GESPAR: Efficient Phase Retrieval of Sparse Signals

Yoav Shechtman, Amir Beck, and Yonina C. Eldar, *Fellow, IEEE*

***Abstract*—We consider the problem of phase retrieval, namely, recovery of a signal from the magnitude of its Fourier transform, or of any other linear transform. Due to the loss of Fourier phase information, this problem is ill-posed. Therefore, prior information on the signal is needed in order to enable its recovery. In this work we consider the case in which the signal is known to be sparse, i.e., it consists of a small number of nonzero elements in an appropriate basis. We propose a fast local search method for recovering a sparse signal from measurements of its Fourier transform (or other linear transform) magnitude which we refer to as GESPAR: GrEedy Sparse PhAse Retrieval. Our algorithm does not require matrix lifting, unlike previous approaches, and therefore is potentially suitable for large scale problems such as images. Simulation results indicate that GESPAR is fast and more accurate than existing techniques in a variety of settings.**

***Index Terms*— Non-convex optimization, phase retrieval, sparse signal processing.**

I. Introduction

RECOVERY of a signal from the magnitude of its Fourier transform, also known as phase retrieval, is of great interest in applications such as optical imaging [1], crystallography [2], and more [3]. Due to the loss of Fourier phase information, the problem (in 1D) is generally ill-posed. A common approach to overcome this ill-posedeness is to exploit prior information on the signal. A variety of methods have been developed that use such prior information, which may be the signal's support (region in which the signal is nonzero), non-negativity, or the signal's magnitude [4], [5].

A popular class of algorithms is based on the use of alternate projections between the different constraints. In order to increase the probability of correct recovery, these methods require the prior information to be very precise, for example, exact/or "almost" exact knowledge of the support set. Since the projections are generally onto non-convex sets, convergence to a correct recovery is not guaranteed [6]. A more recent approach is to use matrix-lifting of the problem which allows to recast phase retrieval as a semi-definite programming (SDP) problem [7]. The algorithm developed in [7] does not require prior information about the signal but instead uses multiple signal measurements (e.g., using different illumination settings, in an optical setup).

In order to obtain more robust recovery without requiring multiple measurements, we develop a method that exploits signal sparsity. Existing approaches aimed at recovering sparse signals from their Fourier magnitude belong to two main categories: SDP-based techniques [8]–[11] and algorithms that use alternate projections (Fienup-type methods) [12]. Phase retrieval of sparse signals can be viewed as a special case of the more general quadratic compressed sensing (QCS) problem considered in [8]. Specifically, QCS treats recovery of sparse vectors from quadratic measurements of the form $y_i = \mathbf{x}^T\mathbf{A}_i\mathbf{x}$, $i = 1, \ldots, N$, where $\mathbf{x}$ is the unknown sparse vector to be recovered, y_i are the measurements, and $\mathbf{A}_i$ are known matrices. In (discrete) phase retrieval, $\mathbf{A}_i = \mathbf{F}_i^*\mathbf{F}_i$ where $\mathbf{F}_i$ is the ith row of the discrete Fourier transform (DFT) matrix. QCS is encountered, for example, when imaging a sparse object using partially spatially-incoherent illumination [8].

A general approach to QCS was developed in [8] based on matrix lifting. More specifically, the quadratic constraints where lifted to a higher dimension by defining a matrix variable $\mathbf{X} = \mathbf{x}\mathbf{x}^T$. The problem was then recast as an SDP involving minimization of the rank of the lifted matrix subject to the recovery constraints as well as row sparsity constraints on $\mathbf{X}$. An iterative thresholding algorithm based on a sequence of SDPs was then proposed to recover a sparse solution. Similar SDP-type ideas were recently used in the context of phase retrieval [9], [10]. However, due to the increase in dimension created by the matrix lifting procedure, the SDP approach is not suitable for large-scale problems.

Another approach for phase retrieval of sparse signals is adding a sparsity constraint to the well-known iterative error reduction algorithm of Fienup [12]. In general, Fienup-type approaches are known to suffer from convergence issues and often do not lead to correct recovery especially in 1D problems; simulation results show that even with the additional information that the input is sparse, convergence is still problematic and the algorithm often recovers erroneous solutions.

In this paper we propose an efficient method for phase retrieval which also leads to good recovery performance. Our approach is based on a fast 2-opt local search method (see [13] for an excellent introduction to such techniques) applied to a sparsity constrained non-linear optimization formulation of the problem. We refer to the resulting algorithm as GESPAR: GrEedy Sparse PhAse Retrieval. Sparsity constrained nonlinear optimization problems have been considered recently in [14]; the method derived in this paper is motivated—although different in many aspects—by the local search-type techniques of

Manuscript received September 25, 2013; accepted December 19, 2013. Date of publication January 02, 2014; date of current version January 20, 2014. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yimin D. Zhang. The work of A. Beck was supported in part by the Israel Science Foundation under Grant 253/12. The work of Y. Eldar was supported in part by the Israel Science Foundation under Grant 170/10, in part by the Ollendorf Foundation, in part by the SRC, and in part by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI).

Y. Shechtman was with the Department of Physics, Technion-Israel Institute of Technology, Haifa 32000, Israel. He is now with the Department of Chemistry, Stanford University, Stanford, CA 94305-5080 USA (e-mail: yoavsh@stanford.edu).

A. Beck is with the Department of Industrial Engineering, Technion-Israel Institute of Technology, Haifa 32000, Israel (e-mail: becka@ie.technion.ac.il).

Y. C. Eldar is with the Department of Electrical Engineering, Technion-Israel Institute of Technology, Haifa 32000, Israel (e-mail: yonina@ee.technion.ac.il).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2013.2297687

1053-587X © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

[14]. In essence, GESPAR is a local-search method, where the support of the sought signal is updated iteratively, according to selection rules described in detail in Section III. A local minimum of the objective function is then found given the current support using the damped Gauss Newton algorithm. Theorem 1 establishes convergence of the iterations to a stationary point of the objective under suitable conditions.

We demonstrate through numerical simulations that GESPAR is both efficient and more accurate than current techniques. Several other aspects of the algorithm are explored via simulations such as robustness to noise, and scalability for larger dimensions. In the simulations performed we found that the number of measurements needed for reliable recovery from Fourier magnitudes seems to scale like s^3, where s is the sparsity level.

GESPAR is applicable to recovery of a sparse vector from general quadratic measurements, and is not restricted to Fourier magnitude measurements. Nonetheless, when the measurements are obtained in the Fourier domain, the algorithm can be implemented efficiently by exploiting the fast Fourier transform, as we discuss in Section IV.

The remainder of the paper is organized as follows. We formulate the problem in Section II. Section III describes our proposed algorithm in detail and establishes convergence of the local iterations. Implementation details for Fourier-based problems are provided in Section IV. Extensive numerical experiments illustrating the empirical performance of GESPAR are presented in Section V.

II. Problem Formulation

A. Sparse Phase Retrieval: Fourier Measurements

We are given a vector of measurements $\mathbf{y} \in \mathbb{R}^N$, that corresponds to the magnitude-squared of an N point DFT of a vector $\mathbf{x} \in \mathbb{R}^N$, i.e.:

$$y_l = \left| \sum_{m=1}^{n} x_m e^{-\frac{2\pi j(m-1)(l-1)}{N}} \right|^2, \quad l = 1, \ldots, N. \tag{1}$$

Here $\mathbf{x}$ is constructed by $(N-n)$ zero padding of a vector $\bar{\mathbf{x}} \in \mathbb{R}^n$ with elements x_i, $i = 1, 2, \ldots, n$. Denoting by $\mathbf{F} \in \mathbb{C}^{N \times N}$ the DFT matrix with elements $\exp\left\{-\frac{2\pi j(m-1)(l-1)}{N}\right\}$, we can express $\mathbf{y}$ as $\mathbf{y} = |\mathbf{F}\mathbf{x}|^2$, where $|\cdot|^2$ denotes the element-wise absolute-squared value. The vector $\bar{\mathbf{x}}$ is known to be s-sparse, that is, it contains at most s nonzero elements. Our goal is to recover $\bar{\mathbf{x}}$, or $\mathbf{x}$, given the measurements $\mathbf{y}$ and the sparsity level s.

The mathematical formulation of the problem that we consider consists of minimizing the sum of squared errors subject to the sparsity constraint:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \sum_{i=1}^{N} \left(|\mathbf{F}_i \mathbf{x}|^2 - y_i \right)^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_0 \le s, \\ & \operatorname{supp}(\mathbf{x}) \subseteq \{1, 2, \ldots, n\}, \\ & \mathbf{x} \in \mathbb{R}^N, \end{aligned} \tag{2}$$

where $\mathbf{F}_i$ is the ith row of the DFT matrix $\mathbf{F}$, and $\|\cdot\|_0$ stands for the zero-"norm", that is, the number of nonzero elements. Note that the unknown vector $\mathbf{x}$ can only be found up to trivial degeneracies that are the result of the loss of Fourier phase information: circular shift, global phase, and signal "mirroring".

Support Information: To aid in solving the phase retrieval problem, we can rely on the fact that the autocorrelation sequence of $\bar{\mathbf{x}}$ (the first n components of $\mathbf{x}$) may be determined from $\mathbf{y}$ if $N \ge 2n-1$. Specifically, let

$$g_m = \sum_{i=1}^{n} x_i x_{i+m}, \quad m = -(n-1), \ldots, n-1 \tag{3}$$

denote the correlation sequence of length $2n-1$. If we choose $N \ge 2n-1$, then $\{g_m\}$ can be obtained by taking the inverse DFT of $\mathbf{y}$.

