Math. Prog. Comp. (2016) 8:253–269
DOI 10.1007/s12532-016-0105-y

FULL LENGTH PAPER

Revisiting compressed sensing: exploiting the efficiency of simplex and sparsification methods

Robert Vanderbei[1] · **Kevin Lin**[2] · **Han Liu**[1] · **Lie Wang**[3]

Received: 5 December 2013 / Accepted: 22 April 2016 / Published online: 9 May 2016

Abstract We propose two approaches to solve large-scale compressed sensing problems. The first approach uses the parametric simplex method to recover very sparse signals by taking a small number of simplex pivots, while the second approach reformulates the problem using Kronecker products to achieve faster computation via a sparser problem formulation. In particular, we focus on the computational aspects of these methods in compressed sensing. For the first approach, if the true signal is very sparse and we initialize our solution to be the zero vector, then a customized parametric simplex method usually takes a small number of iterations to converge. Our numerical studies show that this approach is 10 times faster than state-of-the-art methods for recovering very sparse signals. The second approach can be used when the sensing matrix is the Kronecker product of two smaller matrices. We show that the best-known sufficient condition for the Kronecker compressed sensing (KCS) strategy to obtain a perfect recovery is more restrictive than the corresponding condition if using the first approach. However, KCS can be formulated as a linear program with a very sparse constraint matrix, whereas the first approach involves a completely dense constraint matrix. Hence, algorithms that benefit from sparse problem representation, such as interior point methods (IPMs), are expected to have computational advantages for the KCS problem. We numerically demonstrate that KCS combined with IPMs is up to 10

The first author's research is supported by ONR Award N00014-13-1-0093, the third author's by NSF Grant III–1116730, and the fourth author's by NSF Grant DMS-1005539.

✉ Robert Vanderbei
rvdb@princeton.edu

[1] Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544, USA

[2] Department of Statistics, Carnegie Mellon University, Pittsburgh, PA 15213, USA

[3] Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

times faster than vanilla IPMs and state-of-the-art methods such as $\ell_1\_\ell_s$ and Mirror Prox regardless of the sparsity level or problem size.

Keywords Linear programming · Compressed sensing · Parametric simplex method · Sparse signals · Interior-point methods

Mathematics Subject Classification 65K05 · 62P99

1 Introduction and contribution overview

Compressed sensing (CS) aims to recover a sparse signal from a small number of measurements. The theoretical foundation of compressed sensing was first laid out by Donoho [9] and Candès et al. [4] and can be traced further back to the sparse recovery work of Donoho and Stark [14]; Donoho and Huo [12]; Donoho and Elad [10]. More recent progress in the area of compressed sensing is summarized in Elad [19] and Kutyniok [30].

Let $\mathbf{x}^* := (x_1^*, \ldots, x_n^*)^T \in \mathbb{R}^n$ denote a signal to be recovered. We assume n is large and $\mathbf{x}^*$ is sparse (i.e., many entries of x^* are zero). Let $\mathbf{A}$ be a given (or chosen) $m \times n$ matrix with $m < n$. Let a_{ij} denote the (i, j)th element of $\mathbf{A}$. The *compressed sensing problem* aims to recover $\mathbf{x}^*$ from the compressed, noise-free signal $\mathbf{y} := \mathbf{A}\mathbf{x}^* \in \mathbb{R}^m$.

Specifically, we wish to find the sparsest solution to an underdetermined linear system by solving

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ subject to } \mathbf{A}\mathbf{x} = \mathbf{y}, \tag{P$_0$}$$

where $\|\mathbf{x}\|_0 := \sum_{i=1}^n \mathbf{1}(x_i \neq 0)$. This problem is NP-hard because of the nonconvexity of the ℓ_0 pseudo-norm. To handle this challenge, Chen et al. [6] proposed the *basis pursuit* approach in which $\|\mathbf{x}\|_0$ is replaced by $\|\mathbf{x}\|_1 := \sum_{i=1}^n |x_i|$ to obtain a convex optimization problem

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ subject to } \mathbf{A}\mathbf{x} = \mathbf{y}. \tag{P$_1$}$$

Donoho and Elad [10] and Cohen et al. [7] provide conditions under which the solutions to Problems (P$_0$) and (P$_1$) are unique in their respective problems.

One key question is to understand what conditions guarantee that the solutions to (P$_0$) and (P$_1$) are equal. Various sufficient conditions have been discovered. For example, letting $\mathbf{A}_{*S}$ denote the submatrix of $\mathbf{A}$ with columns indexed by a subset $S \subset \{1, \ldots, n\}$, we say that $\mathbf{A}$ has the *k-restricted isometry property* (*k-RIP*) with constant δ_k if for any S with cardinality k,

$$(1 - \delta_k)\|\mathbf{v}\|_2^2 \leq \|\mathbf{A}_{*S}\mathbf{v}\|_2^2 \leq (1 + \delta_k)\|\mathbf{v}\|_2^2 \quad \text{for any } \mathbf{v} \in \mathbb{R}^k, \tag{1}$$

where $\|\mathbf{v}\|_2 := \sqrt{\sum_{j=1}^n v_j^2}$. We define $\delta_k(\mathbf{A})$ to be the smallest value of δ_k for which the matrix $\mathbf{A}$ has the k-RIP property. This property was first introduced by Candès et al. [4]. Under the assumption that $k := \|\mathbf{x}^*\|_0 \ll n$ and $\mathbf{A}$ satisfies the k-RIP condition,

Cai and Zhang [3] prove that whenever $\delta_k(\mathbf{A}) < 1/3$, the solutions to ($P_0$) and ($P_1$) are the same. The works in Donoho and Tanner [15–17] take a different approach by studying the convex geometric properties based on $\mathbf{A}$ instead of the RIP conditions to derive conditions when the solutions of (P_0) and (P_1) are the same.

Existing algorithms for solving the convex program (P_1) include interior-point methods [4,28], projected gradient methods [20], first-order methods [27] and Bregman iterations [45]. Besides algorithms that solve the convex program (P_1), several greedy algorithms have been proposed, including matching pursuit [32] and its many variants [11,13,22,33,34,38]. To achieve more scalability, combinatorial algorithms such as the so-called "Heavy Hitters on Steroids" (HHS) pursuit algorithm [24] and sub-linear Fourier transform [26] have also been developed.

In this paper, we revisit the optimization aspects of the classical compressed sensing formulation (P_1) and one of its extensions, Kronecker compressed sensing [18]. We consider two ideas for accelerating iterative algorithms. One reduces the total number of iterations, and the other reduces the computation required to do an iteration. We demonstrate the effectiveness of these ideas by thorough numerical simulations.

Our first idea, an optimization algorithm, is motivated by the fact that if the desired solution is very sparse, it should be reached after a relatively small number of simplex iterations starting from the zero vector. Such an idea motivates the usage of an optimization algorithm that can exploit the solution sparsity, e.g, the *parametric simplex method* (see, e.g., [8,41]). In this paper, we propose a customized parametric simplex method. Compared to standard simplex methods, our algorithm exploits a new pivot rule tailored for compressed sensing problems. It is well known that slightly altering the pivot rules and formulation of the simplex method can result in a significant increase in computational speed [21,35].

While the simplex method has an exponential computational complexity in the worse case [29], we emphasize that the parametric simplex method is a suitable optimization method for our compressed sensing problem. This matches existing theoretical results on the simplex method that state the "average complexity" or "smoothed complexity" of the simplex method is polynomial. See Adler et al. [1]; Spielman and Teng [37]; Post and Ye [36] and the references within.

