A Sparse Multi-Dimensional Fast Fourier Transform with Stability to Noise in the Context of Image Processing and Change Detection

Pierre-David Letourneau, M. Harper Langston, and Richard Lethin
Reservoir Labs
letourneau,langston,lethin@reservoir.com

***Abstract*—We present the sparse multidimensional FFT (sMFFT) for positive real vectors with application to image processing. Our algorithm works in any fixed dimension, requires an (almost)-optimal number of samples ($\mathcal{O}\left(R\log\left(\frac{N}{R}\right)\right)$) and runs in $\mathcal{O}\left(R\log\left(\frac{N}{R}\right)\right)$ complexity (to first order) for N unknowns and R nonzeros. It is stable to noise and exhibits an exponentially small probability of failure. Numerical results show sMFFT's large quantitative and qualitative strengths as compared to ℓ_1-minimization for Compressive Sensing as well as advantages in the context of image processing and change detection.**

I. Introduction

Since its popularization in the 1960s [1], the Fast Fourier Transform (FFT) algorithm has played a crucial role in multiple areas of computational mathematics and has rendered possible great achievements in scientific computing [2], signal processing [3] and computer science [4] by reducing the computational cost of computing the Discrete Fourier Transform (DFT) of an N-vector from $\mathcal{O}(N^2)$ to $\mathcal{O}(N\log(N))$.

When the vector to be recovered is sparse, it is possible to significantly improve on this cost [5]–[7]. That is, if f is a $N\times 1$ vector corresponding to the DFT of a noisy $N\times 1$ vector $\hat{f}$ containing at most R nonzero elements, it is possible to recover $\hat{f}$ using a number of samples much smaller than the traditional "Nyquist rate" ($\ll\mathcal{O}(N)$), and in computational complexity much lower than that of the FFT ($\ll\mathcal{O}(N\log(N))$). These schemes are generally referred to as "sparse Fast Fourier Transform" (sFFT) algorithms and fall into two main categories: deterministic (e.g., [7]) or randomized (e.g., [5]) algorithms. Of the two, randomized algorithms have had the most success in practice. Sparse FFT algorithms can further be split into more categories: $1D$ versus multidimensional (d), and exact versus noisy measurements. Also, among the fastest known sFFT algorithms, the probability of failure p of randomized algorithms decays slowly with N [5] and or *a priori* knowledge about the support is required (e.g., contained within a cyclic interval [6], [7]). Further, many require the unknowns N to be a power of two [6], [8], only work in one or two dimensions [6], [8] or do not scale well with ambient dimension d [5], [9].

The patent-pending sparse Multidimensional Fast Fourier Transform (sMFFT)[1] algorithm we have developed provably achieves the best known sample and computational complexities while optimizing over key features as detailed here:

- Requires $\mathcal{O}\left(R\sqrt{\log(R)}\log\left(\frac{N}{R}\right)\right)$ samples and $\mathcal{O}\left(R\log^{3/2}(R)\log\left(\frac{N}{R}\right)\right)$ computational complexity;
- Cost scales linearly with dimension d;
- Exponentially small probability of failure;
- Stable to noise;
- Works for any positive integer N;
- Works for positive real vectors only.

In this sense, our algorithm matches the optimal sampling complexity [10] (within a $\log(\log(N))^{-1}$ factor), and the best time complexity achieved so far [6], [7]. Additionally, the sMFFT is the first multidimensional sFFT that scales linearly with dimension. Indeed, most previous endeavors were targeted at the $1D$ sparse FFT, whereas the only currently-existing multidimensional versions [5], [9] have a cost that scales at least exponentially with dimension. Finally, our algorithm does share with [6] the characteristic of being designed for real positive vectors only, but for many applications of interest, e.g. radar imaging [11], this is a valid hypothesis.

Below, we describe the ideas behind sparse FFTs in Section II, followed by key details of of our sMFFT algorithm in Section III. Numerical results are shown in Section V, highlighting the strengths of our approach over standard FFTs as well as Compressive Sensing [12].

II. Description of the Sparse FFT Problem

In Table I, we summarize all quantities of interest. We begin by describing the $1D$ scheme. See Section IV for the

Symbol	Description
$f(\cdot)$	Periodic, bandlimited function of interest.
$\hat{f}$	Vector containing Fourier coefficients of $f(\cdot)$.
$\mathcal{S}$	Set containing indices of nonzero elements of $\hat{f}$, i.e., support
N	Total number of unknowns
R	Upper bound on cardinality of the support $\mathcal{S}$ of $\hat{f}$
d	Ambient dimension
p	Probability of failure of randomized algorithm
C	Algorithmic constant

