 0, [m_0, m_1]^T \in \mathcal{X}\}$. Moreover, $|\mathcal{P}_m| = N/L, \mathcal{P}_m \cap \mathcal{P}_{m'} = \emptyset$ for $m \neq m', m, m' \in [L]$. Thus, $\hat{x}(\mathbf{m}), \mathbf{m} \in \mathcal{X}$ is uniformly projected to $\hat{s}(\boldsymbol{\alpha}, \boldsymbol{\tau}, m), m \in [L]$.*

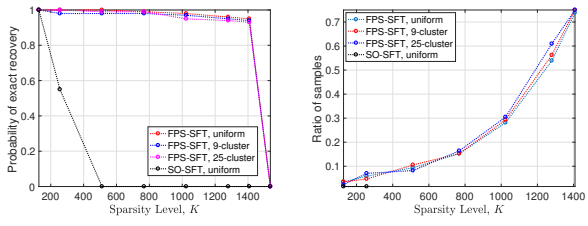
Multidimensional extension: For the D-D case with the data cube size of $N_0 \times N_1 \times \cdots N_{D-1}$, the line length can be set

as $L = \text{LCM}(N_0, \cdots, N_{D-1})$; the slope and offset parameters $[\alpha_0, \cdots, \alpha_{D-1}]^T, [\tau_0, \cdots, \tau_{D-1}]^T$ is randomly taken from $\mathcal{X}_D \triangleq [N_0] \times [N_1] \times \cdots [N_{D-1}]$. Each iteration extracts $D+1$ L-length lines with a same random slope but different offsets from the D-D data cube. The 0-th line offset is set to be $[\tau_0, \cdots, \tau_{D-1}]^T$, while for the $(i+1)$-th line with $0 \leq i \leq D-1$, the offset for the i-th dimension is set to be $[\tau_i + 1]_{N_i}$. With such offset parameters, the frequencies can be decoded independently for each dimension.

Complexity analysis: The FPS-SFT executes T iterations; in the 2-D case, each iteration uses $3L$ samples, since 3 L-length lines, with $L = \text{LCM}(N_0, N_1)$ are extracted in order to decode the two frequency components of a 2-D sinusoid (see (8)). Hence, the sample complexity of FPS-SFT is $O(3TL) = O(L)$. The core processing of FPS-SFT is the L-point 1-D DFT, which can be implemented by the FFT with the computational complexity of $O(L \log L)$. The L-point IDFT in the construction-subtraction procedure can also be implemented by the FFT. In addition to the FFT, each iteration needs to evaluate $O(K)$ frequencies. Hence the computational complexity of FPS-SFT is $O(L \log L + K)$. Assuming that $K = O(L)$, then the sample and computational complexity can be simplified as $O(K)$ and $O(K \log K)$, respectively, which achieves the lowest sample and computational complexities, respectively, of known state-of-the-art SFT algorithms [2, 4]. Generally, in the D-D case, it is easy to see that the sample and computational complexity of FPS-SFT are $O(DK)$ and $O(DK \log(DK))$, respectively, when $K = O(L)$.

3. NUMERICAL RESULTS

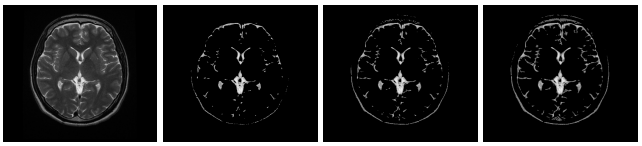
Comparison to SO-SFT: The length of the two dimensions are set to $N_0 = N_1 = 256$. We simulate two scenarios, when frequencies are uniformly distributed and when they are clustered. For the clustered case, we consider clusters of 9 and 25 frequencies. When $N_0 = N_1$, the line length, L, of FPS-SFT equals N_0, and each iteration of FPS-SFT uses $3N_0$ samples. We limit the maximum iterations to $T_{max} = N/(3L) \approx 85$; this corresponds to roughly 100% samples of the input data. Fig. 2 (a) shows the probability of exact recovery versus level of sparsity for FPS-SFT and SO-SFT. When the signal is very sparse, i.e., $K = N_0/2$, SO-SFT has high probability of exact recovery, while it fails when the sparsity is moderately large, i.e., $K = 2N_0$. Moreover, SO-SFT only works for the scenario in which frequencies are distributed uniformly, while it fails when there exists even a single frequency cluster. On the contrary, FPS-SFT applies to signals with a wide range of sparsity levels. For instance, the success rate of FPS-SFT is approximate 97% when $K = 5N_0$. In all cases, the success rates drop to 0 when $K = 6N_0$, since we set $T_{max} = 85$. To achieve an exact recovery, the FPS-SFT needs to run for roughly 100 iterations when $K = 6N_0$. Fig. 2 (b) shows the ratio of samples used by the FPS-SFT for exact recovery to the total number of data sample N versus different sparsity level for the uniform and clustered cases; the result for SO-SFT is also shown. The figure shows that the sparser the signal, the fewer samples are required by the FPS-SFT. For example, when $K = N_0$, only 5.9% of the signal samples are required in the uniform-distributed frequency case or the clustered case. In the case of SO-SFT, in principle, SO-SFT only needs two rows and two columns of the data matrix to process in a very sparse case, which is only 1.6% of the signal samples. However, when the frequency is less sparse or non-uniformly distributed, the SO-SFT fails. Fig. 2 also shows that the performance of FPS-SFT is similar for both uniform-distributed and clustered frequency cases at the same sparsity level. The good performance of FPS-SFT arises because the randomized projections can effectively isolate the 2-D frequencies into 1-sparse bins, even when the signal is less sparse (K is large) and the frequencies are clustered.