Our second idea, a problem reformulation, requires the sensing matrix $\mathbf{A}$ to be the Kronecker product of two smaller matrices, $\mathbf{B}$ and $\mathbf{C}$. Since we are typically allowed to design the sensing matrix $\mathbf{A}$ ourselves, this requirement does not impose any practical limitations. This formulation results in a *Kronecker compressed sensing* (KCS) problem that has been considered before [18]. In our paper, we reformulate the linear program to ensure the constraint matrix is very sparse so that the problem can be solved efficiently. The computational advantage of using sparse constraint matrices has been well established in the linear programming literature [25,31,39,42]. While most optimization research in compressed sensing focuses on creating customized algorithms, our approach uses existing algorithms but a sparser problem formulation to speed up computation. To the best of our knowledge, such an idea has not been exploited in the compressed sensing literature yet.

Theoretically, KCS involves a tradeoff between computation and statistics: it gains computational advantages (as will be shown in the numerical section) at the price of requiring more measurements (i.e., larger m). More specifically, using sub-Gaussian

random sensing matrices (to be defined later), whenever $m = O(k^2 \log^2(\sqrt{n}/k))$, KCS recovers the true signal with probability at least $1 - 4\exp(-C\sqrt{m})$ for some constant $C > 0$. The sample requirement m for KCS is grows quadratically with k, as compared to the linear rate of $m = O(k \log(n/k))$ in standard compressed sensing. We provide more details in later sections.

The rest of this paper is organized as follows. In the next section, we describe how to solve the standard compressed sensing version (P_1) of the problem using the parametric simplex method. Then in Sect. 3 we describe the statistical foundation behind Kronecker compressed sensing (KCS). In Sect. 4 we present the sparse formulation of KCS that dramatically speeds up existing optimization algorithms (such as interior point methods). In Sect. 5 we provide numerical comparisons against state-of-the-art methods to show the advantage of our methods. The problem formulations considered in Sects. 2 through 5 involve no noise—the measurements are assumed to be exact. In Sect. 6 we conclude and discuss how our methods can be used to handle noisy or approximately-sparse settings of compressed sensing.

2 Compressed sensing via the parametric simplex method

Consider the following parametric perturbation to (P_1) for a specified μ:

$$\{\hat{\mathbf{x}}, \hat{\boldsymbol{\epsilon}}\} = \underset{\mathbf{x},\boldsymbol{\epsilon}}{\operatorname{argmin}} \ \mu\|\mathbf{x}\|_1 + \|\boldsymbol{\epsilon}\|_1 \quad \text{subject to} \ \ \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon} = \mathbf{y}, \tag{P_2}$$

where $\boldsymbol{\epsilon} \in \mathbb{R}^m$ denotes the residual. Since μ is a tuning parameter that affects the solution $\hat{\mathbf{x}}$, we should denote the solution to (P_2) as $\hat{\mathbf{x}}_\mu$, but for notational simplicity, we write $\hat{\mathbf{x}}$ as shown in (P_2) instead. For large values of μ, the optimal solution is $\hat{\mathbf{x}} = \mathbf{0}$ and $\hat{\boldsymbol{\epsilon}} = \mathbf{y}$. For small, strictly-positive values of μ, the situation reverses: $\hat{\boldsymbol{\epsilon}} = \mathbf{0}$ and we've solved the original problem (P_1). Belloni and Chernozhukov [2] considered the above formulation, and they provide statistical guarantees on the solution for a particular magnitude of μ.

In this section, we require that the sensing matrix $\mathbf{A}$ satisfies a suitable k-RIP property specified by the following lemma.

Lemma 1 [3] *For a small enough μ, let $\{\hat{\mathbf{x}}, \mathbf{0}\}$ be the optimal solution of (P_2), and let $k = \|\mathbf{x}^*\|_0$ be the sparsity of vector $\mathbf{x}^*$. If $\delta_k(\mathbf{A}) < 1/3$, then $\hat{\mathbf{x}} = \mathbf{x}^*$.*

Problem (P_2) is referred to as ℓ_1-penalized quantile regression. It can be solved by the parametric simplex method [8,41], which is a homotopy method used for sensitivity and perturbation analysis. In particular, we start at a large value of μ and successively reduce it to form a solution path. That is, the solution to Problem (P_2) for a particular value of μ serves as the initialization to solve the same problem for a smaller value of μ. It can be shown that such a solution path is piecewise linear and all the transition points can be easily calculated. Algorithmically, the parametric simplex algorithm calculates the full solution path until we arrive at a value of μ for which the optimal solution has $\hat{\boldsymbol{\epsilon}} = \mathbf{0}$, at which point we have solved the original problem (P_1).

The advantage of the parametric simplex method over the standard simplex method is that it solves the entire regularization path in terms of μ. Specifically, each basic solution produced by the algorithm is optimal for some interval of μ-values. Once the solution path indexed by μ is given, we could pick the desired μ as specified in Belloni and Chernozhukov [2] to satisfy certain statistical properties. However, we focus on picking the first μ for which $\hat{\boldsymbol{\epsilon}} = \mathbf{0}$. More details of the parametric simplex algorithm are illustrated in Fig. 1.

To formulate (P_2) as a linear program, we reparametrize it using nonnegative variables and equality constraints. To this end, we split each variable into the difference between two nonnegative variables:

$$\mathbf{x} = \mathbf{x}^+ - \mathbf{x}^- \quad \text{and} \quad \boldsymbol{\epsilon} = \boldsymbol{\epsilon}^+ - \boldsymbol{\epsilon}^-,$$

where the entries of $\mathbf{x}^+, \mathbf{x}^-, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^-$ are all nonnegative. The next step is to replace $\|\mathbf{x}\|_1$ with $\mathbf{1}^T(\mathbf{x}^+ + \mathbf{x}^-)$ and to make a similar substitution for $\|\boldsymbol{\epsilon}\|_1$. In general, the sum $x_j^+ + x_j^-$ does not equal the absolute value $|x_j|$ but it is easy to see that equality holds at optimality. This is a well-known and standard technique for rewriting problems involving ℓ_1-norms as linear programs. The resulting linear program becomes

$$\textbf{Simplex CS:} \quad \min_{\mathbf{x}^+, \mathbf{x}^-, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^-} \ \mu \mathbf{1}^T(\mathbf{x}^+ + \mathbf{x}^-) + \mathbf{1}^T(\boldsymbol{\epsilon}^+ + \boldsymbol{\epsilon}^-) \qquad (P_3)$$
$$\text{subject to } \mathbf{A}(\mathbf{x}^+ - \mathbf{x}^-) + (\boldsymbol{\epsilon}^+ - \boldsymbol{\epsilon}^-) = \mathbf{y}$$
$$\mathbf{x}^+, \mathbf{x}^-, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^- \geq \mathbf{0}.$$

For μ large enough, the optimal solution must be $\mathbf{x}^+ = \mathbf{x}^- = \mathbf{0}$, and $\boldsymbol{\epsilon}^+ - \boldsymbol{\epsilon}^- = \mathbf{y}$. Given that our variables are required to be nonnegative, we initialize $\boldsymbol{\epsilon}^+$ and $\boldsymbol{\epsilon}^-$ according to