Determining g_m requires oversampling, or zero-padding of $\mathbf{x}$. While this additional information improves the recovery performance, as demonstrated in the simulation section, it is not actually needed for GESPAR to work. Nevertheless, when this information is available, GESPAR exploits it, in the following way. First, we assume that no support cancelations occur in $\{g_m\}$, namely, if $x_i \neq 0$ and $x_j \neq 0$ for some i, j, then $g_{|i-j|} \neq 0$. When the values of $\mathbf{x}$ are random, this is true with probability 1. This fact can be used in GESPAR in order to obtain initial information on the support of $\mathbf{x}$, which we capture by two sets J_1 and J_2.

Denote by J_1 the set of indices known in advance to be in the support. To derive the set J_1, note that due to the existing degree of freedom relating to shift-invariance of $\mathbf{x}$, the index 1 can be assumed to be in the support, thereby removing this degree of freedom. Consequently, the index corresponding to the last nonzero element in the autocorrelation sequence is also in the support, i.e.

$$i_{max} = 1 + \arg\max_i \{i : g_i \neq 0\}.$$

Therefore, $J_1 = \{1, i_{max}\}$.

Next, we denote by J_2 the set of indices that are candidates for being in the support, meaning the indices that are *not* known in advance to be in the off-support (the complement of the support). Specifically, J_2 contains the set of all indices $k \in \{1, 2, \ldots, n\}$ such that $g_{k-1} \neq 0$. Obviously, since we assume that $x_k = 0$ for $k > n$, we have $J_2 \subseteq \{1, 2, \ldots, n\}$. As a concrete example, consider the signal $\bar{\mathbf{x}} = (2, 0, 0, -1, 0, -1.5)^T$. The corresponding 11 point autocorrelation function g_m is given by $g_m = (-3, 0, -2, 1.5, 0, 7.25, 0, 1.5, -2, 0, -3)^T$. The set J_1 is therefore $J_1 = \{1, 6\}$. Next, by examining the zeros of g_m, and using our assumption of no support-cancelations, we deduce that there are no two non-zero elements $x_i \neq 0$ and $x_j \neq 0$ such that $|i-j| = 1, 4$. Therefore, forcing the first element in $\mathbf{x}$ to be non-zero, which removes the shift-invariance degeneracy, immediately implies that $x_2 = x_5 = 0$. In this way J_2 is determined as $J_2 = \{1, 3, 4, 6\}$. Defining $\mathbf{A}_i = \Re(\mathbf{F}_i)^T \Re(\mathbf{F}_i) + \Im(\mathbf{F}_i)^T \Im(\mathbf{F}_i) \in \mathbb{R}^{N \times N}$, problem (2) along with the support information can be written as

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \equiv \sum_{i=1}^{N} \left(\mathbf{x}^T \mathbf{A}_i \mathbf{x} - y_i \right)^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_0 \le s, \\ & J_1 \subseteq \operatorname{supp}(\mathbf{x}) \subseteq J_2, \\ & \mathbf{x} \in \mathbb{R}^N, \end{aligned} \tag{4}$$

which will be the formulation to be studied.

Note that even with knowledge of the exact support of $\mathbf{x}$ there is no guarantee for uniqueness beyond the aforementioned trivial degeneracies. Consider for example the two vectors $\mathbf{u} = (1, 0, -2, 0, -2)$ and $\mathbf{v} = (1 - \sqrt{3}, 0, 1, 0, 1 + \sqrt{3})$. Both of these vectors are $s = 3$ sparse, and they have the same autocorrelation function $g_m = (-2, 0, 2, 0, 9, 0, 2, 0, -2)$. This ambiguity therefore cannot be resolved using any method that uses sparsity (even exact support information) and autocorrelation (or Fourier magnitude) measurements alone.

Finally, when the measurements are noisy, the autocorrelation information is not very useful for support estimation, since very small (noise level) values in the autocorrelation sequence cannot be treated as zero. For this reason, the autocorrelation-derived support information is not used in GESPAR at all in the noisy case. Formally, ignoring this information is equivalent to setting $J_1 = \{1\}$ and $J_2 = \{1, 2, \ldots, n\}$.

B. Sparse Phase Retrieval: General Measurements

Although the problem formulation above assumes Fourier measurements and sparsity of $\bar{\mathbf{x}}$, we show below that our approach applies to arbitrary quadratic measurements of $\bar{\mathbf{x}}$. This includes the case in which $\bar{\mathbf{x}}$ is sparse in a basis other than the identity basis. In fact, in this general case, the formulation given in (4) remains the same, with the only change being the definition of the matrices $\mathbf{A}_i$.

Consider the phase retrieval problem with respect to arbitrary linear measurements, so that

$$\mathbf{y}_i = \left|\langle \boldsymbol{\phi}_i \mathbf{x} \rangle\right|^2, \tag{5}$$

for a set of measurement vectors $\boldsymbol{\phi}_i \in \mathbb{R}^n$, $i = 1, \ldots, N$. The corresponding phase retrieval problem can be written as in (4) with $\mathbf{A}_i = \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T$. Similarly, suppose that $\bar{\mathbf{x}} = \mathbf{D}\mathbf{z}$, where $\mathbf{D} \in \mathbb{R}^{n \times b}$ is some basis in which $\bar{\mathbf{x}}$ is sparse, and $\mathbf{z} \in \mathbb{R}^b$ is a sparse vector. In this case $\mathbf{A}_i = \mathbf{D}^T \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T \mathbf{D}$. Thus, our formulation can accommodate arbitrary sparsity bases and general quadratic measurements.

In the next section, we propose GESPAR—an iterative local-search based algorithm for solving (4). We note that although in the context of phase retrieval the parameters $\mathbf{A}_i, J_1, J_2$ have special properties (e.g., $\mathbf{A}_i$ is positive semidefinite of at most rank 2, $|J_1| = 2$), we will not use these properties in GESPAR. Therefore, our approach is capable of handling general instances of (4) with the sole assumption that $\mathbf{A}_i$ is symmetric for any $i = 1, 2, \ldots, N$. In the Fourier case, the algorithm can be implemented more efficiently, as we discuss in Section IV.

III. GREEDY SPARSE PHASE RETRIEVAL (GESPAR)

A. The Damped Gauss-Newton Method

Before describing our algorithm, we begin by presenting a variant of the damped Gauss-Newton (DGN) method [15], [16] that is in fact the core step of our approach. The DGN method is invoked in order to solve the problem of minimizing the objective function f over a *given* support $S \subseteq \{1, 2, \ldots, n\}$ ($|S| = s$):

$$\min \left\{ f(\mathbf{U}_S \mathbf{z}) : \mathbf{z} \in \mathbb{R}^s \right\}, \tag{6}$$

where $\mathbf{U}_S \in \mathbb{R}^{N \times s}$ is the matrix consisting of the columns of the identity matrix $\mathbf{I}_N$ corresponding to the index set S. With this notation, (6) can be explicitly written as

$$\min \left\{ g(\mathbf{z}) \equiv \sum_{i=1}^{N} \left(\mathbf{z}^T \mathbf{U}_S^T \mathbf{A}_i \mathbf{U}_S \mathbf{z} - y_i \right)^2 : \mathbf{z} \in \mathbb{R}^s \right\}. \tag{7}$$

The minimization in (7) is a nonlinear least-squares problem. A natural approach for tackling it is via the DGN method. This algorithm begins with an arbitrary vector $\mathbf{z}_0$. In our simulations, we choose it as a white random Gaussian vector with zero mean and unit variance. At each iteration, all the terms inside the squares in $g(\mathbf{z})$ are linearized around the previous guess. Namely, we write $g(\mathbf{z})$ from (7) as:

$$g(\mathbf{z}) = \sum_{i=1}^{N} h_i^2(\mathbf{z}), \tag{8}$$

with $h_i(\mathbf{z}) = \mathbf{z}^T \mathbf{B}_i \mathbf{z} - y_i$, and $\mathbf{B}_i = \mathbf{U}_S^T \mathbf{A}_i \mathbf{U}_S$. At each step we replace h_i by its linear approximation around $\mathbf{z}_{k-1}$:

$$\begin{aligned} h_i &\approx h_i(\mathbf{z}_{k-1}) + \nabla h_i(\mathbf{z}_{k-1})^T (\mathbf{z} - \mathbf{z}_{k-1}) \\ &= \mathbf{z}_{k-1}^T \mathbf{B}_i \mathbf{z}_{k-1} - y_i + 2(\mathbf{B}_i \mathbf{z}_{k-1})^T (\mathbf{z} - \mathbf{z}_{k-1}). \end{aligned} \tag{9}$$

We then choose $\mathbf{z}_k$ to be the solution of the problem

$$\min_{\mathbf{z}} \sum_{i=1}^{N} \left(\mathbf{z}_{k-1}^T \mathbf{B}_i \mathbf{z}_{k-1} - y_i + 2(\mathbf{B}_i \mathbf{z}_{k-1})^T (\mathbf{z} - \mathbf{z}_{k-1}) \right)^2. \tag{10}$$

Problem (10) can be written as a linear least-squares problem

$$\tilde{\mathbf{z}}_k = \arg\min \left\| J(\mathbf{z}_{k-1}) \mathbf{z} - \mathbf{b}_k \right\|_2^2 \tag{11}$$

with the ith row of $J(\mathbf{z}_{k-1})$ being $\nabla h_i(\mathbf{z}_{k-1})^T = 2(\mathbf{B}_i \mathbf{z}_{k-1})^T$, and the ith component of $\mathbf{b}_k$ given by $y_i + \mathbf{z}_{k-1}^T \mathbf{B}_i \mathbf{z}_{k-1}$ for $i = 1, 2, \ldots, N$. The solution $\tilde{\mathbf{z}}_k$ is equal to $\tilde{\mathbf{z}}_k = \left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1}) \right)^{-1} J(\mathbf{z}_{k-1})^T \mathbf{b}_k$. We then define a direction vector $\mathbf{d}_k = \mathbf{z}_{k-1} - \tilde{\mathbf{z}}_k$. This direction is used to update the solution with an appropriate stepsize designed to guarantee the convergence of the method to a stationary point of $g(\mathbf{z})$. The stepsize is chosen via a simple backtracking procedure. Algorithm 1 describes the DGN method in detail. In our implementation the stopping parameters were chosen as $\varepsilon = 10^{-4}$ and $L = 100$.