(a) (b)

Fig. 2. Compared to SO-SFT, FPS-SFT can achieve exact recovery of less sparse data of non-uniformly distributed frequencies with high probability. The clustered frequency cases for SO-SFT is not shown as the SO-SFT fails in such cases. (a) Probability of exact recovery versus sparsity level, K. (b) Ratio of samples (the averaged number of samples used by FPS-SFT over N) needed for exact recovery versus K.

Sparse image reconstruction: Due to the duality of the time and frequency, the FPS-SFT is able to reconstruct a signal that is sparse in the time (spatial) domain using the samples in the frequency domain. Here we demonstrate the ability of FPS-SFT to recover images that are sparse in the pixel domain. Such sparse image recovery problem arises in the MRI applications [14]. In MRI, samples are directly taken from the frequency domain, from which the images reflecting the inner structure of the examined objects are reconstructed. Fig. 3 (a) shows a 512×576-pixel brain MRI image [14]. This image is sparsified by applying thresholding on the original image. Next, we convert the sparsified images in the frequency domain via a 512×576-point DFT, on which the 2-D FPS-SFT is applied to reconstruct the images. Figs. 3 (b), (c) and (d) show that the images with 2.85%, 4.48% and 6.61% of non-zero pixels can be perfectly reconstructed by FPS-SFT using 14.0%, 23.4%, and 70.3% samples in the frequency domain, respectively.



(a) (b) (c) (d)

Fig. 3. Image reconstruction. (a) Raw image. (b) 2.9%-sparse, $K = 8411$. (c) 4.5%-sparse, $K = 13219$. (d) 6.6%-sparse, $K = 19506$.

4. CONCLUSION

We have proposed the FPS-SFT, a low-complexity, multidimensional SFT algorithm based on the idea of the Fourier projection-slice theorem. Theoretical and numerical results of FPS-SFT have been provided. While FPS-SFT has achieved good performance on the ideal signal, i.e., exactly sparse signal containing on-grid frequencies, the real-life signals are typically noisy and contain off-grid frequencies. The noise introduces difficulties in the frequency decoding of FPS-SFT, and the leakage generated by the off-grid frequencies destroys the sparsity in the frequency domain. In our future research, we will develop an extension of FPS-SFT to deal with these cases.

5. REFERENCES

[1] Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price, "Nearly optimal sparse Fourier transform," in *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. ACM, 2012, pp. 563–578.

[2] Badih Ghazi, Haitham Hassanieh, Piotr Indyk, Dina Katabi, Erik Price, and Lixin Shi, "Sample-optimal average-case sparse Fourier transform in two dimensions," in *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*. IEEE, 2013, pp. 1258–1265.

[3] Daniel Potts and Toni Volkmer, "Sparse high-dimensional FFT based on rank-1 lattice sampling," *Applied and Computational Harmonic Analysis*, 2015.

[4] S. Pawar and K. Ramchandran, "FFAST: An algorithm for computing an exactly k-sparse DFT in O(klogk) time," *IEEE Transactions on Information Theory*, vol. PP, no. 99, pp. 1–1, 2017.

[5] Haitham Hassanieh, Fadel Adib, Dina Katabi, and Piotr Indyk, "Faster GPS via the sparse Fourier transform," in *Proceedings of the 18th annual international conference on Mobile computing and networking*. ACM, 2012, pp. 353–364.

[6] Haitham Hassanieh, Lixin Shi, Omid Abari, Ezz Hamed, and Dina Katabi, "GHz-wide sensing and decoding using the sparse Fourier transform," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 2256–2264.

[7] Shaogang Wang, Vishal M Patel, and Athina Petropulu, "The robust sparse Fourier transform (RSFT) and its application in radar signal processing," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 6, pp. 2735–2755, 2017.

[8] Lixin Shi, Haitham Hassanieh, Abe Davis, Dina Katabi, and Fredo Durand, "Light field reconstruction using sparsity in the continuous Fourier domain," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 1, pp. 12, 2014.

[9] Haitham Hassanieh, Maxim Mayzel, Lixin Shi, Dina Katabi, and Vladislav Yu Orekhov, "Fast multi-dimensional NMR acquisition and processing using the sparse FFT," *Journal of Biomolecular NMR*, pp. 1–11, 2015.

[10] Anna C Gilbert, Martin J Strauss, and Joel A Tropp, "A tutorial on fast Fourier sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 57–66, 2008.

[11] Anna C Gilbert, Piotr Indyk, Mark Iwen, and Ludwig Schmidt, "Recent developments in the sparse Fourier transform: A compressed Fourier transform for big data," *Signal Processing Magazine, IEEE*, vol. 31, no. 5, pp. 91–100, 2014.

[12] Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price, "Simple and practical algorithm for sparse Fourier transform," in *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*. 2012, SODA '12, pp. 1183–1194, SIAM.

[13] Yonina C Eldar, Patrick Kuppinger, and Helmut Bolcskei, "Block-sparse signals: Uncertainty relations and efficient recovery," *IEEE Transactions on Signal Processing*, vol. 58, no. 6, pp. 3042–3054, 2010.

[14] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, "Compressed sensing MRI," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 72–82, March 2008.