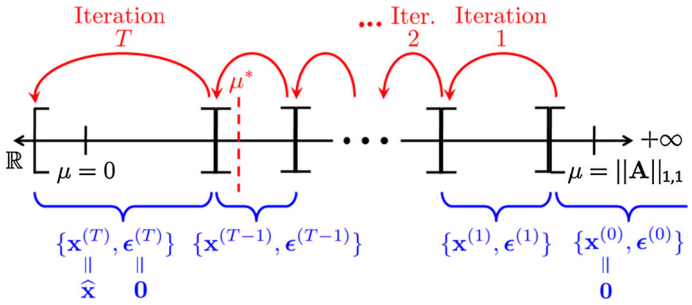

Fig. 1 Illustration of the parametric simplex method. The horizontal line (*black*) corresponds to varying values of μ. We explicitly use superscripts to denote the iteration counter for clarity (the solution path segment between two consecutive transition points is called an iteration). The *horizontal line* is partitioned into a finite number of intervals such that each interval corresponds to a solution $\{\mathbf{x}, \boldsymbol{\epsilon}\}$ that is optimal for any value of μ within that interval. μ is initialized to be $\|\mathbf{A}\|_1$, which ensures the initialization $\{\mathbf{x}^{(0)}, \boldsymbol{\epsilon}^{(0)}\}$ is optimal. The algorithm decreases μ toward 0 until it reaches a solution $\{\mathbf{x}^{(T)}, \boldsymbol{\epsilon}^{(T)}\}$ where $\boldsymbol{\epsilon}^{(T)} = \mathbf{0}$. Then $\mathbf{x}^{(T)} = \hat{\mathbf{x}}$ is our desired optimal solution to Problem (P_1). If the solution to (P_1) is unique, the interval corresponding to $\{\mathbf{x}^{(T)}, \boldsymbol{\epsilon}^{(T)}\}$ will contain $\mu = 0$. Since we obtain the entire solution path, other methods such as Belloni and Chernozhukov [2] can be used to pick the solution $\mathbf{x}^{(T-1)}$ corresponding to μ^*, though it is not the focus of our paper

$$y_i > 0 \implies \epsilon_i^+ = y_i > 0 \quad \text{and} \quad \epsilon_i^- = 0, \tag{2}$$

$$y_i < 0 \implies \epsilon_i^- = -y_i > 0 \quad \text{and} \quad \epsilon_i^+ = 0. \tag{3}$$

The equality case can be decided either way. Such an initialization is feasible for arbitrary μ. Furthermore, declaring the nonzero variables to be *basic* variables and the zero variables to be *nonbasic*, we use this as our initial basic solution for the parametric simplex method. We note that this initial solution is optimal for $\mu \geq \|\mathbf{A}\|_1 := \max_{1\leq j\leq n} \sum_{i=1}^m |a_{ij}|$, the largest column-wise ℓ_1 norm of $\mathbf{A}$. That is, for any solution $\{\mathbf{x}, \boldsymbol{\epsilon}\}$ such that $\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon}$, our initial solution setting $\mathbf{x} = \mathbf{0}$ and $\boldsymbol{\epsilon} = \mathbf{y}$ will be the global optima: $\|\mathbf{y}\|_1 = \|\mathbf{A}\mathbf{x} + \boldsymbol{\epsilon}\|_1 \leq \|\mathbf{A}\|_1\|\mathbf{x}\|_1 + \|\boldsymbol{\epsilon}\|_1 \leq \mu\|\mathbf{x}\|_1 + \|\boldsymbol{\epsilon}\|_1$. The pseudo-code to initialize and determine μ is presented in Algorithm 1, where Step 3 and Step 4 refer to the pivot step described in Chapters 7 and 8 of Vanderbei [43].

Algorithm 1 Pseudo-Code for Parametric Simplex Method

Require: Inputs $\mathbf{A}$ and $\mathbf{y}$ as in Problem (P_3).
1: Set $\mu = \|\mathbf{A}\|_{1,1}$. Set initial optimal solution to be $\mathbf{x}^+ = \mathbf{x}^- = \mathbf{0}$ and $\boldsymbol{\epsilon}^+$ and $\boldsymbol{\epsilon}^-$ to follow (2) and (3) respectively, as illustrated in Figure 1.
2: **while** $\boldsymbol{\epsilon}^+ + \boldsymbol{\epsilon}^- \neq \mathbf{0}$ **do**
3: Determine the smallest value of μ such that the current solution for (P_3) is optimal.
4: For the current value of μ, apply a simplex pivot step to determine a new feasible solution.
5: **end while**
6: **return** The optimal solution to (P_1), $\hat{\mathbf{x}} = \mathbf{x}^+ - \mathbf{x}^-$.

3 Kronecker compressed sensing formulation

In this section, we introduce the Kronecker compressed sensing idea [18]. While Duarte and Baraniuk [18] discusses Kronecker sensing to handle multi-dimensional signals, we show that representing one-dimensional signals as multi-dimensional signals will bear computational benefits. Unlike the previous section where we required the sensing matrix $\mathbf{A}$ to satisfy the $k-$RIP condition, in this section we impose a structural requirement on the sensing matrix. In particular, for given matrices $\mathbf{B}$ and $\mathbf{C}$, we consider the problem formulation that has the following representation:

$$\textbf{IPM CS:} \qquad \min \|\mathbf{x}\|_1 \ \text{ subject to } \ \big(\mathbf{B} \otimes \mathbf{C}\big)\mathbf{x} = \mathbf{y}. \tag{P_4}$$

The matrices $\mathbf{B}$ and $\mathbf{C}$ are of size $m_1 \times n_1$ and $m_2 \times n_2$ respectively, and $\mathbf{A}$, our sensing matrix, is $(m_1 m_2) \times (n_1 n_2)$ and defined as the Kronecker product of $\mathbf{B}$ and $\mathbf{C}$:

$$\mathbf{A} := \mathbf{B} \otimes \mathbf{C} = \begin{bmatrix} \mathbf{C}b_{11} & \cdots & \mathbf{C}b_{1n_1} \\ \vdots & \ddots & \vdots \\ \mathbf{C}b_{m_1 1} & \cdots & \mathbf{C}b_{m_1 n_1} \end{bmatrix}.$$

This Kronecker structural assumption on the new sensing matrix $\mathbf{A}$ motivates a new matrix-based sensing strategy based on a left and right sensing matrix. To see this,

assuming we want to recover a signal vector $\mathbf{x}^* \in \mathbb{R}^n$, we first reparameterize $\mathbf{x}^*$ into a matrix $\mathbf{X}^* \in \mathbb{R}^{n_2 \times n_1}$ by assigning each consecutive length-n_2 sub-vector of $\mathbf{x}^*$ to a column of $\mathbf{X}^*$. Here, without loss of generality, we assume $n = n_1 \times n_2$. Then, the observed matrix $\mathbf{Y}$ is given by $\mathbf{Y} := \mathbf{C}\mathbf{X}^*\mathbf{B}^T$. We define $\|\mathbf{X}\|_0 := \sum_{i,j} \mathbf{1}(x_{ij} \neq 0)$ and $\|\mathbf{X}\|_{1,1} := \sum_{i,j} |x_{ij}|$. Let the vec$(\cdot)$ operator take a matrix and concatenate its elements column-by-column to build one large column-vector containing all the elements of the matrix.