The following theorem establishes the rate of convergence of the norm of the gradient of the objective function to zero, and consequently proves that the limit points of the sequence are stationary points.

Theorem 1: Let $\{\mathbf{z}_k\}$ be the sequence generated by the DGN method. Assume that $\sum_{i=1}^{N} \mathbf{B}_i \succ \mathbf{0}$ and that there exists $\underline{\lambda} > 0$ such that for all k

$$\lambda_{\min} \left(J(\mathbf{z}_k)^T J(\mathbf{z}_k) \right) \geq \underline{\lambda}.$$

Then $\nabla g(\mathbf{z}_k) \to \mathbf{0}$ as $k \to \infty$ and there exists a constant $C > 0$ such that

$$\min_{p=1,\ldots,k} \left\| \nabla g(\mathbf{z}_p) \right\| \leq \frac{\sqrt{g(\mathbf{z}_0)}}{C\sqrt{k+1}}. \tag{12}$$

Moreover, each limit point of the sequence is a stationary point of g.

Proof: See Appendix A. □

Note that the proof requires $J(\mathbf{z}_k)$ to have full column rank, and in fact that the minimum eigenvalues of $J(\mathbf{z}_k)^T J(\mathbf{z}_k)$ are uniformly bounded below. In the vast majority of our runs this assumption held true; however, we did encounter in our numerical experiments a few cases in which this condition was not valid. In these situations, our implementation chose one of the optimal solutions of the corresponding least-squares problem. We noticed that these cases had negligible effect on the results.

Algorithm 1 DGN for solving (7)

Input: $(\mathbf{A}_i, y_i, S, \varepsilon, L)$.

$\mathbf{A}_i \in \mathbb{R}^{N\times N}$, $i = 1, 2, \ldots, N$—symmetric matrices.

$y_i \in \mathbb{R}$, $i = 1, 2, \ldots, N$.

$S \subseteq \{1, 2, \ldots, n\}$—index set.

ε—stopping criteria parameter.

L—maximum allowed iterations.

Output: $\mathbf{z}$—an optimal (or suboptimal) solution of (7).

Initialization: Set $\mathbf{B}_i = \mathbf{U}_S^T \mathbf{A}_i \mathbf{U}_S$, $t_0 = 0.5$, $\mathbf{z}_0$ a random vector.

General Step $k (k \geq 1)$: Given the iterate $\mathbf{z}_{k-1}$, the next iterate is determined as follows:

1. **Gauss-Newton Direction**: Let $\tilde{\mathbf{z}}_k$ be the solution of the linear least-squares problem (11), given by:

$$\tilde{\mathbf{z}}_k = \left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1})\right)^{-1} J(\mathbf{z}_{k-1})^T \mathbf{b}_k$$

with the ith row of $J(\mathbf{z}_{k-1})$ being $2(\mathbf{B}_i \mathbf{z}_{k-1})^T$, and the ith component of $\mathbf{b}_k$ given by $y_i + \mathbf{z}_{k-1}^T \mathbf{B}_i \mathbf{z}_{k-1}$. The Gauss-Newton direction is

$$\mathbf{d}_k = \mathbf{z}_{k-1} - \tilde{\mathbf{z}}_k.$$

2. **Stepsize Selection via Backtracking**: set $u = \min\{2t_{k-1}, 1\}$. Choose a stepsize t_k as $t_k = \left(\frac{1}{2}\right)^m u$, where m is the minimal nonnegative integer for which

$$g\left(\mathbf{z}_{k-1} - \left(\frac{1}{2}\right)^m u \mathbf{d}_k\right) < g(\mathbf{z}_{k-1}) - u\left(\frac{1}{2}\right)^{m+1} \nabla g(\mathbf{z}_{k-1})^T \mathbf{d}_k,$$

with $g(\mathbf{z})$ given by (7).

3. **Update**: set $\mathbf{z}_k = \mathbf{z}_{k-1} - t_k \mathbf{d}_k$.
4. **Stopping rule**: STOP if either $\|\mathbf{z}_k - \mathbf{z}_{k-1}\| < \varepsilon$ or $k > L$.

B. The 2-opt Local Search Method

The GESPAR method consists of repeatedly invoking a local-search method on an initial random support set. In this section we describe the local search procedure. At the beginning, the support is chosen to be a set of s random indices chosen to satisfy the support constraints $J_1 \subseteq S \subseteq J_2$. Then, at each iteration a swap between a support and an off-support index is performed such that the resulting solution via the DGN method improves the objective function. Since at each iteration only two elements are changed (one in the support and one in the off-support), this is a so-called "2-opt" method (see [13]). The swaps are always chosen to be between the index corresponding to components in the current iterate $\mathbf{x}_{k-1}$ with the smallest absolute value and the off-support index corresponding to the component of $\nabla f(\mathbf{x}_{k-1}) = 4\sum_i (\mathbf{x}_{k-1}^T \mathbf{A}_i \mathbf{x}_{k-1} - \mathbf{c}_i)\mathbf{A}_i \mathbf{x}_{k-1}$ with the largest absolute value. This process continues as long as the objective function decreases and stops when no improvement can be made. A detailed description of the method is given in Algorithm 2.

Algorithm 2 2-opt

Input: $(\mathbf{A}_i, y_i)$.

$\mathbf{A}_i \in \mathbb{R}^{N\times N}$, $i = 1, 2, \ldots, N$—symmetric matrices.

$y_i \in \mathbb{R}$, $i = 1, 2, \ldots, N$.

Output: $\mathbf{x}$—a suggested solution for problem (4).

T—total number of required swaps.

1) **Initialization**:
   a) Set $T = 0$.
   b) Generate a random index set S_0 ($|S_0| = s$) satisfying the support constraints ($J_1 \subseteq S_0 \subseteq J_2$).
   c) Invoke the DGN method with parameters $(\mathbf{A}_i, y_i, S_0, 10^{-4}, 100)$ and obtain an output $\mathbf{z}_0$. Set $\mathbf{x}_0 = \mathbf{U}_{S_0} \mathbf{z}_0$.
2) **General Step** $(k = 1, 2, \ldots)$:
   a) Let i be the index from $S_{k-1} \setminus J_1$ corresponding to the component of $\mathbf{x}_{k-1}$ with the smallest absolute value. Let j be the index from $S_{k-1}^c \cap J_2$ corresponding to the component of $\nabla f(\mathbf{x}_{k-1})$ with the highest absolute value.
   b) Set $\tilde{S} = S_{k-1}$, and make a swap between the indices i and j

$$\tilde{S} = (S_{k-1} \setminus \{i\}) \cup \{j\}.$$

   Invoke DGN with input $(\mathbf{A}_i, y_i, \tilde{S}, 10^{-4}, 100)$ and obtain an output $\tilde{\mathbf{z}}$. Set $\tilde{\mathbf{x}} = \mathbf{U}_{\tilde{S}} \tilde{\mathbf{z}}$. Advance T: $T \leftarrow T + 1$.
   If $f(\tilde{\mathbf{x}}) < f(\mathbf{x}_{k-1})$, then set $S_k = \tilde{S}$, $\mathbf{x}_k = \tilde{\mathbf{x}}$, advance k and goto 2.a.
   c) If none of the swaps resulted with a better objective function value, then STOP. The output is $\mathbf{x} = \mathbf{x}_{k-1}$ and T.

C. The GESPAR Algorithm

The 2-opt method can have the tendency to get stuck at local optima points. Therefore, our final algorithm, which we call GESPAR, is a restarted version of 2-opt. The 2-opt method is repeatedly invoked with different initial random support sets until

the resulting objective function value is smaller than a certain threshold (success) or the number of maximum allowed total number of swaps was passed (failure). A detailed description of the method is given in Algorithm 3. One element of our specific implementation that is not described in Algorithm 3 is the incorporation of random weights added to the objective function, giving randomly different weights to the different measurements. Namely, the objective function used is actually chosen as $f(\mathbf{x}) = \sum_{i=1}^{N} w_i(\mathbf{x}^T\mathbf{A}_i\mathbf{x} - y_i)^2$ with $w_i = 1$ or 2 with equal probability. The random generation of weights is done each time the DGN procedure is invoked. We observed that this modification reduced the probability of the 2-opt procedure to get stuck in non-optimal points.