TABLE I
Summary of important quantities

[1]U.S. Patent Application No. 62/286,732

978-1-5090-3525-0/16/$31.00 ©2016 IEEE

multidimensional version. For some finite $0 < N \in \mathbb{N}$ and vector $\{\hat{f}_j\}_{j=0}^{N-1} \geq 0$, assume $f(x)$ takes the form

$$f(x) = \sum_{j=0}^{N-1} e^{-2\pi i\, x\, j}\ \hat{f}_j\,, \tag{1}$$

Further, assume $\hat{f}$ has sparsity level to $0 < R < N$; that is, the *support* $\mathcal{S} := \left\{n \in \{0, 1, ..., N-1\} : |\hat{f}_n| \neq 0\right\}$ of $\hat{f}$ has cardinality $0 \leq \#\mathcal{S} \leq R < N < \infty$, where the *positivity* requirement dictates $\hat{f}_j > 0$ if $j \in \mathcal{S}$ and 0 otherwise. Finally, The size-N Discrete Fourier Transform (DFT),

$$f_{n;N} = \left[F_N\, \hat{f}\right]_n = \frac{1}{N} \sum_{j=0}^{N-1} e^{-2\pi i \frac{n\, j}{N}}\ \hat{f}_j,\ n = 0, 1, ..., N-1, \tag{2}$$

with inverse Fourier transform $\mathcal{F}^*$. We are particularly interested in the case where $R \ll N$. Consider some vector $\hat{f}$ satisfying the positivity requirement with samples of a "clean" signal plus some additive noise. The sparse FFT problem then involves computing an approximation to $\hat{f}$ from the knowledge of $\mathcal{O}\left(R \log\left(\frac{N}{R}\right)\right)$ samples,

$$f_{n;N} = f_{n;N}^{\text{noiseless}} + \eta\, \nu_{n;N} = \sum_{j=0}^{N-1} e^{-2\pi i\, \frac{n j}{N}}\ \hat{f}_j + \eta\, \nu_{n;N}$$

and in $\mathcal{O}\left(R \log\left(\frac{N}{R}\right)\right)$ computational complexity, where $0 < \eta < 1$ is some small *noise* parameter. It is further assumed that there is access to a decomposition of $N = K \prod_{i=1}^{P} \rho_i$, where $2 \leq \rho_i \leq \rho$, $K = \mathcal{O}(\rho R)$ and $P = \mathcal{O}\left(\log\left(\frac{N}{K}\right)\right)$. We have developed the fastest available way to compute a multidimensional DFT of a bandlimited and periodic function $f(x)$ satisfying Eq.(1) with positivity; we begin by outlining the steps for the one-dimensional case and then show how to extend to arbitrarily large dimensions.

III. Our Sparse FFT in One Dimension

The sMFFT approach to computing the one-dimensional DFT of a bandlimited and periodic sparse function $f(x)$ involves two key algorithmic steps: in the first step (Section III-A-III-B), the support $\mathcal{S}$ of the vector $\hat{f}$ is recovered, and in the second step (Section III-C), the nonzero values of $\hat{f}$ are computed using the knowledge of the support. For the noiseless case, pseudo-code is provided in Algorithms 1-4. For proofs of robustness and stability to noise, we refer to [13].

Algorithm 1 Output: $\hat{f}, \mathcal{S}$ = 1DSFFT(R, N, p)

1: Let μ, Δ and η be estimates for $\min_{j\in\mathcal{S}} |\hat{f}_j|, \frac{||\hat{f}||_\infty}{\mu}$ and the noise $\sqrt{N}\,\eta||\nu||_2$ respectively.
2: (In the noiseless case, let η be the desired level of accuracy)
3: $\mathcal{S} \leftarrow$ FIND_SUPPORT(R, N, p, μ, Δ)
4: $\hat{f} \leftarrow$ COMPUTE_VALUES($\mathcal{S}, R, N, p, \mu, \Delta, \eta$)
5: Output: $\hat{f}, \mathcal{S}$.

A. Finding the support

Using Figure 1 as a high-level example, the support-finding scheme, presented in Algorithm 2, makes use of three major ingredients: 1)sub-sampling, 2)shuffling, and 3)low-pass filtering. Sub-sampling reduces the size of the problem to a manageable level. For descriptive purposes, let $k, N, M_k \in \mathbb{N}$, $0 < \alpha < 1$ and $\mathcal{S}_k, \mathcal{W}_k, \mathcal{M}_k \subset \{0, 1, ..., N-1\}$. Then,

- *Aliased support* $\mathcal{S}_k$ at step k corresponds to indices of elements of the true support $\mathcal{S}$ modulo M_k.
- *Working support* $\mathcal{W}_k$ at step k corresponds to the set $\{0, 1, ..., M_k - 1\}$.
- *Candidate support* $\mathcal{M}_k$ at step k is any set satisfying $\mathcal{S}_k \subset \mathcal{M}_k \subset \mathcal{W}_k$ of size $\mathcal{O}(R)$.

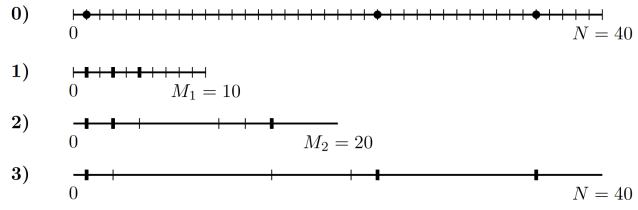

Fig. 1. Computing the support $\mathcal{S}$: In Line **0)**, we initialize elements of $\mathcal{S}$, corresponding to black dots in the grid $\{0, 1, ..., N-1\}$. In Line **1)**, elements of the candidate support $\mathcal{M}_1$ and aliased support $\mathcal{S}_1$ are identified, represented by thin and thick tickmarks, respectively. Note that $\mathcal{S}_1 \subset \mathcal{M}_1 \subset \{0, 1, ..., M_1 - 1\}$. In Line **2)**, elements of the candidate support $\mathcal{M}_2$, represented by thin tickmarks, are obtained through de-aliasing of $\mathcal{S}_1$. Aliased support $\mathcal{S}_2$ is represented by thick tickmarks. Note that $\mathcal{S}_2 \subset \mathcal{M}_2 \subset \{0, 1, ..., M_2 - 1\}$, and $M_2 = cM_1$. In Line **3)**, we finalize the process the working is equal to $\{0, 1, ..., N-1\}$ and $N = \alpha M_2$.