Given $\mathbf{Y} \in \mathbb{R}^{m_2 \times m_1}$ and the sensing matrices $\mathbf{B}$ and $\mathbf{C}$, we make one important observation. Recall the constraint in Problem (P_4). If $\mathbf{y} := \mathbf{A}\mathbf{x}^*$ where $\mathbf{x}^* = \text{vec}(\mathbf{X}^*)$, we have $\mathbf{y} = \text{vec}(\mathbf{Y})$. This means that Problem (P_4) is equivalent to

$$\min \|\mathbf{X}\|_{1,1} \quad \text{subject to} \quad \mathbf{C}\mathbf{X}\mathbf{B}^T = \mathbf{Y}. \tag{P_5}$$

In other words, if $\hat{\mathbf{x}}$ is the solution to (P_4) and $\hat{\mathbf{X}}$ is the solution to (P_5), then $\hat{\mathbf{x}} = \text{vec}(\hat{\mathbf{X}})$. Hence, we can interpret our Kronecker compressed sensing scheme as either having a Kronecker structural assumption on $\mathbf{A}$ or having two separate (left and right) sensing matrices $\mathbf{B}$ and $\mathbf{C}$.

Recall the definition of RIP given in Eq. (1). To understand the statistical theory of imposing the Kronecker structure on the sensing matrix $\mathbf{A}$, we use Lemma 2 from Duarte and Baraniuk [18], which establishes the relationships between the k-RIP constants of $\delta_k(\mathbf{B})$, $\delta_k(\mathbf{C})$, and $\delta_k(\mathbf{A})$.

Lemma 2 [18] *Suppose* $\mathbf{A} = \mathbf{B} \otimes \mathbf{C}$. *Then*

$$1 + \delta_k(\mathbf{A}) \leq (1 + \delta_k(\mathbf{B}))(1 + \delta_k(\mathbf{C})). \tag{4}$$

In addition, we define a sub-Gaussian distribution as follows.

Definition 1 (Sub-Gaussian distribution) We say a mean-zero random variable X follows a *sub-Gaussian distribution* if there exists some $\sigma \in \mathbb{R}_+$ such that

$$\mathbb{E} \exp(tX) \leq \exp\left(\frac{\sigma^2 t^2}{2}\right) \quad \text{for all } t \in \mathbb{R}.$$

It is clear that the Gaussian distribution with mean 0 and variance σ^2 satisfies the above definition. The next theorem provides sufficient conditions for perfect recovery of KCS.

Theorem 1 *Suppose the entries of matrices* $\mathbf{B}$ *and* $\mathbf{C}$ *follow a sub-Gaussian distribution with parameter* σ. *Then there exists a constant* $C > 0$ (*depending on* σ) *such that whenever*

$$m_1 \geq C \cdot k \log(n_1/k) \quad \text{and} \quad m_2 \geq C \cdot k \log(n_2/k),$$

the convex program (P_5) attains perfect recovery with probability

$$\mathbb{P}\left(\hat{\mathbf{X}} = \mathbf{X}^*\right) \geq 1 - \underbrace{\left(2e^{-\frac{m_1}{2C}} + 2e^{-\frac{m_2}{2C}}\right)}_{\rho(m_1,m_2)}.$$

Proof (for Theorem 1) We use the equivalence between Problem (P_5) and Problem (P_4). From Lemma 1 and Lemma 2, it suffices to show that $\delta_k(\mathbf{B})$ and $\delta_k(\mathbf{C})$ are both less than $2/\sqrt{3}-1$. Let $\tau := 2/\sqrt{3}-1$. From Theorem 9.2 of Foucart and Rauhut [23], there exist constants C_1 and C_2 (depending on σ) such that if $m_1 \geq 2C_1\tau^{-2}k\log(n_1/k)$ and $m_2 \geq 2C_2\tau^{-2}k\log(n_2/k)$, then

$$\begin{aligned}\mathbb{P}\left(\delta_k(\mathbf{B}) < \frac{2}{\sqrt{3}}-1 \text{ and } \delta_k(\mathbf{C}) < \frac{2}{\sqrt{3}}-1\right) &= 1-\left(\mathbb{P}\left(\delta_k(\mathbf{B}) \geq \tau\right)+\mathbb{P}\left(\delta_k(\mathbf{C}) \geq \tau\right)\right)\\ &\geq 1-\left(2e^{-\frac{\tau^2 m_1}{2C_1}} + 2e^{-\frac{\tau^2 m_2}{2C_2}}\right)\\ &\geq 1-\rho(m_1,m_2).\end{aligned}$$

□

An analogous result can be derived from the results in Duarte and Baraniuk [18] which uses the wavelet basis. From the above theorem, we see that for $m_1 = m_2 = \sqrt{m}$ and $n_1 = n_2 = \sqrt{n}$, whenever the number of measurements satisfies

$$m = O\left(k^2 \log^2\left(\frac{\sqrt{n}}{k}\right)\right), \tag{5}$$

we have $\hat{\mathbf{X}} = \mathbf{X}^*$ with probability at least $1 - 4\exp(-C\sqrt{m})$ for some constant C.

Here we compare the above result to that of (standard) compressed sensing problem (P_1). Following the same argument as in Theorem 1, whenever

$$m = O\left(k\log\left(\frac{n}{k}\right)\right), \tag{6}$$

we have $\hat{\mathbf{x}} = \mathbf{x}^*$ with probability at least $1-2\exp(-Cm)$. Comparing (6) to (5), we see that KCS needs more stringent conditions for perfect recovery. Specifically, for a fixed n, as k (the unknown sparsity level) increases, the required number of samples for KCS will grow quadratically with k rate as opposed to linearly. However, in the next section and in the numerical results, we will see that KCS enjoys a tremendous improvement in computation time. This will illustrate our tradeoff between computational performance and statistical recovery.

4 Sparsifying the constraint matrix for efficient computation

We can use standard LP algorithms such as an interior-point algorithm to solve (P_4). However, to achieve improved computational performance, we carefully formulate

the problem so that our algorithm can explicitly exploit the Kronecker structure of $\mathbf{A}$. The key to efficiently solving the linear programming problem associated with the Kronecker sensing problem lies in noting that the dense, Kronecker product $\mathbf{A}$ can be factored into a product of two sparse matrices:

$$\mathbf{A} = \begin{bmatrix} \mathbf{C}b_{11} & \cdots & \mathbf{C}b_{1n_1} \\ \vdots & \ddots & \vdots \\ \mathbf{C}b_{m_1 1} & \cdots & \mathbf{C}b_{m_1 n_1} \end{bmatrix} = \begin{bmatrix} \mathbf{C} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{C} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{C} \end{bmatrix} \times \begin{bmatrix} b_{11}\mathbf{I}_{n_2} & b_{12}\mathbf{I}_{n_2} & \cdots & b_{1n_1}\mathbf{I}_{n_2} \\ b_{21}\mathbf{I}_{n_2} & b_{22}\mathbf{I}_{n_2} & \cdots & b_{2n_1}\mathbf{I}_{n_2} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m_1 1}\mathbf{I}_{n_2} & b_{m_1 1}\mathbf{I}_{n_2} & \cdots & b_{m_1 n_1}\mathbf{I}_{n_2} \end{bmatrix} =: \mathbf{VW},$$

where $\mathbf{I}_{n_2}$ denotes an $n_2 \times n_2$ identity matrix and $\mathbf{0}$ denotes an $m_2 \times m_2$ zero matrix. Notice that while the matrix $\mathbf{A}$ is usually completely dense, it is a product of two very sparse matrices: $\mathbf{V} = \mathbf{I}_{m_1} \otimes \mathbf{C} \in \mathbb{R}^{m \times m_1 n_2}$ and $\mathbf{W} = \mathbf{B} \otimes \mathbf{I}_{n_2} \in \mathbb{R}^{m_1 n_2 \times n}$. Hence, if we introduce new variables $\mathbf{z}$, we can rewrite (P_4) equivalently as

$$\begin{aligned} \min_{\mathbf{x},\mathbf{z}} \ & \|\mathbf{x}\|_1 \\ \text{subject to} \ & \mathbf{Wx} - \mathbf{z} = 0 \\ & \mathbf{Vz} = \mathbf{y}. \end{aligned} \tag{P_6}$$

