Primal-Dual First-Order Methods for a Class of Cone Programming

Zhaosong Lu*

March 9, 2011

Abstract

In this paper we study primal-dual first-order methods for a class of cone programming problems. In particular, we first present four natural primal-dual smooth convex minimization reformulations for them, and then discuss first-order methods, especially a variant of Nesterov's smooth (VNS) method [2] for solving these reformulations. The associated worst-case major arithmetic operation costs of the VNS method are estimated and compared. We conclude that the VNS method based on the last reformulation generally outperforms the others. Finally, we justify our theoretical prediction on the behavior of the VNS method by conducting numerical experiments on Dantzig selector, basis pursuit de-noising, MAXCUT SDP relaxation and Lovász capacity problems.

AMS 2000 subject classification: 65K05, 65K10, 90C05, 90C22, 90C25

1 Introduction

In [27, 28], Nesterov proposed an optimal algorithm for solving convex programming problems of the form

$$\inf\{f(u) : u \in \mathcal{U}\}, \tag{1}$$

where f is a convex function with Lipschitz continuous gradient and $\mathcal{U}$ is a sufficiently simple closed convex set. It is shown that his method has $\mathcal{O}(\sqrt{L/\epsilon})$ iteration-complexity bound, where L is the Lipschitz constant for the gradient of f and $\epsilon > 0$ is the absolute precision of the final objective function value. It shall be mentioned that each iterate of his method needs to solve two proximal subproblems or one proximal subproblem plus one projection subproblem. Recently, Auslender and Teboulle [2] proposed a variant of Nesterov's smooth (VNS) method that enjoys the same complexity as Nesterov's smooth method [27, 28], but it requires solving only one (proximal) subproblem per iteration. Thus, in contrast with Nesterov's smooth method, the

*Department of Mathematics, Simon Fraser University, Burnaby, BC, V5A 1S6, Canada (Email: zhaosong@sfu.ca). This author was supported in part by NSERC Discovery Grant.

VNS method [2] is generally computationally more efficient. More recently, Tseng [31] extended the VNS method [2] to a broader class of convex optimization problems and also proposed an alternative VNS method.

Lan, Lu and Monteiro [24] recently studied first-order methods for general cone programming (CP) problems. In particular, they proposed a class of primal-dual convex (smooth and/or nonsmooth) minimization reformulations for them, and discussed suitable first-order methods for solving these reformulations such as Nesterov's optimal method [27, 28], Nesterov's smooth approximation scheme [28], Nemirovski's prox-method [26] and a variant of Nesterov's optimal method [24]. They also provided iteration-complexity bounds for these first-order methods when applied to the aforementioned reformulations of CP problems. It shall be mentioned that the methods studied in [24] require solving two subproblems per iteration. Additionally, for a general CP, it is not clear which reformulation is mostly suitable for the user since the performance of first-order methods can vary much with different reformulations.

In this paper we consider a class of CP problems which, for example, include the MAXCUT semidefinite programming (SDP) relaxation [19], Lovász capacity [25] and those arising in compressed sensing [30, 14, 7, 8, 9, 10, 12, 13, 11]. In particular, we first present four natural primal-dual convex smooth minimization reformulations (13)-(16) for them, and then discuss first-order methods, especially the VNS method [2] for solving these reformulations. The associated worst-case major arithmetic operation costs of the VNS method are estimated and compared. We conclude that the VNS method based on reformulation (16) generally outperforms that applied to the others, namely, (13)-(15). Finally, we justify our theoretical prediction on the behavior of the VNS method by conducting numerical experiments on Dantzig selector, basis pursuit de-noising, MAXCUT SDP relaxation and Lovász capacity problems.

The rest of paper is organized as follows. In Section 2, we introduce a class of CP problems of our interest and present four natural primal-dual smooth convex minimization reformulations for them. In Sections 3 and 4, we review the VNS method [2] and discuss its application to these reformulations, respectively. In Section 5, we conduct numerical experiments to compare the performance of the VNS method when applied to the aforementioned reformulations of Dantzig selector, basis pursuit de-noising, MAXCUT SDP relaxation and Lovász capacity problems.