At step $k = 0$ only the fact that $\mathcal{S} \subset \{0, 1, ..., N-1\}$ is known (e.g., Line **0)** in Figure 1). At step $k = 1$, we let $M_1 = \frac{N}{\prod_{i=2}^{P} \rho_i} = \rho_1 K = \mathcal{O}(R\sqrt{\log(R)})$ and compute the samples,

$$f_{n_1;\, M_1} = \sum_{l=0}^{M_1-1} e^{-2\pi i\, \frac{n_1\, l}{M_1}} \left(\sum_{j:\, j \bmod M_1 = l} \hat{f}_j\right) = \sum_{l=0}^{M_1-1} e^{-2\pi i\, \frac{n_1\, l}{M_1}}\ \hat{f}_j^{(1)}$$

for $n_1 \in \mathcal{M}_1 := \{0, 1, ..., M_1 - 1\}$, the candidate support. The samples correspond to a DFT of size M_1 of the vector $\hat{f}^{(1)}$ with *aliased* entries of the original $\hat{f}$. These can be computed through the FFT in order $\mathcal{O}(M_1 \log(M_1)) = \mathcal{O}(R \log^{\frac{3}{2}}(R))$. The aliased support $\mathcal{S}_1$ is quickly identified from $\hat{f}^{(1)}$ since the latter is positive if and only if the corresponding aliased lattice contains an element of $\mathcal{S}$.

In Line **1)**, $\mathcal{S}_1$ and $\mathcal{M}_1$ are identified, and $\mathcal{W}_1 = \mathcal{M}_1$. Proceeding to step $k = 2$ in Line **2)**, from $\mathcal{S}_1$, $\mathcal{M}_2$ is constructed. Here however, the cost of recovering $\mathcal{S}_2$ in the previous way is higher since computing $\hat{f}^{(2)}$ involves performing an FFT of size M_2. Additional steps, involving a shuffling and filtering of the samples followed by an FFT of size $\mathcal{O}(R\sqrt{\log(R)})$, are required to bring the cost of identifying $\mathcal{S}_2$ from $\mathcal{M}_2$ down; these steps are described in Section III-B.

At any step k, $\mathcal{S}_k$ can be recovered from $\mathcal{M}_k$ using $\mathcal{O}(R\sqrt{\log(R)})$ samples and $\mathcal{O}(R \log^{\frac{3}{2}}(R))$ computations. Following the rapid recovery of $\mathcal{S}_2$, we iterate until $\mathcal{S}_k = \mathcal{S}$, i.e., $\mathcal{O}\left(\log\left(\frac{N}{R}\right)\right)$ times. This implies a cost of $\mathcal{O}\left(R \log\left(\frac{N}{R}\right)\right)$ measurements and $\mathcal{O}\left(R \log\left(\frac{N}{R}\right)\right)$ computational complexity (to first order) as fully proved in [13].

978-1-5090-3525-0/16/$31.00 ©2016 IEEE

Algorithm 2 Output: $\mathcal{S}_L$ = FIND_SUPPORT$(R, \tilde{N}, p, \mu, \Delta)$

Pick $2 \leq \rho \in \mathbb{R}$ and $0 < \delta \ll 1$.
Let $\alpha = \frac{1}{\rho}$, $K = \frac{\max\{8, \frac{2}{\alpha}\}}{\pi} R \sqrt{\log\left(\frac{2R\Delta}{\delta}\right) \log\left(\frac{2\Delta}{\delta}\right)}$ and choose $N = K \prod_{i=1}^{P} \rho_i \geq \tilde{N}$ where $2 \leq \rho_i \leq \rho \; \forall \; i$
Let $M_1 = K$ and $\mathcal{M}_1 := \{0, 1, ..., M_1 - 1\}$.
for k from 1 to P **do**
  $\mathcal{S}_k \leftarrow$ FIND_ALIASED_SUPPORT$(\mathcal{M}_k, M_k, K, \alpha, p, \delta, \mu, \Delta)$
  $\mathcal{M}_{k+1} := \cup_{m=0}^{\rho_k - 1} (\mathcal{S}_k + m M_k)$.
  $M_{k+1} := \rho_k M_k$
  $\alpha \leftarrow \frac{1}{\rho_i}$
end for
Output: $\mathcal{S}_P$.

B. Rapid recovery of $\mathcal{S}_k$ from knowledge of $\mathcal{M}_k$

Given a candidate support $\mathcal{M}_k$ at step k, Algorithm 3 allows for rapidly recovering the aliased support $\mathcal{S}_k$ in four major steps: 1)Permute samples randomly, 2) Apply a diagonal Gaussian filter, 3) Compute a small FFT and 4) Identify aliased support.