Using our previous matrix notation, we can rewrite (P_6) as

$$\begin{aligned} \textbf{IPM KCS:} \quad \min_{\mathbf{X},\mathbf{Z}} \ & \|\mathbf{X}\|_{1,1} \\ \text{subject to} \ & \mathbf{CX} - \mathbf{Z} = 0 \\ & \mathbf{ZB}^T = \mathbf{Y}. \end{aligned} \tag{P_7}$$

If we want to use a parametric simplex method to solve (P_6), we can, as before, split $\mathbf{x}$ and $\boldsymbol{\epsilon}$ into a difference between their positive and negative parts and enforce equality constraints to convert the problem into a linear program:

$$\begin{aligned} \textbf{Simplex KCS:} \quad \min_{\mathbf{x}^+,\mathbf{x}^-,\boldsymbol{\epsilon}^+,\boldsymbol{\epsilon}^-,\mathbf{z}} \ & \mu \mathbf{1}^T(\mathbf{x}^+ + \mathbf{x}^-) + \mathbf{1}^T(\boldsymbol{\epsilon}^+ + \boldsymbol{\epsilon}^-) \\ \text{subject to} \ & \mathbf{z} - \mathbf{W}(\mathbf{x}^+ - \mathbf{x}^-) = 0 \\ & \mathbf{Vz} + (\boldsymbol{\epsilon}^+ - \boldsymbol{\epsilon}^-) = \mathbf{y} \\ & \mathbf{x}^+, \mathbf{x}^-, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^- \geq 0. \end{aligned} \tag{P_8}$$

This formulation (P_8) has more variables and more constraints than (P_3), but now the constraint matrix is very sparse. We reiterate that while (P_8) and (P_3) are mathematically equivalent when $\mathbf{A}$ is a Kronecker product, the reparameterization in (P_8)

is more amendable to fast computational performance. Specifically, for linear programming, the sparsity of the constraint matrix is a significant contributor towards computational efficiency [25,39]. In fact, we can view the decomposition in (P_7) as a sparsification technique analogous to one step of the fast-Fourier optimization idea described in [44].

5 Numerical results and comparisons

Before showing our simulation results, we briefly describe three other popular methods to solve compressed sensing problems: $\ell_1\_\ell_s$, *Mirror Prox*, and *Fast Hard Thresholding Pursuit*. We will compare the performance of our method against these three methods on noiseless compressed sensing problems.

Proposed methods In this paper, we have presented two ideas, the parametric simplex method (an optimization algorithm) and Kronecker compressed sensing (a problem reformulation). We have made implementations that use these two ideas either separately or jointly. We use "KCS" to refer to optimization problems that explicitly exploit sparsity structure in the formulation, (P_7) and (P_8). We modify the parametric simplex algorithm described in Vanderbei [41] implemented in C found at http://www.orfe.princeton.edu/~rvdb/LPbook/src/index.html, to solve both (P_3) and (P_8). We refer to these implementations as "Simplex" and "Simplex KCS" respectively in our simulation results. We also use an interior-point solver called LOQO ([40]) to solve (P_4) and (P_7). We refer to these implementations as "IPM" and "IPM KCS" respectively in our simulation results.

(Specialized) interior-point method The $\ell_1\_\ell_s$ method [28] is a truncated Newton interior-point method using a preconditioner to solve

$$\hat{\mathbf{x}} = \operatorname*{argmin}_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}\|_1 \tag{P_9}$$

for a given regularization parameter λ. Since computing the Newton direction is computationally prohibitive for large-scale problems due to its Hessian, $\ell_1\_\ell_s$ circumvents this problem by approximating the Newton direction using preconditioned conjugate gradients. In general, interior-point methods are very efficient because they use second-order information. The MATLAB code for this algorithm can be found at http://stanford.edu/~boyd/l1_ls/.

Greedy method Fast Hard Thresholding Pursuit (FHTP) [22] is a greedy algorithm that alternates between two steps to approximately solve

$$\hat{\mathbf{x}} = \operatorname*{argmin}_{\mathbf{x}:\|\mathbf{x}\|_0=k} \|\mathbf{x}\|_1 : \mathbf{A}\mathbf{x} = \mathbf{y} \tag{P_{10}}$$

for a given sparsity level k. In the first step, it chooses the best k coordinates of $\mathbf{x}$ according to a certain criterion, and in the next step it optimizes $\mathbf{x}$ for only thse k coordinates while setting the remaining coordinates to 0. This algorithm is appealing because of its simplicity and its exact recovery as long as $\mathbf{A}$ satisfies an RIP condition

and k is correctly chosen. The MATLAB code for this algorithm can be found at http://www.math.drexel.edu/~foucart/HTP.zip.

First-order method The Mirror Prox algorithm [27] is a first-order algorithm that solves

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{x}\|_1 : \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2 \leq \delta \tag{P$_{11}$}$$

for a given tolerance δ by reformulating the problem as a saddle-point problem and using a proximal method to solve its variational inequality. First-order algorithms are favored in compressed sensing literature for their computational speed since they avoid calculating the Hessian matrix, and saddle point formulations are advantageous because they naturally combine the primal and dual of the problem via variational inequalities. The MATLAB code for this algorithm can be found at http://www2.isye.gatech.edu/~nemirovs/MirrorProxJan10_2012.zip.

Experimental protocol In the rest of this section, let $\mathbf{x}^*$ denote the true signal and $\hat{\mathbf{x}}$ denote the estimated signal using one of the above algorithms. We measure the accuracy of the solution by

$$\textit{Relative } \ell_1 \textit{ error: } \frac{\|\mathbf{x}^* - \hat{\mathbf{x}}\|_1}{\|\mathbf{x}^*\|_1} \quad \text{and} \quad \ell_\infty \textit{ error: } \|\mathbf{x}^* - \hat{\mathbf{x}}\|_\infty.$$

We compare five different algorithms, the parametric simplex method, the interior point method, $\ell_1\_\ell_s$, FHTP and Mirror Prox at different sparsity levels k. FHTP requires two different modes, oracle and agnostic. In the former, FHTP is given the true sparsity of $\mathbf{x}^*$. In the latter, FHTP is always given a sparsity of 100 regardless of the true $\mathbf{x}^*$.

Since each algorithm is solving a slightly different optimization problem, we devise a methodology for fair comparison. An important quantity to achieve fairness is the *imprecision*, $\|\mathbf{A}\hat{\mathbf{x}} - \mathbf{y}\|_2^2$, the degree to which the solution $\hat{\mathbf{x}}$ satisfies the constraints. We first apply our proposed methods (Simplex KCS, IPM KCS, Simplex CS, IPM CS) and record their imprecision. Since simplex methods are exact algorithms, we cannot reasonably expect the same magnitude of constraint error with $\ell_1\_\ell_s$ (P$_9$) and Mirror Prox (P$_{11}$). Hence, we require these last two to have imprecision of only up to two magnitudes more than the simplex-based algorithms. For each k, we found that $\lambda = 0.01$ in (P$_9$) achieves a comparable imprecision. Given the solution to $\ell_1\_\ell_s$, we can easily set δ in (P$_{11}$) to match the precision. The parametric simplex method and oracle FHTP naturally achieve the highest precision in most cases.