Algorithm 3 GESPAR

Input: $(\mathbf{A}_i, y_i, \tau, \mathrm{ITER})$.

$\mathbf{A}_i \in \mathbb{R}^{N\times N}$, $i = 1, 2, \ldots, N$—symmetric matrices.

$y_i \in \mathbb{R}$, $i = 1, 2, \ldots, N$.

τ—threshold parameter.

ITER—Maximum allowed total number of swaps.

Output: $\mathbf{x}$—an optimal (or suboptimal) solution of (4).

Initialization. Set $C = 0$, $k = 0$.

- **Repeat**
  Invoke the 2-opt method with input $(\mathbf{A}_i, y_i)$ and obtain an output $\mathbf{x}$ and T. Set $\mathbf{x}_k = \mathbf{x}$, $C = C + T$ and advance k: $k \leftarrow k + 1$.
  Until $f(\mathbf{x}) < \tau$ or $C > \mathrm{ITER}$.
- The output is $\mathbf{x}_\ell$ where $\ell = \underset{m=0,1\ldots k-1}{\arg\min} f(\mathbf{x}_m)$.

IV. Fourier Implementation Details

In principle, GESPAR may be used to find sparse solutions to any system of quadratic equations, i.e. problems of the form:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \sum_{i=1}^{N} \left(\mathbf{x}^T\mathbf{A}_i\mathbf{x} - y_i\right)^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_0 \le s, \\ & \mathbf{x} \in \mathbb{R}^N. \end{aligned} \tag{13}$$

However, when the matrices $\mathbf{A}_i$ correspond to transforms that can be implemented efficiently, GESPAR takes on a particularly simple form.

For example, consider the case in which $\{\mathbf{A}_i\}$ represent Fourier measurements. In this case, the creation and storing of the matrices $\mathbf{A}_i$ defined in Section II, can be avoided in the implementation, by using the FFT. Specifically, to calculate the weighted objective function, we note that

$$f(\mathbf{x}) = \sum_{i=1}^{N} w_i(\mathbf{x}^T\mathbf{A}_i\mathbf{x} - y_i)^2 = \sum_{i=1}^{N} w_i\left(|\hat{x}_i|^2 - y_i\right)^2 \tag{14}$$

where $\hat{x}_i$ is the ith DFT component of $\mathbf{x}$, which can be computed via the FFT. Clearly, $J(\mathbf{z})$, which is used in the DGN procedure (Algorithm 1) can also be computed efficiently since $\mathbf{B}_i = \mathbf{U}_S^T\mathbf{A}_i\mathbf{U}_S$ only involves a small (s) number of columns of the Fourier matrix $\mathbf{F}$.

The FFT can also be used in the calculation of the gradient $\nabla f(\mathbf{x})$, used in the 2-opt stage 2:

$$\begin{aligned} \nabla f(\mathbf{x}) &= 4\sum_i w_i(\mathbf{x}^T\mathbf{A}_i\mathbf{x} - y_i)\mathbf{A}_i\mathbf{x} \\ &= 4N\mathrm{IFFT}\left[\left(|\hat{x}_i|^2 - y_i\right) w_i\hat{x}_i\right]. \end{aligned} \tag{15}$$

Consequently, in no step of the algorithm is it necessary to calculate the set of matrices $\mathbf{A}_i$ explicitly.

This fact is even more important in the 2D Fourier phase retrieval problem, as the relevant vector sizes become very large. Since a major advantage of GESPAR over other methods (e.g. SDP based) is its low computational cost, GESPAR may be used to find a sparse solution to the 2D Fourier phase retrieval—or phase retrieval of images. The only adjustments needed in the algorithm are in the implementation, for example, using FFT2 instead of storing the large matrices $\mathbf{A}_i$.

Fig. 1 shows a recovery example of a sparse 195 × 195 pixel image, comprised of $s = 15$ circles at random locations and random values on a grid containing 225 points, recovered from its 38,025 2D-Fourier magnitude measurements, using GESPAR. The dictionary used in this example contains 225 elements consisting of non-overlapping circles located on a 15 × 15 point cartesian grid, each with a 13 pixel diameter. The solution took 80 seconds. Solving the same problem using the sparse Fienup algorithm did not yield a successful reconstruction, and using the SDP method is not practical due to the large matrix sizes.

Further investigation of the algorithm's performance in the 2D case is presented in Section V. An implementation of GESPAR in the 1D and 2D Fourier cases can be found online [17].

V. Numerical Simulations

In order to demonstrate the performance of GESPAR, we conduct several numerical simulations. The algorithm is compared to other existing methods, and is evaluated in terms of signal-recovery accuracy, computational efficiency, and robustness to noise.

A. Signal-Recovery Accuracy

In this subsection we examine the recovery success rate of GESPAR as a function of the number of non-zero elements in the signal. A runtime comparison of the tested methods is also performed.

We choose $\bar{\mathbf{x}}$ as a random vector of length n. The vector contains uniformly distributed values in the range $[-4, -3] \cup [3, 4]$ in s randomly chosen elements. The N point DFT of the signal is calculated, and its magnitude-square is taken as $\mathbf{y}$, the vector of measurements. The $2n - 1$ point correlation is also calculated. In order to recover the unknown vector $\mathbf{x}$, the GESPAR algorithm is used with $\tau = 10^{-4}$ and $ITER = 6400$. We also test two other algorithms for comparison purposes: An SDP based algorithm (Algorithm 2, [9]), and an iterative Fienup algorithm with a sparsity constraint [12]. In our simulation $n = 64$ and $N = 128$. The Sparse-Fienup algorithm is run using 100

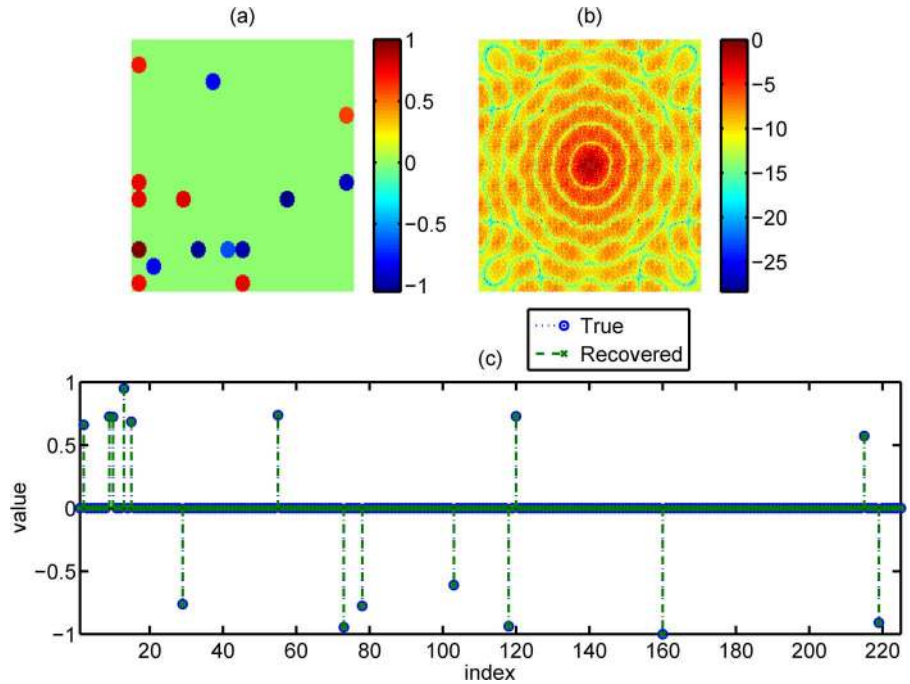

Fig. 1. 2D Fourier phase retrieval example. (a) True 195 × 195 sparse circle image ($s = 15$ circles). (b) Measured 2D Fourier magnitude (38,025 measurements, log scale). (c) True and recovered coefficient vectors, corresponding to circle amplitudes at each of the 225 grid points.

TABLE I
RUNTIME COMPARISON

	SDP		Sparse-Fienup		GESPAR	
	recovery %	runtime sec	recovery %	runtime sec	recovery %	runtime sec
$s = 3$	0.93	1.32	0.98	0.09	1	0.04
$s = 5$	0.86	1.78	0.97	0.12	1	0.05
$s = 8$	0.9	3.85	0.82	0.50	1	0.06

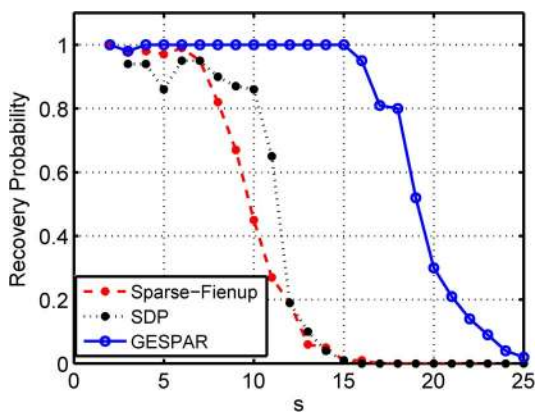

Fig. 2. Recovery probability versus sparsity (s)

random initial points, out of which the chosen solution is the one that best matches the measurements. Namely, $\hat{\mathbf{x}}$ is selected as the s sparse output of the Sparse-Fienup algorithm with the minimal cost $f(\mathbf{x}) = \sum_{i=1}^{N} \left(|\mathbf{F}_i\mathbf{x}|^2 - y_i\right)^2$ out of the 100 runs.