Algorithm 3 FIND_ALIASED_SUPPORT$(\mathcal{M}_k, M_k, K, \alpha, p, \delta, \mu, \Delta)$

Let $\sigma = \frac{\alpha \frac{M_k}{2R}}{\sqrt{\log(\frac{2R\Delta}{\delta})}}$ and $L = \log_\alpha(p)$.
Let $g_m = \frac{\sqrt{\pi}\sigma}{M_k} \sum_{h \in \mathbb{Z}} e^{-\pi^2 \sigma^2 \left(\frac{m + hM}{M}\right)^2}$, form = 0, 1, ..., $\mathrm{M}_k - 1$.
$\mathcal{S}_k \leftarrow \mathcal{M}_k$.
for l **from** 1 **to** L **do**
  Pick $Q \in \mathbb{Z}$ uniform randomly in $\{1 \leq Q \leq M_k - 1 : Q \perp M_k\}$.
  Compute:
  $\phi_j^{(k)} = \sum_{\substack{m=1 \\ |m \bmod M_k| \leq \frac{K}{2}}}^{M_k - 1} e^{2\pi i \frac{jm}{K}} \, g_m \, f_{(mQ) \bmod M_k;\, M_k}$,
  for $j = 0, 1, ..., K - 1$ rapidly through FFT.
  for $j \in \mathcal{M}_k$ **do**
    if $\left| \phi^{(k)}_{\left[(jQ \bmod M_k)\frac{K}{M_k}\right]} \frac{M_k}{K}(Q) \right| < \delta \, \mu$ **then**
      Remove j from $\mathcal{S}_k$.
    end if
  end for
end for
Output: $\mathcal{S}_k$.

To better understand each step, re-consider Figure 1 at step $k = 3$ with $M_k = 40$. First, randomly shuffle $\mathcal{M}_k$ within $\mathcal{W}_k$ by applying a permutation operator as described in [13]; for appropriately chosen random parameters, this shuffling is equivalent to uniform shuffling in Fourier space. The result is shown on Line **B)** of Figure 2. The shuffling step is followed by the application of a diagonal Gaussian filtering operator with an equivalent convolution in frequency space. The result is subsequently discretized to produce a small DFT of a permuted version the original Fourier coefficients convoluted with a Gaussian rapidly and accurately through a size-K FFT. The result is shown on line **C)** of Figure 2, and this information is sufficient to recover an aliased support from a candidate support. Details and proofs of correctness are presented in [13].

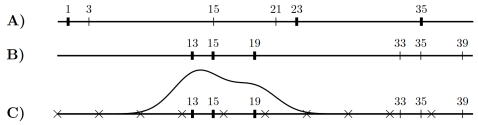

Fig. 2. Beginning with $\mathcal{S}_k = \{1, 23, 35\}$ $\mathcal{M}_k = \{1, 3, 15, 21, 23, 35\}$, and $\mathcal{W}_k = \{1, 2, ..., 39\}$, for finding the aliased support $\mathcal{S}_k$ from knowledge of $\mathcal{M}_k$ in Line **A)**, indices are shuffled in value space leading to a shuffling in frequency space in Line **B)** to obtain $\mathcal{S}_k^{\text{shuffled}} = \mathcal{S}_k \cdot 13 = \{13, 15, 19\}$ and $\mathcal{M}_k^{\text{shuffled}} = \mathcal{M}_k \cdot 13 = \{13, 15, 19, 33, 35, 39\}$. A Gaussian filter is applied, followed by a small FFT in Line **C)** on a grid G. The points of $\mathcal{M}_k$ for which the value of the result of the last step at their closest neighbor in G is small are discarded leaving only the aliased support $\mathcal{S}_k$.

The permutation and multiplications step both incur a cost of $\mathcal{O}(R)$, followed by an FFT of size $\mathcal{O}(K) = \mathcal{O}(R\sqrt{\log(R)})$ with $\mathcal{O}(R \log^{\frac{3}{2}}(R))$ cost. Finally, identifying the aliased support involves checking a simple property on each of the M_k elements of $\mathcal{M}_k$, incurring a cost $\mathcal{O}(R\sqrt{\log(R)})$. Repeated $\mathcal{O}(\log(p))$ times for a probability $(1 - p)$ of success implies that extracting $\mathcal{S}_k$ from $\mathcal{M}_k$ requires $\mathcal{O}(\log(p)\, R)$ samples with $\mathcal{O}(\log(p)\, R)$ complexity (to first order) as claimed.