To ensure that each optimization algorithm is solving the same problem, we sample $\mathbf{A}$ and $\mathbf{B}$ as Gaussian matrices where each entry is a standard Gaussian (mean 0, standard deviation 1). "Simplex KCS" and "IPM KCS" use the matrices $\mathbf{V} = \mathbf{I} \otimes \mathbf{C}$ and $\mathbf{W} = \mathbf{B} \otimes \mathbf{I}$, while all the other methods use the sensing matrix $\mathbf{A} = \mathbf{B} \otimes \mathbf{C}$. In the following, we perform two different simulation sets. In the first simulation set, we vary the sparsity level of $\mathbf{x}$. In the second, we vary the length of $\mathbf{x}$. In either simulation set, we simulate 10 trials for each sparsity level or length. Instructions for downloading and running the various codes/algorithms described in this section can be found at http://www.orfe.princeton.edu/~rvdb/tex/CTS/kronecker_sim.html.

Table 1 This table shows the time (seconds), relative ℓ_1 error and relative ℓ_∞ error for 6 selected sparsity levels averaged (median) over 10 trials

	Sparsity $k = 2$			Sparsity $k = 20$		
	Time (s)	Rel. ℓ_1 Error	ℓ_∞ Error	Time (s)	Rel.ℓ_1 Error	ℓ_∞ Error
Simplex KCS	1e−8	0	0	2.000	0	0
IPM KCS	44.650	3.71e−10	9.41e−8	48.000	1.98e−8	1.36e−6
Simplex	2.000	0	0	3.500	0	0
IPM	730.350	3.78e−10	9.73e−8	758.700	2.54e−10	9.23e−8
l1ls	110.120	6.02e−6	1.08e−5	96.753	9.68e−6	0.00010
Mirror Prox	28.070	0.00025	0.00099	82.740	0.00143	0.01535
FHTP (Oracle)	16.912	1.30e−11	2.60e−11	14.270	2.67e−5	8.79e−5
FHTP (Agnostic)	191.575	0.25993	4.67440	196.25500	0.17329	3.49615
	Sparsity $k = 50$			Sparsity $k = 70$		
	Time (s)	Rel. ℓ_1 Error	ℓ_∞ Error	Time (s)	Rel.ℓ_1 Error	ℓ_∞ Error
Simplex KCS	12.000	0	0	25.500	0	0
IPM KCS	47.700	3.41e−8	7.62e−6	48.950	3.67e−8	6.95e−6
Simplex	19.500	0	0	66.000	0	0
IPM	758.700	3.49e−8	7.62e−6	821.250	4.23e−8	2.65e−6
l1ls	170.315	1.45e−5	0.00031	267.155	1.84e−5	0.00052
Mirror Prox	42.840	0.00011	0.06293	64.030	0.00015	0.11449
FHTP (Oracle)	13.273	0.00020	0.00160	15.575	0.00017	0.00144
FHTP (Agnostic)	146.215	0.00504	1.773	35.64750	0.00518	1.42405
	Sparsity $k = 100$			Sparsity $k = 150$		
	Time (s)	Rel. ℓ_1 Error	ℓ_∞ Error	Time (s)	Rel.ℓ_1 Error	ℓ_∞ Error
Simplex KCS	70.500	0	0	462.500	4.50e−6	0.00017
IPM KCS	49.950	5.2e−7	2.31e−5	56.500	1.40e−5	0.00136
Simplex	234.500	0	0	1587.500	2.00e−6	0.00010
IPM	783.450	5.31e−7	0.00035	794.500	1.50e−5	0.00150
l1ls	377.315	2.43e−5	0.00104	789.165	9.79e−5	0.00683
Mirror Prox	410.050	0.00011	0.42348	635.085	0.00036	2.43170
FHTP (Oracle)	16.231	0.00040	0.00471	79.677	0.01460	128.01000
FHTP (Agnostic)	20.439	0.00040	0.00471	148.945	0.01646	145.07500

If the median value is smaller than 1e−4, we write out at least two significant digits in scientific notation. We write "0" to denote exact recovery (achieved by only the simplex method). The first two rows of each table represent our proposed methods. Parametric simplex method outperforms other methods for very sparse problems $k \leq 70$. Naturally, FHTP (oracle) is the fastest for $k \leq 100$, but we see that both our methods outperform FHTP in relative ℓ_1 error and uniform error. By incorporating Kronecker structure, we see that previously slow methods can experience a drastic speed-up (i.e., the difference between IPM and IPM KCS)

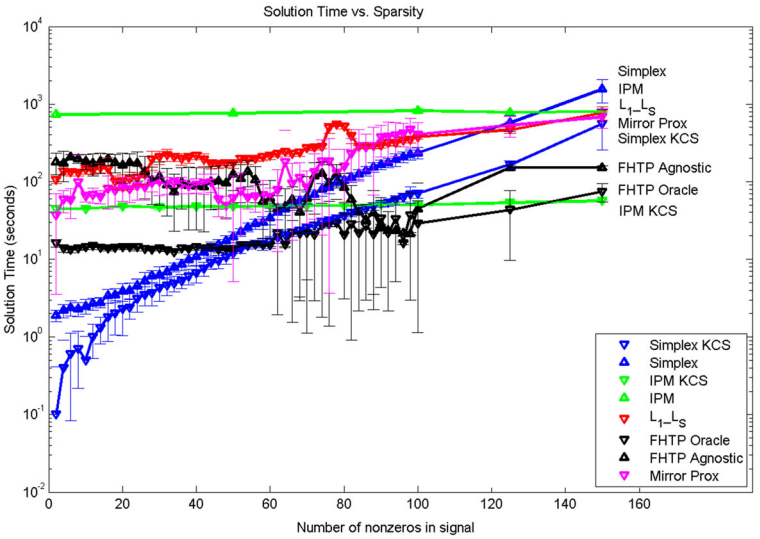

Fig. 2 Solution times for a large number of trials having $m = 1,122$, $n = 20,022$, and various degrees of sparsity in the underlying signal. The *horizontal axis* shows the number of nonzeros in the signal. The *vertical axis* gives a semi-log scale of solution times. The *error bars* have lengths equal to one standard deviation based on the multiple trials. Between Simplex KCS and IPM KCS, we outperform $\ell_1\_\ell_s$ and Mirror Prox

Results (Varying sparsity) In the first simulation set, we vary the true sparsity level by generating random problems using $m = 1,122 = 33 \times 34$ and $n = 20,022 = 141 \times 142$ and vary the number of nonzeros k in signal $\mathbf{x}^*$ from 2 to 150.

Table 1 provides time measured in seconds, the relative ℓ_1 error and the ℓ_∞ error averaged across 10 trials for each level of sparsity. All the simulation times are shown in Fig. 2. There are two observations. First, when the true sparsity is very small ($k \leq 70$), the parametric simplex method without the Kronecker structure (Simplex) outperforms most modern methods in terms of precision and time. Second, we see that previously slow methods (Simplex and IPM) are tremendously sped up once the problem is formulated using the Kronecker structure (Simplex KCS and IPM KCS). There is roughly a ten-fold improvement in speed. In fact, IPM KCS is uniformally faster than $\ell_1\_\ell_s$ and Mirror Prox. Our results show that if our sensing matrix has Kronecker structure and our algorithm is adapted to exploit the this property, simplex and interior point methods are highly competitive.