1.1 Notation

The following notation is used throughout our paper. All vector spaces given in this paper are assumed to be finite dimensional. The symbol $\Re^n$ denotes the n-dimensional Euclidean space. The 1-norm, Euclidean norm and infinity-norm in $\Re^n$ are denoted by $\|\cdot\|_1$, $\|\cdot\|_2$ and $\|\cdot\|_\infty$, respectively. We denote by $\mathbf{1}$ the vector of all ones whose dimension should be clear from the context. Given a sequence of vectors $x^i \in \Re^{n_i}$ for $i = 1, \ldots, m$, let $(x^1; \cdots; x^m)$ denote the vector in $\Re^n$ with $n = \sum_{i=1}^m n_i$ obtained by stacking the vectors x^i one by one for $i = 1, \ldots, m$. The space of all $m \times n$ matrices with real entries is denoted by $\Re^{m\times n}$. We denote by I the identity matrix whose size should be clear from the context. Given any $Z \in \Re^{m\times n}$, we denote its operator norm by $\|Z\|_2$, i.e., $\|Z\|_2 = \max\{\|Zu\|_2 : \|u\|_2 \le 1\}$. We define the condition number of a real nonsingular matrix Z as $\kappa(Z) = \|Z\|_2\|Z^{-1}\|_2$. We let $\mathrm{Tr}(Z)$ denote the trace of a matrix $Z \in \Re^{n\times n}$. By $\mathcal{S}^n$ we denote the space of real $n \times n$ symmetric matrices, and we define

$\mathcal{S}^n_+$ to be the subset of $\mathcal{S}^n$ consisting of the positive semidefinite matrices. We write $Z \succeq 0$ to indicate $Z \in \mathcal{S}^n_+$. Given any $Z \in \mathcal{S}^n$, let $\lambda_i(Z)$ denote its ith largest eigenvalue for $i = 1, ..., n$, and $\lambda_{\min}(Z)$ (resp., $\lambda_{\max}(Z)$) denote its minimal (resp., maximal) eigenvalue.

Given a real Hilbert space U, we denote its inner product by $\langle \cdot, \cdot \rangle_U$, which gives rise to the inner product norm $\| \cdot \|_U$ on U, that is, $\| \cdot \|_U = \sqrt{\langle \cdot, \cdot \rangle_U}$. If V denotes another Hilbert space, and $\mathcal{E} : U \to V$ is a linear operator, the adjoint of $\mathcal{E}$ is the linear operator $\mathcal{E}^* : V \to U$ defined by

$$\langle \mathcal{E}^* v, u \rangle_U = \langle \mathcal{E} u, v \rangle_V, \quad \forall u \in U, \ v \in V.$$

With a slight abuse of notation, we sometimes write $\mathcal{E}$ and $\mathcal{E}^*$ as their representation matrices with respect to some natural (standard) basis. Further, the operator norm of $\mathcal{E}$ is defined as

$$\|\mathcal{E}\| = \max_u \{\|\mathcal{E} u\|_V : \|u\|_U \leq 1\}. \tag{2}$$

Also, if $\mathcal{E}$ is invertible, its condition number is defined as

$$\kappa(\mathcal{E}) = \|\mathcal{E}\| \|\mathcal{E}^{-1}\|. \tag{3}$$

In addition, for an operator $\mathcal{E}$, $\mathrm{Im}(\mathcal{E})$ denotes its range space. We use $\mathcal{I}$ to denote the identity operator.

Let U be a normed vector space whose norm is denoted by $\| \cdot \|$. The dual space of U, denoted by U^*, is the normed vector space consisting of all linear functionals of $u^* : U \to \Re$, endowed with the dual norm $\| \cdot \|^*$ defined as

$$\|u^*\|^* = \max_u \{u^*(u) : \ \|u\| \leq 1\}, \quad \forall u^* \in U^*. \tag{4}$$

Given a closed convex set $\mathcal{C} \subseteq U$ and an arbitrary norm $\| \cdot \|$ on U, let $d_{\mathcal{C}} : U \to \Re$ denote the distance function for $\mathcal{C}$ measured in terms of $\| \cdot \|$, namely,

$$d_{\mathcal{C}}(u) := \inf_{\tilde{u} \in \mathcal{C}} \|u - \tilde{u}\|, \quad \forall u \in U. \tag{5}$$

We further assume that U is a real Hilbert space. A function $f : \Omega \subseteq U \to \Re$ is said to be L-Lipschitz-differentiable with respect to $\| \cdot \|$ if it is differentiable and

$$\|\nabla f(u) - \nabla f(\tilde{u})\|^* \leq L \|u - \tilde{u}\|, \quad \forall u, \tilde{u} \in \Omega.$$

2 A class of CP problems and reformulations

In this section we introduce a class of CP problems of our interest and present four natural primal-dual convex minimization reformulations for them.