Signal recovery results of the numerical simulation are shown in Fig. 2, where the probability of successful recovery is plotted for different sparsity levels. The success probability is defined as the ratio of correctly recovered signals $\mathbf{x}$ out of 100 simulations. In each simulation both the support and the signal values are randomly selected. The three algorithms (GESPAR, SDP and Sparse-Fienup) are compared. The results clearly show that GESPAR outperforms the other methods in terms of probability of successful recovery—over 90% successful recovery up to $s = 15$, vs. $s = 8$ and $s = 7$ in the other two techniques.

Average runtime comparison of the three algorithms is shown in Table I for $n = 64$ and $N = 128$. The runtime is averaged over all successful recoveries. The computer used has an intel i5 CPU and 4 GB of RAM. As seen in the table, the SDP based algorithm is significantly slower than the other two methods. Fienup iterations are slightly slower than GESPAR and lead to a much lower success rate. In these simulations, GESPAR is both fast and more accurate than its competitors.

B. Sensitivity to Exact Sparsity Knowledge

Since the exact value of the signal's sparsity s may not be known, the performance of GESPAR is examined when only an upper limit on s is given. To this end we run GESPAR twice: Once with s known exactly at each realization, and once with only an upper limit on s. The upper limit is taken as 25. The other simulation settings are the same as in Section V-A.

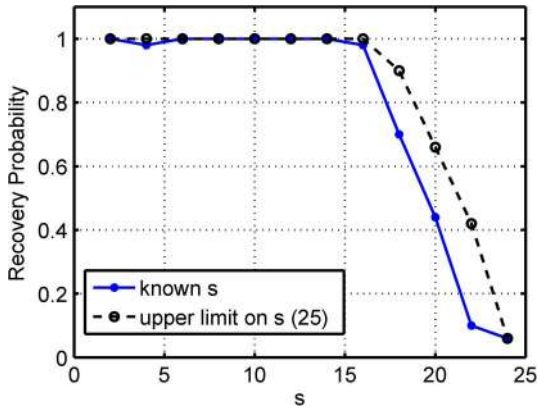

Fig. 3. Effect of unknown exact sparsity level s on recovery probability.

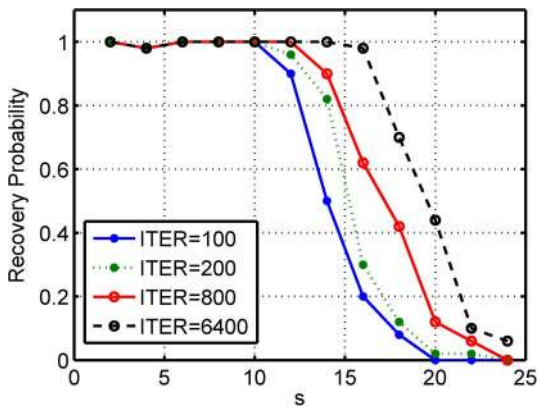

Fig. 4. Effect of number of swaps $(ITER)$ on recovery probability.

Fig. 3 shows the probability for successful recovery of the two simulations. The rather loose upper limit on s does not seem to affect the results significantly—in fact, the performance is somewhat improved when allowing more nonzero elements during the iterations.

C. Effect of the Number of Allowed Swaps

One of the stopping criteria for the GESPAR algorithm is when the total number of swaps exceeds a predefined parameter (the input parameter $ITER$ in Algorithm 3). Naturally, increasing the allowed number of index swaps will increase the probability of finding a correct solution, but at the cost of increased computation time. It is therefore important to quantify this effect, which is the purpose of the current simulation.

We run GESPAR with the same parameters as in Section V-A several times, where in each simulation a different value for the parameter $ITER$ is used, in the range [100,25600]. Fig. 4 shows the results. As expected, increasing the number of possible swaps increases the recovery probability. Note that increasing the value of $ITER$ above 6400 demonstrated no improvement in the recovery results—for the unsuccessful recoveries, increasing the number of swaps even up to $ITER = 25600$ did not help. This means that for these simulation values (e.g. $N = 128$, $s < 25$), using a value of $ITER$ larger than 6400 only increases computation time without improving the results.

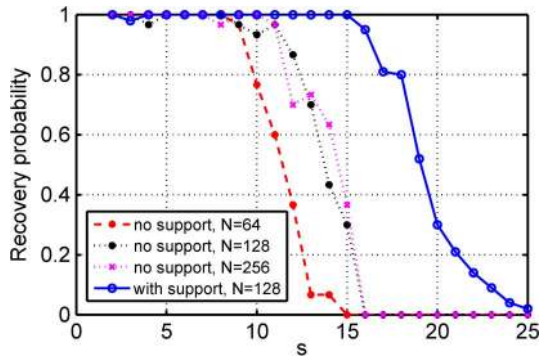

Fig. 5. Effect of oversampling and support information from the autocorrelation sequence $(n = 64)$.

D. Effect of Oversampling and Support Information

Here we examine the effect of oversampling and of autocorrelation-derived support information. GESPAR is run on random vectors $\mathbf{x}$ of length $n = 64$, with a varying amount of noiseless Fourier magnitude measurements, obtained by the N point DFT of $\mathbf{x}$ with $N = 64$, 128, 256. In these cases, no support information was used—i.e. $J_1 = \{1\}$ and $J_2 = \{1, 2, \ldots, n\}$. In addition, in order to investigate the effect of support information, we run GESPAR with $n = 64$, $N = 128$ (i.e. oversampling by a factor of 2), and use the support information derived from the autocorrelation sequence. The results, shown in Fig. 5, clearly show that both oversampling and support information improve performance.

E. Robustness to Noise

We now evaluate GESPAR as a function of SNR, and compare it with sparse Fienup [12]. The SDP based method presented in [9] is not designed to deal with noise and therefore we did not apply it here. The SDP approach of [10] considers random measurements, and does not produce comparable results from direct Fourier measurements.

As in Section V-A, we choose $\bar{\mathbf{x}}$ as a vector of length n, with s randomly chosen elements containing uniformly distributed values, and evaluate its N point Fourier magnitude-square. White-gaussian noise $\mathbf{v}$ is added to the measurements, at different SNR values, defined as: $SNR = 20 \log \frac{\|\mathbf{y}\|}{\|\mathbf{v}\|}$. In order to recover the unknown vector $\mathbf{x}$, the GESPAR algorithm is used with $\tau = 10^{-4}$ and $ITER = 10000$, as well as the sparse-Fienup algorithm, for comparison purposes. In our simulation $n = 64$ and $N = 128$. The sparse-Fienup algorithm is run with a maximum of 1000 iterations, and with 100 random initial points.

Note that even with little noise, the information on the support obtained by the zeros of the autocorrelation is no longer available. This is due to the fact that in the presence of noise, there will be no true zeros in the measured (or calculated) autocorrelation. In this case, one might try to threshold the autocorrelation values, rendering small autocorrelation values as zeros. However, this might result in zeroing of small (yet non-zero) values of the true autocorrelation function. Therefore, in the noisy case, we do not use support information obtained by the autocorrelation function in GESPAR, namely $J_2 = \{1, 2, \ldots, n\}$.

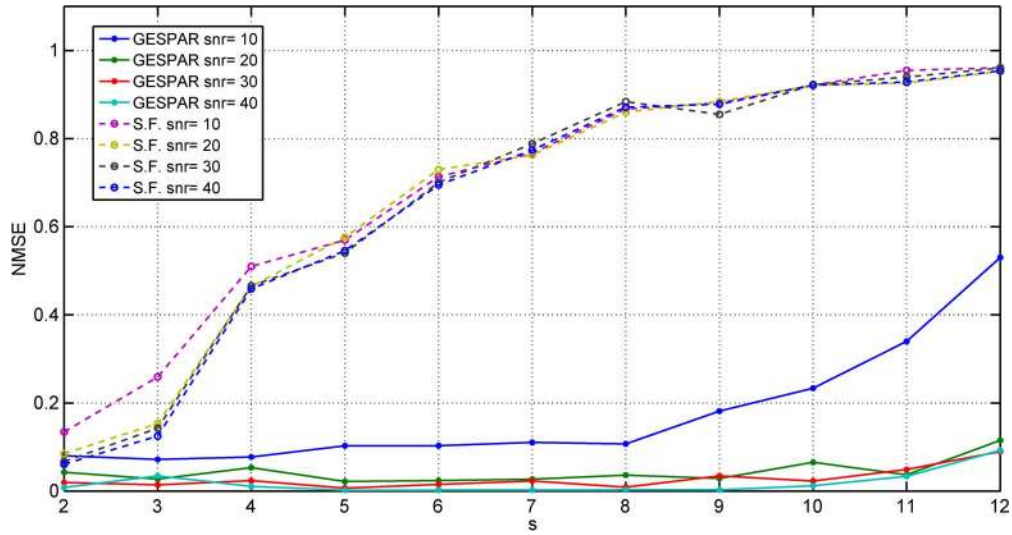

Fig. 6. Normalized MSE versus sparsity level. The performance is plotted for several SNR values for GESPAR (full lines) and Sparse-Fienup (S.F.—dashed lines).