C. Recovering values from knowledge of the support

Assume a set $\mathcal{S}$ of size $\mathcal{O}(R)$ containing the support has been recovered. We compute the values of the nonzero Fourier coefficients of $\hat{f}$ in Eq. (1) rapidly using this information. For this purpose, sample the $f(x)$ at $\left\{ \frac{n_t \bmod P^{(t)}}{P^{(t)}} \right\}_{n_t=0,t=1}^{P^{(t)}-1,T}$ (see Algorithm 4 for choice of $P^{(t)}$ and T). It can be shown that this leads to a system of the form,

$$[FB]\,\hat{f} = \begin{bmatrix} F^{(1)} & ... & 0 \\ ... & ... & ... \\ 0 & ... & F^{(T)} \end{bmatrix} \begin{bmatrix} B^{(1)} \\ ... \\ B^{(T)} \end{bmatrix} \hat{f} = f^{\dagger}. \quad (3)$$

where the first matrix is a T-block diagonal matrix having diagonal entries corresponding to DFT matrix of size $\mathcal{O}(R)$, and B is a sparse matrix with $\mathcal{O}(R)$ nonzero entries. In [13], we prove that FB as constructed is invertible in order $\mathcal{O}(R \log(N))$ and well-conditioned.

Algorithm 4 COMPUTE_VALUES$(\mathcal{S}, R, N, p, \mu, \Delta, \eta)$

Let $T = 4$, $Z = \lceil \log_{\frac{1}{2}}(\eta) \rceil$ and $L = \lceil \log_{\frac{1}{2}}(p) \rceil$.
for t **from** 1 **to** L **do**
  Pick $\{P^{(t)}\}_{t=1}^{T}$ i.i.d. uniform r.v. chosen among the set containing the smallest $4R \log_R(N)$ prime numbers greater than R.
  Sample $\left\{ f_{n \bmod P^{(t)};P^{(t)}} \right\}_{n=0}^{P^{(t)}-1}$, $t = 1, ..., T$
  Compute $\hat{f}_0 \leftarrow (FB)^* f_0$ (Eq.(3)).
  if $||(B^*B)\,\hat{f}_0||_2 < \frac{1}{2}$ **then**
    $\hat{f} \leftarrow \sum_{n=0}^{Z} (I - B^*B)^n \hat{f}_0$
    Output: $\hat{f}$.
    Exit.
  end if
end for

IV. The Multi-Dimensional Sparse FFT

By nature, whenever dealing with the multidimensional DFT/FFT, it must be assumed that the function of interest is both periodic and bandlimited with fundamental period $[0, 1)$ in each dimension, i.e., that it has an expansion of the form

$$f(x) = \sum_{j \in ([0,M) \cap \mathbb{Z})^d} e^{-2\pi i \, x \cdot j} \, \hat{f}_j \quad (4)$$

978-1-5090-3525-0/16/$31.00 ©2016 IEEE

for some finite $M \in \mathbb{N}$ and $j \in \mathbb{Z}^d$. Computing its Fourier coefficients is traditionally achieved through a tensor product of quadratures [14]::

$$\hat{f}_{(j_1,j_2,\ldots,j_d)} = \sum_{k=1}^{d} \sum_{n_k=0}^{M-1} \left(\prod_{k=1}^{d} e^{2\pi i \frac{j_k n_k}{M}} \right) f_{(n_1,n_2,\ldots,n_d)}.$$

In [13], however, we prove that if $f : [0,1)^d \to \mathbb{C}$ is bandlimited and periodic and has form (4), then

$$\int_{[0,1]^d} e^{-2\pi i\, j\cdot x}\, f(x)\,\mathrm{d}x = \frac{1}{N} \sum_{n=0}^{N-1} e^{-2\pi i\, j\cdot x_n}\, f(x_n)$$

for all $j \in \{0,1,\ldots,M-1\}$, where, $x_n = \frac{ng \bmod N}{N}$, $g = (1, M, M^2, \ldots, M^{d-1})$ and $N = M^d$. The summation can be written in two different ways due to periodicity, namely,

$$\frac{1}{N} \sum_{n=0}^{N-1} e^{-2\pi i\, k\cdot \frac{ng \bmod N}{N}}\, f\left(\frac{ng \bmod N}{N}\right),$$
$$\text{or} \quad \frac{1}{N} \sum_{n} e^{-2\pi i\, k\cdot \frac{ng}{N}}\, f\left(\frac{ng}{N}\right)$$

The former represents a quadrature rule with points $x_n = \frac{ng \bmod N}{N}$ distributed (more-or-less uniformly) in $[0,1)^d$ as depicted in Figure 3 (Left) while the latter represents an equivalent quadrature where the points $\tilde{x}_n = \frac{ng}{N}$ now lie on a *line* embedded in $\mathbb{R}^d$ as in Figure 3 (Right).

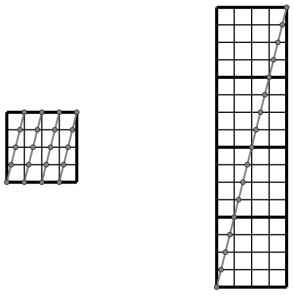

Fig. 3. Geometric interpretation of rank-1 lattice in 2D. The thick black box represents fundamental periodic domain. Grey dots represent rank-1 discretization points, thin lattice points represent quadrature point used with a tensor product, and the 2D grid represents standard/tensor product discretization points. Left: the rank-1 quadrature interpreted as a 2D discretization over the fundamental periodic region. Right: the rank-1 quadrature interpreted as a uniform discretization over a line in $\mathbb{R}^2$.

We can thus express any d-dimensional DFT as a one-dimensional DFT by choosing the appropriate sample points and re-ordering the Fourier coefficients through a simple isomorphism. Further details are available in [13].