Before proceeding, we adopt the same convention as in Golub and Van Loan [20] for counting number of floating point operations (e.g., the inner product of two n-vectors involves $2n$ floating point operations).

Definition 1 *For any map $\psi : P \to Q$ between two sets P and Q, and for each $p \in P$, we denote by* $\mathrm{fps}(\psi(p))$ *the number of floating point operations required to compute $\psi(p) \in Q$. Moreover, we define* $\mathrm{fps}(\psi) = \sup\{\mathrm{fps}(\psi(p)) : p \in P\}$.

We are now ready to introduce a class of CP problems of our interest. Assume X and Y are two real Hilbert spaces. Given a linear operator $\mathcal{A} : X \to Y$, $c \in X$ and $b \in Y$ and a closed convex cone $\mathcal{L} \subseteq X$, consider the CP problem

$$\begin{array}{ll} \min\limits_{x} & \langle c, x\rangle_X \\ \text{s.t.} & \mathcal{A}x = b, \quad x \in \mathcal{L}, \end{array} \tag{6}$$

and its associated dual problem

$$\begin{array}{ll} \max\limits_{(y,s)} & \langle b, y\rangle_Y \\ \text{s.t.} & \mathcal{A}^* y + s = c, \quad s \in \mathcal{L}^*, \end{array} \tag{7}$$

where $\mathcal{A}^*$ is the adjoint operator of $\mathcal{A}$, and $\mathcal{L}^* := \{s \in X : \langle s, x\rangle_X \geq 0, \, \forall x \in \mathcal{L}\}$ is the dual cone of $\mathcal{L}$. We make the following assumptions regarding CP problems (6) and (7) throughout this paper:

A.1 The pair of CP problems (6) and (7) have optimal solutions and their associated duality gap is zero;

A.2 $\mathrm{Im}(\mathcal{A}) = Y$ and $b \neq 0$;

A.3 $\mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1})$ are comparable to $\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)$.

We remark that Assumption A.2 is fairly standard. Indeed, the assumption $\mathrm{Im}(\mathcal{A}) = Y$ is often imposed in the literature. Also, the assumption $b \neq 0$ is very mild. In fact, for the case where $b = 0$, CP problems (6) and (7) become trivial and their solutions can be obtained straightforwardly. In addition, Assumption A.3 plays a crucial role in estimating and comparing the worst-case major arithmetic operation costs of the VNS method for solving CP problems (6) and (7) based on several reformulations (see Section 4). It actually holds for a variety of important CP problems (see the discussion below).

It follows from Definition 1 that $\mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1})$ can be measured by the arithmetic operation cost of solving linear systems:

$$\mathcal{A}\mathcal{A}^* v = h, \quad (\mathcal{I} + \mathcal{A}\mathcal{A}^*)v = h \tag{8}$$

for some $0 \neq h \in Y$, respectively. We now look into two cases where Assumption A.3 holds.

1) $\kappa(\mathcal{A}\mathcal{A}^*)$ is small. Noting that $\kappa(\mathcal{I} + \mathcal{A}\mathcal{A}^*) \leq \kappa(\mathcal{A}\mathcal{A}^*)$, so $\kappa(\mathcal{I} + \mathcal{A}\mathcal{A}^*)$ is also small. Thus, the number of iterations performed by the conjugate gradient (CG) method for solving (8) is expected to be reasonably small. Moreover, we know that the arithmetic operation cost of CG method per iteration is $\mathcal{O}(\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*))$. Therefore, $\mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1})$ are comparable to $\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)$ when CG method is applied to solve (8).

2) $\mathcal{A}\mathcal{A}^*$ is a diagonal operator. In this case, (8) can be trivially solved, and thus $\mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\mathrm{fps}((\mathcal{I}+\mathcal{A}\mathcal{A}^*)^{-1})$ are even much less than $\mathrm{fps}(\mathcal{A})+\mathrm{fps}(\mathcal{A}^*)$.