Fig. 6 shows the normalized mean squared reconstruction error (NMSE), defined as $NMSE = \frac{\|\mathbf{x}-\hat{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2}$, as a function of sparsity, for different SNR values. Each point represents an average over 100 different random realizations. The performance under different SNR values is plotted for GESPAR (full lines), and for sparse-Fienup (dashed-lines). The performance of GESPAR naturally improves as SNR increases, and it clearly outperforms sparse-Fienup in terms of noise-robustness.

F. Scalability

As one of the main advantages of GESPAR over SDP based methods is its ability to solve large problems efficiently, we now examine its performance for different vector sizes.

We simulate GESPAR for various values of $n \in [64, 2048]$. In all cases $N = 2n$. The other simulation parameters are as in Section V-A. The recovery probability vs. sparsity s for different vector lengths is shown in Fig. 7. The maximal sparsity s allowing successful recovery is shown to increase with vector length n, and seems to scale like $n^{1/3}$, which is consistent with the same scaling observation presented in [9]. The mean reconstruction time for a signal with $n = 512$, $s = 35$ from $N = 1024$ measurements, allowing $ITER = 6400$ replacements, is 33.5 seconds. For comparison, a corresponding plot representing the scalability of the sparse-Fienup algorithm is presented in Fig. 8. Plotting a similar scalability plot for the SDP based method is not possible due to the high computational cost which under our simulation conditions limits the application of this method to around $n \sim 400$.

G. Computation Time

The most time consuming part of GESPAR is the matrix inversion process in the DGN segment of the algorithm. Therefore, computation time scales approximately linearly with the number of swaps—as each swap corresponds to a single DGN iteration. The approximately linear dependence of runtime in the number of swaps is displayed in Fig. 9. Each point in the plot represents the mean time it took GESPAR to run $ITER$ iterations, averaged over 50 random input signals with $N = 128$, $n = 64$, $s = 10$.

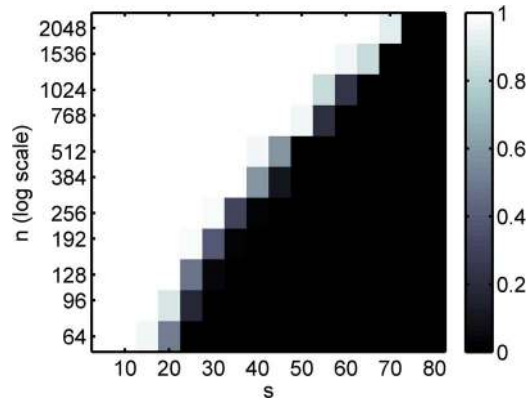

Fig. 7. 1D-Scalability—GESPAR recovery probability as a function of signal sparsity s, for various vector lengths ($n \in [64, 2048]$), and with oversampling, i.e. $N = 2n$. White corresponds to high recovery probability.

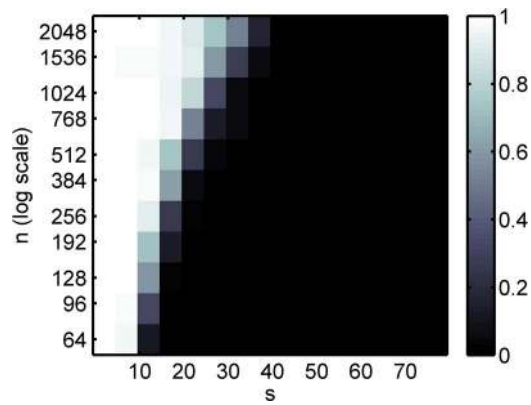

Fig. 8. Sparse Fienup scalability—recovery probability as a function of signal sparsity s, for various vector lengths ($n \in [64, 2048]$), and with oversampling, i.e. $N = 2n$. White corresponds to high recovery probability.

A major factor that determines the computation time is the number of index swaps required to find a solution. The mean

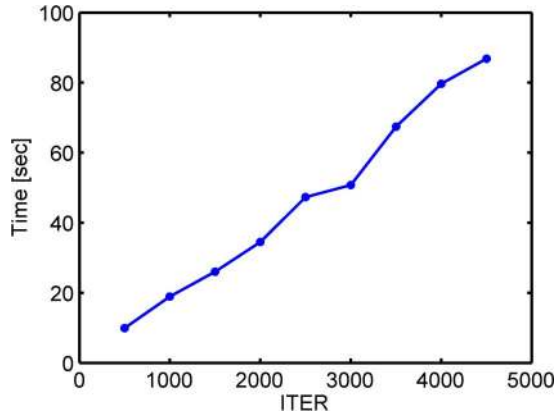

Fig. 9. Time vs. number of swaps ($ITER$).

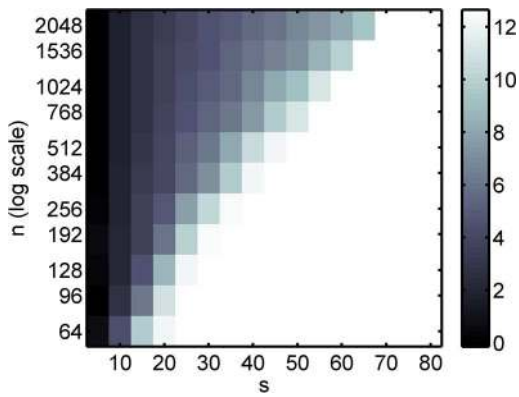

Fig. 10. Number of swaps as a function of s and n. The colorbar is in $\log_2$ scale, e.g. $10 \Rightarrow 2^{10} = 1024$ swaps.

number of swaps as a function of s, n is shown in Fig. 10. Beyond the successful recovery region (the white region in Fig. 7), the maximal number of swaps (6400) is used, without yielding a correct solution.

H. Two-Dimensional Fourier Phase Retrieval

In this section we apply GESPAR to 2D Fourier phase retrieval problems, showing its ability to solve large scale problems efficiently.

We generate random s-sparse 2D signals of sizes $\sqrt{n} \times \sqrt{n}$, with varying values for s and n, in the ranges $s \in [2 : 82]$ and $n \in [256 : 6400]$. Each signal is recovered from the noiseless magnitude of its 2D DFT, with no oversampling, using GESPAR. Similarly to the 1D noisy simulation, no autocorrelation-derived support information was used here. The parameter $ITER$ is taken as 6400. The recovery probability vs. sparsity s for different vector lengths is shown in Fig. 11. Similarly to the 1D case, the maximal sparsity allowing successful recovery increases with n. For comparison, Fig. 12 shows the result of a sparse-Fienup scalability simulation for the 2D case, under the same conditions, with 200 initial points per signal (increasing this parameter did not affect the results significantly). GESPAR is shown to outperform the sparse-Fienup method in the 2D case as well. As in the 1D case, a comparison to SDP based methods is not included here, since applying the SDP based method on the 2D case is very difficult due to memory limitations.

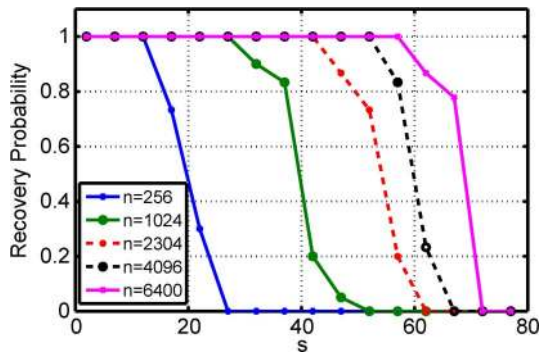

Fig. 11. GESPAR 2D-Scalability—recovery probability as a function of signal sparsity for various image sizes ($n = 256$, 1024, 2304, 4096, 6400).

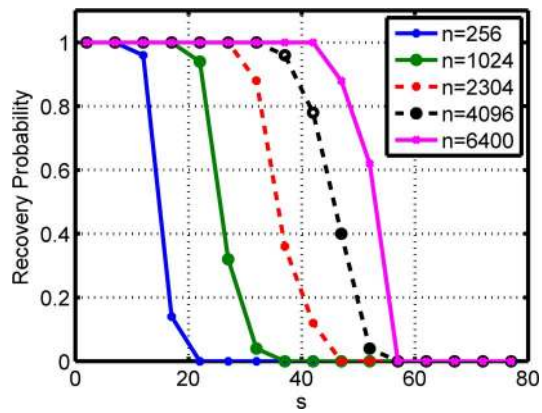

Fig. 12. Sparse-Fienup 2D-Scalability—recovery probability as a function of signal sparsity for various image sizes ($n = 256$, 1024, 2304, 4096, 6400).

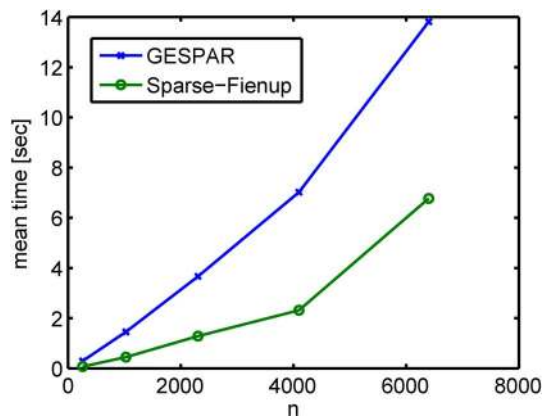

Fig. 13. Runtime comparison—average computation time for a succesful 2D recovery, for GESPAR and for sparse-Fienup, as a function of n.