In four of the trials for Simplex throughout the entire simulation set, the solver erroneously reported that the solution was unbounded. We suspect this is due to numerical imprecision after the hundreds of thousands of simplex iterations for large values of k. This is not unexpected since we coded the entire parametric simplex method ourselves in C instead of using commercialized functions in MATLAB.

(Varying size) In the second simulation set, we vary the problem size by fixing the number of nonzeros k in signal $\mathbf{x}^*$ to 100 and constructing random problems using $m = 1,122 = 33 \times 34$ and varying $n = 141 \times 142$ to $n = 141 \times 402$. Let n_2 denote the varying dimension. Table 2 shows the same attributes as in the previous table. As

Table 2 This table, similar format as Table 1, shows 4 selected dimension sizes n_2 averaged (median) over 10 trials

	Size $n_2 = 102$			Size $n_2 = 202$		
	Time (s)	Rel. ℓ_1 Error	ℓ_∞ Error	Time (s)	Rel.ℓ_1 Error	ℓ_∞ Error
Simplex KCS	23.500	0	0	165.500	1.50e−6	1.50e−5
IPM KCS	40.732	3.54e−8	6.34e−8	92.351	2.56e−6	5.23e−5
Simplex	55.300	0	0	552.500	2.50e−6	2.50e−5
IPM	452.140	2.64e−8	4.34e−8	1409.206	6.35e−6	6.54e−5
l1ls	359.672	3.54e−5	1.21e−5	691.541	2.21e−5	1.62e−5
Mirror Prox	421.268	0.00985	0.00242	502.532	0.00983	0.00357
FHTP (Oracle)	24.327	0.30032	0.01964	115.465	0.51763	0.10540
	Size $n_2 = 302$			Size $n_2 = 402$		
	Time (s)	Rel. ℓ_1 Error	ℓ_∞ Error	Time (s)	Rel.ℓ_1 Error	ℓ_∞ Error
Simplex KCS	652.100	0.20000	0.00200	1750.100	0.50000	0.01023
IPM KCS	185.532	0.01729	0.40000	306.290	0.00966	0.40001
Simplex	2283.200	0.01063	0.01065	5555.700	0.01594	0.01597
IPM	3143.280	0.09238	0.01829	6503.130	0.12620	0.01810
l1ls	1541.290	0.20003	0.01731	2575.930	0.60002	0.09930
Mirror Prox	1043.022	0.21061	0.02327	1682.610	0.60639	0.12080
FHTP (Oracle)	382.556	1.16788	0.47230	727.805	1.19667	0.87309

Simplex KCS and IPM KCS outperform most methods in terms of time and accuracy. The parametric simplex (without Kronecker structure) performs better than $\ell_1\_\ell_s$ for $k = 100$ even as n_2 grows. Note that the ℓ_∞ error for $n_2 = 402$ is large for $\ell_1\_\ell_s$, Mirror Prox and FHTP, meaning at least one dimension is drastically incorrect

shown in Fig. 3, the relative order of the seven algorithms mostly persists throughout the simulation set. In particular, in all cases tried, Simplex KCS outperforms $\ell_1\_\ell_s$ and Mirror Prox. IPM KCS outperforms most methods throughout the entire simulation set.

6 Discussion and conclusions

We revisit compressed sensing from an optimization perspective. We advance the field of compressed sensing in two ways.

First, despite having an exponential worst-case complexity, the parametric simplex method is competitive for very sparse signals. It outperforms $\ell_1\_\ell_s$ and Mirror Prox under this regime (and we suspect many other methods) in both time and precision. Also, by adopting a parametric simplex method, we solve the problem for all values of μ, thereby finding the entire solution path in one shot. This feature of the parametric

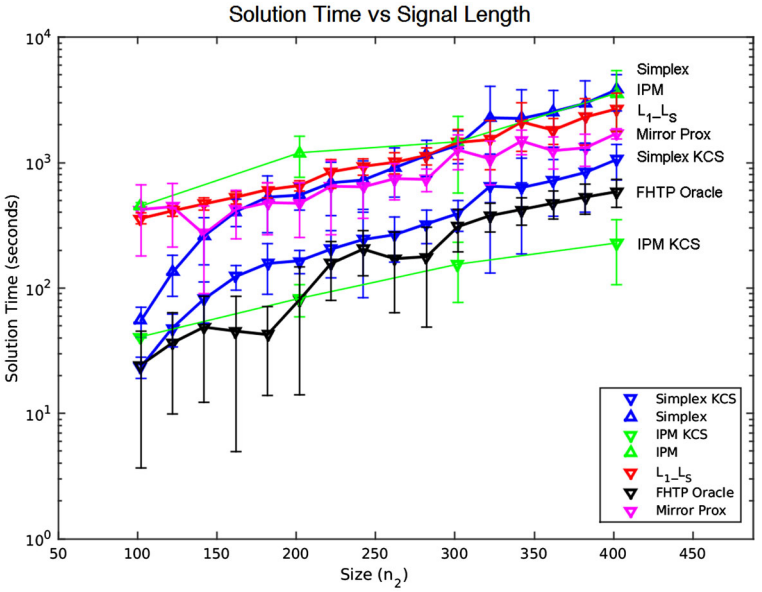

Fig. 3 Solution times for a large number of trials having $m = 1,122$ and $k = 100$. The number of dimensions for $\mathbf{x}^*$ in each trial is $n = 141 \times n_2$, and the *horizontal axis* shows the range of n_2. The *vertical axis* and *error bars* denote the same quantities as in Fig. 2. IPM KCS outperforms most methods throughout the entire simulation set

simplex method allows the user to pick a particular value of μ or take the largest μ for which $\boldsymbol{\epsilon} = 0$. Our paper focused on the latter.

Second, if we use the Kronecker structure, both the parametric simplex and interior point methods speed up ten-fold, making them competitive with other modern CS algorithms. But, as explained earlier, the Kronecker sensing problem involves changing the underlying problem being solved. The sensing matrix is now viewed as the Kronecker product of two sub-Gaussian matrices. While we presented this idea using only the parametric simplex and interior point methods, we expect this idea to benefit most optimization methods. This is left for future investigation. The Kronecker idea is inspired by the fast-Fourier transform where dense matrices are split into products of sparse matrices ([39], [44]). Hence, any optimization method that accommodates sparse matrix multiplication operations can potentially be altered to benefit further from a Kronecker compressed sensing scheme. The theoretical guarantees for using Kronecker compressed sensing are more stringent, however, which illustrates the tradeoff between computational efficiency and statistics.

In most applications, we expect some noise to corrupt the true signal. As long as the signal-to-noise ratio is high, we can adapt our methods to handle inexact constraints. Furthermore, the parametric simplex method computes solutions for all values of μ. As mentioned before, we can pick the solution associated with the value suggested in Belloni and Chernozhukov [2] to achieve statistical properties for noisy cases. If we have a specific residual size $\|\boldsymbol{\epsilon}\|_1$ that we are willing to tolerate, we can pick the appropriate solution from the solution path. On the other hand, if one were inter-

ested settings where $\mathbf{x}$ is approximately sparse and we wanted to recover the s-largest entries of $\mathbf{x}$, one could use either our "Simplex" or "IPM KCS" method. The relation between the solution and the true signal can then be determined through existing theory in Candès[5]. In future work, we plan to investigate noisy and approximately sparse settings more thoroughly and extend the proposed method to the setting of 1-bit compressed sensing.