V. Numerical Results

In this section, we present numerical results demonstrating the validity and performance of our sMFFT scheme. The results can be grouped in three categories: 1) a synthetic 2D example using a traditional "Phantom," with comparison against Compressed Sensing (CS) (ℓ_1-minimization), 2) a 3D synthetic example and 3) 2D real-data experiments in the context of change detection.

A. Comparison with 2D Compressive Sensing

Our first set of experiments involves 2D images of a Phantom at different resolution ranging from 16^2 pixels to 1024^2 pixels as shown in Figure 4 (left). Each image was obtained through 2D linear interpolation of a high-resolution image and has a sparsity level of $s = \frac{\#\text{nonzeros}}{\#\text{unknowns}} = \frac{R}{N} \approx 8\%$.

For each resolution level we proceed as follows: first, we computed the Fourier transform of the image at appropriately-chosen points (see [12] for CS sampling). We then added Gaussian noise to every sample in order to obtain an Signal-to-Noise Ratio (SNR) of $\mathrm{SNR} \approx 20\mathrm{dB}$. Finally, we proceed to the reconstruction of the image through the sMFFT, and CS by solving the following ℓ_1-minimization problem,

$$\begin{aligned} &\text{minimize} ||x||_1 \\ &\text{subject to} ||SFx - b||_2 \leq \epsilon, \end{aligned}$$

where F is the Discrete Fourier Transform (DFT) matrix, S is a selector matrix which selects only the rows corresponding to the location the random samples, and b is a vector containing the values of the Fourier transform of the image at random locations. All computations were performed on a small cluster possessing 4x Intel® Xeon® Processor (E7-4860 v2) and 256GB of RAM using MATLAB, and the optimization problem was solved using the `cvx` package [15], [16]. We measured both the time taken and the accuracy of each method in every case. The results are shown in Figure 4 and 5.

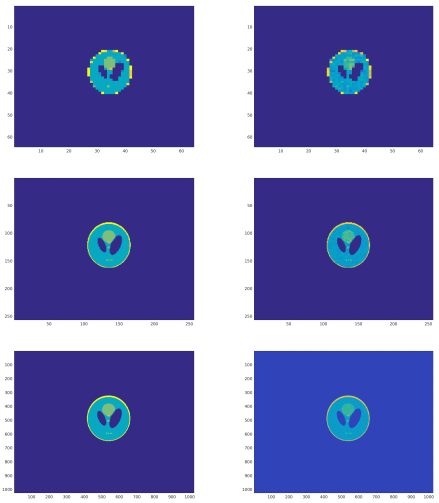

Fig. 4. 2D Phantom synthetic experiments. Left: original image. Right: reconstructed image (sMFFT). From top to bottom: 64^2 resolution, 256^2 resolution and 1024^2 resolution. Each reconstructions is in good agreement with the original and the relative error lies within the noise level.

We observe both techniques recover the original image with relative error on the order of noise as expected. However, the sMFFT is orders of magnitude faster than ℓ_1-minimization, which becomes more pronounced as the total number of unknowns (N) increases. Although CS requires a number of samples much smaller than traditional techniques based on the FFT, the *cost of reconstruction* remains prohibitively expensive

978-1-5090-3525-0/16/$31.00 ©2016 IEEE

due to two major issues [17]: 1) ℓ_1-solvers often exhibit sublinear convergence and 2) each iteration of an ℓ_1 solver requires at least one matrix-vector product at a cost of at least $\mathcal{O}(N)$. The dependency of the sMFFT on the total number of unknowns, however, is (provably) on the order of $\mathcal{O}(\log(N))$, which is significant especially in higher dimensions.

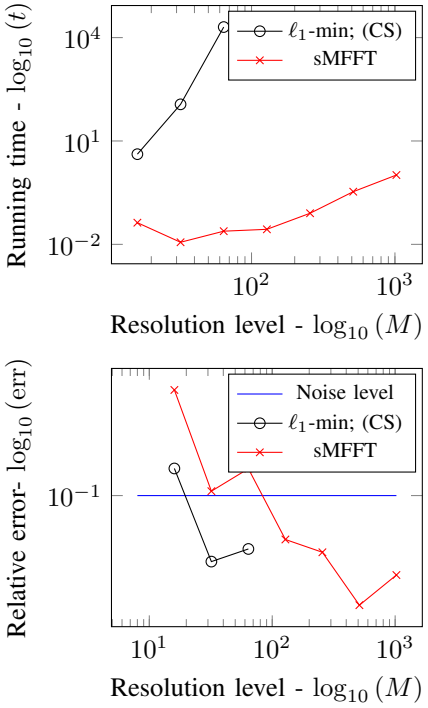

Fig. 5. 2D Phantom synthetic experiments. Comparison of performance between sMFFT and Compressed Sensing (ℓ_1-minimization). Top: reconstruction wall time for sMFFT (x) and CS (o) versus number of unknowns. Bottom: relative error of reconstruction for sMFFT (x) and CS (o) versus resolution (M) for a noise level of order 10^{-1}. Both techniques use roughly the same amount of measurements and recover the original image with a relative on the order of the noise level . However, sMFFT is orders of magnitude faster than ℓ_1-minimization.