A comparison between GESPAR and the sparse-Fienup method is shown in Fig. 13. The comparison shows the average time a successful recovery in the simulation took, as a function of vector size n. Sparse-Fienup is seen to be faster, however comparing Fig. 11 to Fig. 12 shows that GESPAR can recover signals up with a higher value of s: For example, when $n = 6400$, GESPAR recovers with very high probability signals up to sparsity $s = 57$, while sparse Fienup only recovers up to $s = 42$.

VI. Conclusion

We proposed and demonstrated GESPAR—a fast algorithm for recovering a sparse vector from its Fourier magnitude, or more generally, from quadratic measurements. We showed via simulations that GESPAR outperforms alternative approaches suggested for this problem in terms of complexity and success probability. The algorithm does not require matrix-lifting, and therefore is potentially suitable for large scale problems such as 2D images. The simulations demonstrated robustness of GESPAR to noise and other inexact knowledge, as well as its ability to successfully treat a variety of phase retrieval problems in one and two dimensions.

Appendix

Define the vector-valued function $\mathbf{h}$ by

$$\mathbf{h}(\mathbf{z}) = (h_1(\mathbf{z}), h_2(\mathbf{z}), \ldots, h_N(\mathbf{z}))^T,$$

with $h_i(\mathbf{z}) = \mathbf{z}^T\mathbf{B}_i\mathbf{z} - y_i$ With this notation, the vector $\mathbf{b}_k$ can be written as

$$\mathbf{b}_k = J(\mathbf{z}_{k-1})\mathbf{z}_{k-1} - \mathbf{h}(\mathbf{z}_{k-1}),$$

and the solution of the least-squares problem is

$$\begin{aligned}\tilde{\mathbf{z}}_k &= \left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1})\right)^{-1} J(\mathbf{z}_{k-1})^T \\ &\quad \times \left(J(\mathbf{z}_{k-1})\mathbf{z}_{k-1} - \mathbf{h}(\mathbf{z}_{k-1})\right) \\ &= \mathbf{z}_{k-1} - \left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1})\right)^{-1} J(\mathbf{z}_{k-1})^T \mathbf{h}(\mathbf{z}_{k-1}) \\ &= \mathbf{z}_{k-1} - \frac{1}{2}\left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1})\right)^{-1} \nabla g(\mathbf{z}_{k-1}). \qquad (16)\end{aligned}$$

Finally,

$$\mathbf{d}_k = \frac{1}{2}\left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1})\right)^{-1} \nabla g(\mathbf{z}_{k-1}). \qquad (17)$$

From (16) it follows that $-\mathbf{d}_k$ is a descent direction since

$$\begin{aligned}&-\mathbf{d}_k^T \nabla g(\mathbf{z}_{k-1}) \\ &= -\frac{1}{2}\nabla g(\mathbf{z}_{k-1})\left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1})\right)^{-1} \nabla g(\mathbf{z}_{k-1}) < 0. \qquad (18)\end{aligned}$$

We now show that the sequence generated by the DGN method is bounded. Indeed, since $-\mathbf{d}_k$ is a descent direction,

$$\begin{aligned}\sqrt{g(\mathbf{z}_0)} &\geq \sqrt{g(\mathbf{z}_k)} \\ &= \sqrt{\sum_{i=1}^{N}\left(\mathbf{z}_k^T\mathbf{B}_i\mathbf{z}_k - y_i\right)^2} \\ &\geq \frac{1}{\sqrt{N}}\sum_{i=1}^{N}\left|\mathbf{z}_k^T\mathbf{B}_i\mathbf{z}_k - y_i\right| \\ &\geq \frac{1}{\sqrt{N}}\left(\mathbf{z}_k^T\left(\sum_{i=1}^{N}\mathbf{B}_i\right)\mathbf{z}_k - \sum_{i=1}^{N} y_i\right),\end{aligned}$$

where the second inequality is due to Cauchy-Schwarz and the last inequality is a result of the fact that $\sum_{i=1}^{N}\mathbf{B}_i \succ 0$ and $y_i \geq 0$. Therefore,

$$\|\mathbf{z}_k\|^2 \leq \frac{1}{\lambda_{\min}\left(\sum_{i=1}^{N}\mathbf{B}_i\right)}\left(\sqrt{N g(\mathbf{z}_0)} + \sum_{i=1}^{N} y_i\right) \equiv \alpha,$$

proving that $\{\mathbf{z}_k\} \subseteq B[\mathbf{0}, \sqrt{\alpha}] = \{\mathbf{z} : \|\mathbf{z}\| \leq \sqrt{\alpha}\}$.

Since g is twice continuously differentiable, and $J(\mathbf{z})$ is continuous, it follows that there exists $M > 0$ and $\Lambda > 0$ such that $\lambda_{\max}(\nabla^2 g(\mathbf{z})) \leq M$ and $\lambda_{\max}(J(\mathbf{z})^T J(\mathbf{z})) \leq \Lambda$ for any $\mathbf{z} \in B[\mathbf{0}, 2\sqrt{\alpha}]$. In addition, since ∇g is continuous over $B[\mathbf{0}, 2\sqrt{\alpha}]$, there exist $\beta > 0$ such that $\|\nabla g(\mathbf{z})\| \leq \beta$ for all $\mathbf{z} \in B[0, 2\sqrt{\alpha}]$. Therefore, by (17) it follows that

$$\|\mathbf{d}_k\| \leq \frac{\beta}{2\underline{\lambda}}. \qquad (19)$$

The fact that $\lambda_{\max}(\nabla^2 g(\mathbf{z})) \leq M$ for all $\mathbf{z} \in B[\mathbf{0}, 2\sqrt{\alpha}]$ implies that ∇g is Lipschitz continuous over $B[\mathbf{0}, 2\sqrt{\alpha}]$ with parameter $M > 0$. Hence, by the descent lemma [15],

$$g(\mathbf{y}) \leq g(\mathbf{x}) + \nabla g(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \frac{M}{2}\|\mathbf{y} - \mathbf{x}\|^2 \qquad (20)$$

for any $\mathbf{x}, \mathbf{y} \in B[\mathbf{0}, 2\sqrt{\alpha}]$.

From $\|\mathbf{z}_{k-1}\| \leq \sqrt{\alpha}$ and $\|\mathbf{d}_k\| \leq \beta/(2\underline{\lambda})$, it follows that $\mathbf{z}_{k-1} - t\mathbf{d}_k \in B[\mathbf{0}, 2\sqrt{\alpha}]$ whenever $t \leq \frac{2\underline{\lambda}\sqrt{\alpha}}{\beta}$. Therefore, we can plug $\mathbf{y} = \mathbf{z}_{k-1} - t\mathbf{d}_k$ and $\mathbf{x} = \mathbf{z}_{k-1}$ into (20) to obtain

$$g(\mathbf{z}_{k-1} - t\mathbf{d}_k) \leq g(\mathbf{z}_{k-1}) - t\nabla g(\mathbf{z}_{k-1})^T\mathbf{d}_k + \frac{Mt^2}{2}\|\mathbf{d}_k\|^2.$$

Using (17),

$$\begin{aligned}\|\mathbf{d}_k\|^2 &= \frac{1}{4}\nabla g(\mathbf{x})^T\left(J(\mathbf{z}_k)^T J(\mathbf{z}_k)\right)^{-2}\nabla g(\mathbf{x}) \\ &\leq \frac{1}{4\underline{\lambda}}\nabla g(\mathbf{x})^T\left(J(\mathbf{z}_k)^T J(\mathbf{z}_k)\right)^{-1}\nabla g(\mathbf{x}) \\ &= \frac{1}{2\underline{\lambda}}\nabla g(\mathbf{z}_{k-1})^T\mathbf{d}_k,\end{aligned}$$

which yields

$$g(\mathbf{z}_{k-1}) - g(\mathbf{z}_{k-1} - t\mathbf{d}_k) \geq t\left(1 - \frac{M}{4\underline{\lambda}}t\right)\nabla g(\mathbf{z}_{k-1})^T\mathbf{d}_k.$$

Therefore, if $t \leq \min\left\{\frac{2\underline{\lambda}}{M}, \frac{2\underline{\lambda}\sqrt{\alpha}}{\beta}\right\}$, then

$$g(\mathbf{z}_{k-1}) - g(\mathbf{z}_{k-1} - t\mathbf{d}_k) \geq \frac{t}{2}\nabla g(\mathbf{z}_{k-1})^T\mathbf{d}_k. \qquad (21)$$

By the way the backtracking procedure is defined, we have that either $t_k = 1$ or $2t_k > \min\left\{\frac{2\underline{\lambda}}{M}, \frac{2\underline{\lambda}\sqrt{\alpha}}{\beta}\right\}$ and hence $t_k \geq \min\left\{1, \frac{\underline{\lambda}}{M}, \frac{\underline{\lambda}\sqrt{\alpha}}{\beta}\right\}$. Together with (21) this results in the inequality

$$\begin{aligned}g(\mathbf{z}_{k-1}) - g(\mathbf{z}_k) &\geq \frac{t_k}{2}\nabla g(\mathbf{z}_{k-1})^T\mathbf{d}_k \\ &\geq \min\left\{\frac{1}{2}, \frac{\underline{\lambda}}{2M}, \frac{\underline{\lambda}\sqrt{\alpha}}{2\beta}\right\}\nabla g(\mathbf{z}_{k-1})^T\mathbf{d}_k.\end{aligned}$$