Acknowledgments The authors would like to offer their sincerest thanks to the referees and the editors all of whom read earlier versions of the paper very carefully and made many excellent suggestions on how to improve it.

References

1. Adler, I., Karp, R.M., Shamir, R.: A simplex variant solving an $m \times d$ linear program in $O(\min(m_2, d_2)$ expected number of pivot steps. J. Complex. **3**, 372–387 (1987)
2. Belloni, A., Chernozhukov, V.: ℓ_1-penalized quantile regression in high-dimensional sparse models. Ann. Stat. **39**, 82–130 (2011)
3. Cai, T.T., Zhang, A.: Sharp RIP bound for sparse signal and low-rank matrix recovery. Appl. Comput. Harmonic Anal. **35**, 74–93 (2012)
4. Candès, E., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inf. Theory **52**, 489–509 (2006)
5. Candès, E.J.: The restricted isometry property and its implications for compressed sensing. C. R. Math. **346**, 589–592 (2008)
6. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM J. Sci. Comput. **20**, 33–61 (1998)
7. Cohen, A., Dahmen, W., Devore, R.: Compressed sensing and best k-term approximation. J. Am. Math. Soc. **22**, 211–231 (2009)
8. Dantzig, G.B.: Linear Programming and Extensions. Princeton University Press, Princeton (1998)
9. Donoho, D.L.: Compressed sensing. IEEE Trans. Inf. Theory **52**, 1289–1306 (2006)
10. Donoho, D.L., Elad, M.: Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1-minimization. Proc. Natl. Acad. Sci USA **100**, 2197–2202 (2003)
11. Donoho, D.L., Elad, M., Temlyakov, V.N.: Stable recovery of sparse overcomplete representations in the presence of noise. IEEE Trans. Inf. Theory **52**, 6–18 (2006)
12. Donoho, D.L., Huo, X.: Uncertainty principles and ideal atomic decomposition. IEEE Trans. Inf. Theory **47**, 2845–2862 (2001)
13. Donoho, D.L., Maleki, A., Montanari, A.: Message passing algorithms for compressed sensing. Proc. Natl. Acad. Sci. USA **106**, 18914–18919 (2009)
14. Donoho, D.L., Stark, P.B.: Uncertainty principles and signal recovery. SIAM J. Appl. Math. **49**, 906–931 (1989)
15. Donoho, D.L., Tanner, J.: Neighborliness of randomly projected simplices in high dimensions. Proc. Natl. Acad. Sci. **102**, 9452–9457 (2005)
16. Donoho, D. L., Tanner, J.: Sparse nonnegative solutions of underdetermined linear equations by linear programming. Proc. Natl. Acad. Sci. **102**, 9446–9451 (2005)
17. Donoho, D.L., Tanner, J.: Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing. Philos. Trans. Roy. Soc. S. A **367**, 4273–4273 (2009)
18. Duarte, M.F., Baraniuk, R.G.: Kronecker compressive sensing. IEEE Trans. Image Process. **21**, 494–504 (2012)
19. Elad, M.: Sparse and Redundant Representations—From Theory to Applications in Signal and Image Processing. Springer, New York (2010)
20. Figueiredo, M., Nowak, R., Wright, S.: Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. IEEE J. Sel. Top. Signal Process. **1**, 586–597 (2008)
21. Forrest, J.J., Goldfarb, D.: Steepest-edge simplex algorithms for linear programming. Math. Program. **57**, 341–374 (1992)

22. Foucart, S.: Hard thresholding pursuit: an algorithm for compressive sensing. SIAM J. Numer. Anal. **49**, 2543–2563 (2011)
23. Foucart, S., Rauhut, H.: A Mathematical Introduction to Compressive Sensing. Springer, New York (2013)
24. Gilbert, A.C., Strauss, M.J., Tropp, J.A., Vershynin, R.: One sketch for all: fast algorithms for compressed sensing. In: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing, pp. 237–246. ACM, New York (2007)
25. Gill, P.E., Murray, W., Ponceleon, D.B., Saunders, M.A.: Solving reduced KKT systems in barrier methods for linear and quadratic programming. Tech. rep, DTIC Document (1991)
26. Iwen, M.A.: Combinatorial sublinear-time Fourier algorithms. Found. Comut. Math. **10**, 303–338 (2010)
27. Juditsky, A., Karzan, F.K., Nemirovski, A.: ℓ_1 minimization via randomized first order algorithms. Université Joseph Fourier, Tech. rep. (2014)
28. Kim, S., Koh, K., Lustig, M., Boyd, S., Gorinevsky, D.: An interior-point method for large-scale l_1-regularized least squares. IEEE Trans. Sel. Top. Signal Process. **1**, 606–617 (2007)
29. Klee, V., Minty, G. J.: How good is the simplex method? Inequalities-III, pp. 159–175 (1972)
30. Kutyniok, G.: Compressed sensing: theory and applications. CoRR . arXiv:1203.3815 (2012)
31. Lustig, I.J., Mulvey, J.M., Carpenter, T.J.: Formulating two-stage stochastic programs for interior point methods. Oper. Res. **39**, 757–770 (1991)
32. Mallat, S., Zhang, Z.: Matching pursuits with time-frequency dictionaries. Signal Process. IEEE Trans. **41**, 3397–3415 (1993)
33. Needell, D., Tropp, J.A.: CoSaMP: iterative signal recovery from incomplete and inaccurate samples. Commun. ACM **53**, 93–100 (2010)
34. Needell, D., Vershynin, R.: Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. Found. Comut. Math. **9**, 317–334 (2009)
35. Pan, P.-Q.: A largest-distance pivot rule for the simplex algorithm. Eur. J. Oper. Res. **187**, 393–402 (2008)
36. Post, I., Ye, Y.: The simplex method is strongly polynomial for deterministic markov decision processes. Math. Oper. Res. **40**, 859–868 (2015)
37. Spielman, D.A., Teng, S.-H.: Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. J. ACM (JACM) **51**, 385–463 (2004)
38. Tropp, J.A.: Greed is good: algorithmic results for sparse approximation. IEEE Trans. Inf. Theory **50**, 2231–2242 (2004)
39. Vanderbei, R.: Splitting dense columns in sparse linear systems. Linear Algebra Appl. **152**, 107–117 (1991)
40. Vanderbei, R.: LOQO: an interior point code for quadratic programming. Optim. Methods Softw. **12**, 451–484 (1999)
41. Vanderbei, R.: Linear Programming: Foundations and Extensions, 3rd edn. Springer, New York (2007)
42. Vanderbei, R.J.: Alpo: another linear program optimizer. ORSA J. Comput. **5**, 134–146 (1993)
43. Vanderbei, R. J.: Linear programming. Foundations and extensions, International Series in Operations Research & Management Science, vol. 37 (2001)
44. Vanderbei, R.J.: Fast Fourier optimization. Math. Prog. Comp. **4**, 1–17 (2012)
45. Yin, W., Osher, S., Goldfarb, D., Darbon, J.: Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing. SIAM J. Imaging Sci. **1**, 143–168 (2008)