B. 3D synthetic data

To demonstrate performance of our method in a difficult case, we chose a problem mimicking an example of radar and/or Computed Tomography (CT) imaging problem in $3D$ [11]. For this purpose, we generated an image from the extrusion of Reservoir Labs' logo (shown in Figure 6, left), embedded in a 3D region. The latter region is meshed using a uniform 1024^3 grid with $N = 1.07 \cdot 10^9$ unknowns. The scale of the extrusion is chosen such that the sparsity level is ≈ 0.002 with $R = 1.83 \cdot 10^4$ nonzero elements). We first computed $9.83 \cdot 10^6 \sim \mathcal{O}(R \log(N))$ samples at the appropriate locations dictated by the sMFFT, followed by the addition of Gaussian noise (SNR $= 5$dB), and a reconstruction through the sMFFT. The parameters used are summarized in Table II with results shown in Figure 6. The original and reconstruction are in good agreement qualitatively and quantitatively. Furthermore, the scheme requires very little time to reconstruct the solution due to the sMFFT's advantageous scaling.

TABLE II
PARAMETERS USED FOR THE 3D SYNTHETIC EXPERIMENT

# unknowns/pixels (N)	$1.07 \cdot 10^9$
# nonzeros (R)	$1.83 \cdot 10^4$
# samples	$9.83 \cdot 10^6$
Reconstruction time (s)	1.05
Relative error	$1.04 \cdot 10^{-1}$
SNR (dB)	5.23

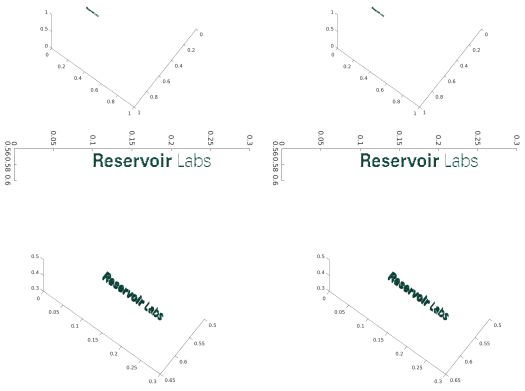

Fig. 6. Left: Original image. Right: Reconstructed image (sMFFT). From top to bottom: full-scale image, front view (zoomed) and side view (zoomed). Signal-to-Noise ratio: SNR $= 5$dB, total number of unknowns $N = 1.0737 \cdot 10^9$, total number of nonzeros: $R = 1.8346 \cdot 10^4$, number of samples: $9.8304 \cdot 10^6$, reconstruction time: 1.0511s.

C. Change detection on 2D real data

For the third numerical experiment, the sMFFT is applied to Change Detection (CD) problems. Figures 7 and 8 show results from two different experiments. All images were obtain from the CDW-2014 dataset [18]. In each case, two 512^2 pixeled greyscale images of the same scene are considered. The first scene contains the background, whereas the second scene contains the addition of a sparse perturbation, i.e., a person walking in front of the camera. The (forward) sMFFT of each image is computed and the two are subtracted. Since all operations are linear, this is equivalent to the computation of the sMFFT of the change itself. Finally, the (inverse) sMFFT of the difference is computed rapidly to obtain the change. Relevant parameters are summarized in Table III, and results are shown in Figures 7 and 8.

TABLE III
PARAMETERS USED IN CD EXPERIMENTS

Case	Type	# pixels (N)	# nonzeros (R)	time (s)
1	Optical	$2.62144 \cdot 10^5$	$1.0894 \cdot 10^4$	0.5292
2	Infrared (IR)	$2.62144 \cdot 10^5$	$1.886 \cdot 10^4$	1.1236

978-1-5090-3525-0/16/$31.00 ©2016 IEEE

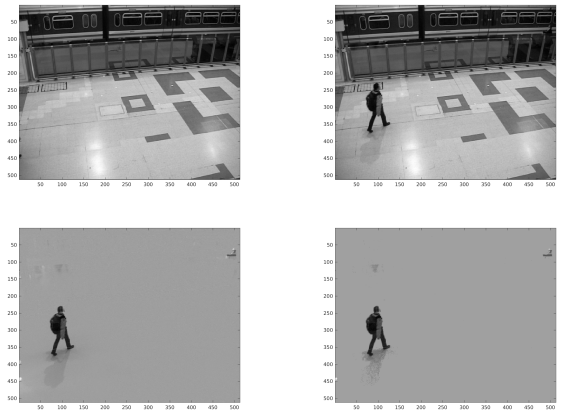

Fig. 7. Case 1. Optical wavelength change detection experiment. Top left: Original background image. Top right: Original background image with perturbation (change). Bottom left: difference between original images. Bottom right: reconstruction of change through sMFFT. Each original images contain a total of $512 \times 512 = 2.62144 \cdot 10^5$ pixels. The perturbation occupies $1.09 \cdot 10^4$ pixels. The reconstruction time is: 0.53s.