Since

$$\begin{aligned}&\nabla g(\mathbf{z}_{k-1})^T\mathbf{d}_k \\ &\quad = \nabla g(\mathbf{x}_{k-1})^T\left(J(\mathbf{z}_{k-1})^T J(\mathbf{z}_{k-1})\right)^{-1}\nabla g(\mathbf{z}_{k-1}) \\ &\quad \geq \frac{1}{\Lambda}\left\|\nabla g(\mathbf{z}_{k-1})\right\|^2,\end{aligned}$$

we conclude that

$$g(\mathbf{z}_{k-1}) - g(\mathbf{z}_k) \geq C\left\|\nabla g(\mathbf{z}_{k-1})\right\|^2, \qquad (22)$$

where $C = \min\left\{\frac{1}{2\Lambda}, \frac{\underline{\lambda}}{2M\Lambda}, \frac{\underline{\lambda}\sqrt{\alpha}}{2\beta\Lambda}\right\}$. Noting that $\{g(\mathbf{z}_k)\}$ is a bounded below and nonincreasing sequence, it follows that it converges. The left-hand side of (22) therefore converges to zero and we obtain the result that $\nabla g(\mathbf{z}_k)$ converges to zero as

k tends to infinity. This fact also readily implies that all accumulation points of the sequence are stationary. Summing the inequality (22) over $p = 1, 2, \ldots, k+1$ we obtain that

$$g(\mathbf{z}_0) - g(\mathbf{z}_{k+1}) \geq C \sum_{p=1}^{k+1} \|\nabla g(\mathbf{z}_{p-1})\|^2,$$

and consequently, (also using the fact that $g(\mathbf{z}_{k+1}) \geq 0$),

$$g(\mathbf{z}_0) \geq C(k+1) \min_{p=1,\ldots,k+1} \|\nabla g(\mathbf{z}_{p-1})\|^2,$$

from which the inequality (12) follows.□

REFERENCES

[1] A. Walther, "The question of phase retrieval in optics," *J. Modern Opt.*, vol. 10, no. 1, pp. 41–49, 1963.

[2] R. Harrison, "Phase problem in crystallography," *JOSA A*, vol. 10, no. 5, pp. 1046–1055, 1993.

[3] N. Hurt, *Phase Retrieval and Zero Crossings: Mathematical Methods in Image Reconstruction*. Berlin, Germany: Springer-Verlag, 2001, vol. 52.

[4] J. Fienup, "Phase retrieval algorithms: A comparison," *Appl. Opt.*, vol. 21, no. 15, pp. 2758–2769, 1982.

[5] R. Gerchberg, "Super-resolution through error energy reduction," *J. Modern Opt.*, vol. 21, no. 9, pp. 709–720, 1974.

[6] H. Bauschke, P. Combettes, and D. Luke, "Phase retrieval, error reduction algorithm, and fienup variants: A view from convex optimization," *JOSA A*, vol. 19, no. 7, pp. 1334–1345, 2002.

[7] E. Candes, Y. C. Eldar, T. Strohmer, and V. Voroninski, "Phase Retrieval via Matrix Completion" 2011, *arXiv preprint arXiv:1109.0573*.

[8] Y. Shechtman, Y. C. Eldar, A. Szameit, and M. Segev, "Sparsity based sub-wavelength imaging with partially incoherent light via quadratic compressed sensing," *Opt. Exp.*, vol. 19, no. 16, pp. 14 807–14 822, 2011.

[9] K. Jaganathan, S. Oymak, and B. Hassibi, "Recovery of sparse 1-d signals from the magnitudes of their fourier transform," *CoRR*, vol. abs/1206.1405, 2012.

[10] H. Ohlsson, A. Yang, R. Dong, and S. Sastry, "Compressive Phase Retrieval From Squared Output Measurements via Semidefinite Programming" 2011, *arXiv preprint arXiv:1111.6323*.

[11] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase Recovery, Maxcut and Complex Semidefinite Programming" 2012, *arXiv preprint arXiv:1206.0102*.

[12] S. Mukherjee and C. Seelamantula, "An iterative algorithm for phase retrieval with sparsity constraints: Application to frequency domain optical coherence tomography," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2012, pp. 553–556, IEEE.

[13] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. New York, NY, USA: Dover, 1998.

[14] A. Beck and Y. C. Eldar, Sparsity Constrained Nonlinear Optimization: Optimality Conditions and Algorithms *arXiv preprint arXiv:1203.4580*, 2012.

[15] D. Bertsekas, *Nonlinear programming*. Nashua, NH, USA: Athena Scientific, 1999.

[16] A. Bjorck, *Numerical Methods for Least Squares Problems*. Philadelphia, PA, USA: Society for Industrial Mathematics, 1996, no. 51.

[17] Y. Shechtman, A. Beck, and Y. C. Eldar, GESPAR Code [Online]. Available: http://webee.technion.ac.il/people/YoninaEldar/software.php 2013

Yoav Shechtman received the B.Sc. degree in Electrical Engineering and the B.Sc degree in Physics in 2007 (cum Laude), and the Ph.D. in Physics 2013, all from the Technion—Israel Institute of Technology, Haifa, Israel. He is now a postdoctoral scholar in the Department of Chemistry in Stanford University, CA. Yoav's research interests include optical imaging, sparse information processing, and single molecule measurements. In 2011 he was awarded the Irwin and Joan Jacobs Excellence Scholarship for graduate studies and research, and in 2012 he was awarded the Gutwirth Excellence Scholarship for graduate studies. In 2013 he received the Hershel Rich Innovation Award.

Amir Beck received the B.Sc. degree in pure mathematics (cum laude) in 1991, the M.Sc. degree in operations research (summa cum laude), and the Ph.D. degree in operations research—all from Tel Aviv University (TAU). From 2003 to 2005, he was a Postdoctoral Fellow at the Minerva Optimization Center, Technion, Haifa, Israel. He is currently an Associate Professor in the Department of Industrial Engineering at The Technion—Israel Institute of Technology. His research interests are in continuous optimization, including theory, algorithmic analysis, and its applications. He has published numerous papers and has given invited lectures at international conferences. He was awarded the Salomon Simon Mani Award for Excellence in Teaching and the Henry Taub Research Prize award. He is in the editorial board of Mathematics of Operations Research, Operations Research and Journal of Optimization Theory and Applications. His research has been supported by various funding agencies, including the Israel Science Foundation, the German-Israeli Foundation, the Binational US-Israel foundation, the Israeli Science and Energy Ministries and the European community FP6 program.

Yonina C. Eldar (S'98–M'02–SM'07–F'12) received the B.Sc. degree in physics and the B.Sc. degree in electrical engineering both from Tel-Aviv University (TAU), Tel-Aviv, Israel, in 1995 and 1996, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, in 2002.

From January 2002 to July 2002, she was a Postdoctoral Fellow at the Digital Signal Processing Group at MIT. She is currently a Professor in the Department of Electrical Engineering at the Technion—Israel Institute of Technology, Haifa and holds the Edwards Chair in Engineering. She is also a Research Affiliate with the Research Laboratory of Electronics at MIT and a Visiting Professor at Stanford University, Stanford, CA. Her research interests are in the broad areas of statistical signal processing, sampling theory and compressed sensing, optimization methods, and their applications to biology and optics.

Dr. Eldar was in the program for outstanding students at TAU from 1992 to 1996. In 1998, she held the Rosenblith Fellowship for study in electrical engineering at MIT, and in 2000, she held an IBM Research Fellowship. From 2002 to 2005, she was a Horev Fellow of the Leaders in Science and Technology program at the Technion and an Alon Fellow. In 2004, she was awarded the Wolf Foundation Krill Prize for Excellence in Scientific Research, in 2005 the Andre and Bella Meyer Lectureship, in 2007 the Henry Taub Prize for Excellence in Research, in 2008 the Hershel Rich Innovation Award, the Award for Women with Distinguished Contributions, the Muriel & David Jacknow Award for Excellence in Teaching, and the Technion Outstanding Lecture Award, in 2009 the Technion's Award for Excellence in Teaching, in 2010 the Michael Bruno Memorial Award from the Rothschild Foundation, and in 2011 the Weizmann Prize for Exact Sciences. In 2012 she was elected to the Young Israel Academy of Science and to the Israel Committee for Higher Education, and elected an IEEE Fellow. In 2013 she received the Technion's Award for Excellence in Teaching, the Hershel Rich Innovation Award, and the IEEE Signal Processing Technical Achievement Award. She received several best paper awards together with her research students and colleagues. She is a Signal Processing Society Distinguished Lecturer, and Editor in Chief of Foundations and Trends in Signal Processing. In the past, she was a member of the IEEE SIGNAL PROCESSING THEORY AND METHODS and Bio Imaging Signal Processing technical committees, and served as an associate editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the EURASIP Journal of Signal Processing, the SIAM Journal on Matrix Analysis and Applications, and the SIAM Journal on Imaging Sciences.