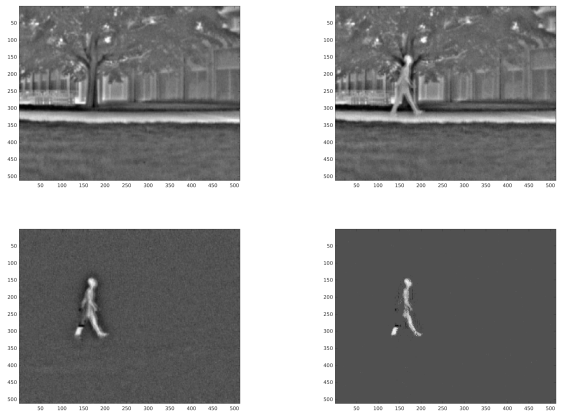

Fig. 8. Case 2. IR wavelength change detection experiment. Top left: Original background image. Top right: Original background image with perturbation (change). Bottom left: difference between original images. Bottom right: reconstruction of change through sMFFT. Each original images contain a total of $512 \times 512 = 2.62 \cdot 10^5$ pixels. The perturbation occupies $1.89 \cdot 10^4$ pixels. The reconstruction time is: 1.12s. The change and its reconstruction are in good agreement despite noise.

Qualitatively, the sMFFT produces very good results with low computational cost compared to existing techniques based on CS (e.g. [19]). These results suggest that the sMFFT possesses great potential for sparse image processing, and in particular, large-scale change detection. This can be accomplished through the use of a one-pixel or a "few-pixels" camera for instance. Finally, although this example was conducted in 2D, the sMFFT linear scaling in dimension would also allow the treatment of the temporal case (video change detection) as well as the coherent case (Coherent Change Detection; CCD).

VI. Conclusion

We have introduced a highly-efficient and stable multi-dimensional Fourier transform for real and positive sparse vectors. We have shown performance through various synthetic and real-data examples. The characteristics of the sMFFT are such that previously intractable problems can enter into the realm of the practical.

We are currently extending our algorithm to general (fully complex) sparse vectors. We also plan to treat different types of transforms such as wavelets and curvelets with the expectation of reducing the computational burden of various machine learning and image recognition algorithms [20].

References

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of Computation*, vol. 19, no. 90, pp. 297–301, 1965.

[2] T. Darden, D. York, and L. Pedersen, "Particle mesh ewald: An $nlog(n)$ method for ewald sums in large systems," *The Journal of Chemical Physics*, vol. 98, no. 12, pp. 10 089–10 092, 1993.

[3] S. Mallat, *A wavelet tour of signal processing*. Academic Press, 1999.

[4] M. Frigo and S. G. Johnson, "FFTW: An adaptive software architecture for the FFT," in *Acoustics, Speech and Signal Processing, 1998. Proc. of the 1998 IEEE Intl. Conf. on*, vol. 3. IEEE, 1998, pp. 1381–1384.

[5] P. Indyk and M. Kapralov, "Sample-optimal Fourier sampling in any constant dimension," in *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*. IEEE, 2014, pp. 514–523.

[6] G. Plonka and K. Wannenwetsch, "A sparse fast Fourier algorithm for real nonnegative vectors," *arXiv preprint arXiv:1602.05444*, 2016.

[7] ——, "Deterministic sparse FFT algorithms," *PAMM*, vol. 15, no. 1, pp. 667–668, 2015.

[8] B. Ghazi, H. Hassanieh, P. Indyk, D. Katabi, E. Price, and L. 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.

[9] A. C. Gilbert, S. Muthukrishnan, and M. Strauss, "Improved time bounds for near-optimal sparse Fourier representations," in *Optics & Photonics 2005*. International Society for Optics and Photonics, 2005, pp. 59 141A–59 141A.

[10] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Nearly optimal sparse Fourier Transform," in *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*. ACM, 2012, pp. 563–578.

[11] J. M. Lopez-Sanchez and J. Fortuny-Guasch, "3-D radar imaging using range migration techniques," *Antennas and Propagation, IEEE Transactions on*, vol. 48, no. 5, pp. 728–737, 2000.

[12] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[13] P. Letourneau, M. Langston, B. Meister, and R. Lethin, "A sparse multidimensional FFT for real positive vectors," *ArXiv e-prints: arXiv:1604.06682*, Apr. 2016.

[14] C. Van Loan, *Computational frameworks for the Fast Fourier Transform*, 1992, vol. 10.

[15] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[16] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110.

[17] A. Y. Yang, Z. Zhou, A. G. Balasubramanian, S. S. Sastry, and Y. Ma, "Fast-minimization algorithms for robust face recognition," *Image Processing, IEEE Transactions on*, vol. 22, no. 8, pp. 3234–3246, 2013.

[18] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection. net: A new change detection benchmark dataset," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. IEEE, 2012, pp. 1–8.

[19] G. K. Atia, "Change detection with compressive measurements," *Signal Processing Letters, IEEE*, vol. 22, no. 2, pp. 182–186, 2015.

[20] Y.-l. Tian, T. Kanade, and J. F. Cohn, "Evaluation of gabor-wavelet-based facial action unit recognition in image sequences of increasing complexity," in *Automatic Face and Gesture Recognition, 2002. Proc. 5th IEEE Inter. Conf. on*. IEEE, 2002, pp. 229–234.

978-1-5090-3525-0/16/$31.00 ©2016 IEEE