Combining the above two cases, we can further conclude that when $\mathcal{A}\mathcal{A}^*$ is a block diagonal operator whose each diagonal block is either a diagonal operator or an operator with a small conditional number, $\mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\mathrm{fps}((\mathcal{I}+\mathcal{A}\mathcal{A}^*)^{-1})$ are comparable to $\mathrm{fps}(\mathcal{A})+\mathrm{fps}(\mathcal{A}^*)$. Indeed, there are a variety of CP problems from application whose operator $\mathcal{A}$ possesses such a nice property, and Assumption A.3 thus holds for them. For example, for MAXCUT SDP relaxation [19] and Lovász capacity problem [25], $\mathcal{A}\mathcal{A}^*$ is a diagonal operator. Additionally, several CP problems arising in compressed sensing proposed in [30, 14, 7, 8, 9, 10, 12, 13, 11] also enjoy the aforementioned property (see Section 5).

We next aim to reformulate CP problems (6) and (7) into smooth convex minimization problems which are suitable for first-order methods (e.g., the VNS method [2]).

In view of Assumption A.1, a pair of primal-dual optimal solutions of (6) and (7) can be found by solving the following constrained system of linear equations:

$$\begin{array}{rcll} \mathcal{A}x-b & = & 0, & \\ \mathcal{A}^*y+s-c & = & 0, & (x,s,y)\in\mathcal{L}\times\mathcal{L}^*\times Y. \\ \langle c,x\rangle_X-\langle b,y\rangle_Y & = & 0, & \end{array} \tag{9}$$

Clearly, the primal-dual system (9) can be viewed as a cone linear system:

$$\mathcal{E}u-e=0,\quad u\in\mathcal{K}, \tag{10}$$

where

$$\mathcal{K}=\mathcal{L}\times\mathcal{L}^*\times Y,\quad \mathcal{E}=\begin{pmatrix}\mathcal{A} & 0 & 0\\ 0 & \mathcal{I} & \mathcal{A}^*\\ c & 0 & -b\end{pmatrix},\quad u=\begin{bmatrix}x\\ s\\ y\end{bmatrix},\quad e=\begin{bmatrix}b\\ c\\ 0\end{bmatrix}. \tag{11}$$

We easily observe that $\mathcal{E}$ is a linear operator from U to V, where $U=X\times X\times Y$ and $V=Y\times X\times\Re$.

Our approach to solving (10) (or, equivalently, (9)) will be to reformulate it as a smooth convex minimization problem. To proceed, we define the linear manifold $\mathcal{M}$ as follows:

$$\mathcal{M}=\{u\in U:\mathcal{E}u-e=0\}. \tag{12}$$

Let $d_{\mathcal{K}}(\cdot)$ and $d_{\mathcal{M}}(\cdot)$ denote the distance functions for $\mathcal{K}$ and $\mathcal{M}$ measured in terms of the norm $\|\cdot\|_U$, respectively. We immediately observe that problem (10) can be reformulated into the following minimization problems:

$$\min\{f_1(u):=\|\mathcal{E}u-e\|_V^2:\ u\in\mathcal{K}\}, \tag{13}$$

$$\min\{f_2(u):=(d_{\mathcal{M}}(u))^2+(d_{\mathcal{K}}(u))^2:\ u\in U\}, \tag{14}$$

$$\min\{f_3(u):=(d_{\mathcal{M}}(u))^2:\ u\in\mathcal{K}\}, \tag{15}$$

$$\min\{f_4(u):=(d_{\mathcal{K}}(u))^2:\ u\in\mathcal{M}\}. \tag{16}$$

We remark that reformulation (13) was proposed and studied in [24], in which Nemirovski's prox-method [26], Nesterov's smooth method [27, 28] and its variant [24] were applied to solve it. In addition, (14) has been recently studied in Jarre and Rendl [23], where nonlinear conjugate gradient methods were used to solve it.

Before ending this section, we observe that the objective functions of problems (13)-(16) are convex differentiable and their gradients are Lipschitz continuous, and hence (13)-(16) are smooth convex minimization reformulations of problem (10).

Proposition 1 *The functions $f_1(u)$, $f_2(u)$, $f_3(u)$ and $f_4(u)$ are convex and $2\|\mathcal{E}\|^2$, 4, 2, 2-Lipschitz differentiable with respect to the norm $\|\cdot\|_U$, respectively, where $\|\mathcal{E}\|$ is defined in (2).*

Proof. The conclusion immediately follows from Propositions 1 and 15 of [24]. ∎

In Section 4, we will discuss first-order methods, especially the VNS method [2] for solving CP problems (6) and (7) based on (13)-(16). We also estimate and compare the worst-case major arithmetic operation costs of the VNS method for them.

3 A variant of Nesterov's smooth method

In this section, we review a variant of Nesterov's smooth (VNS) method recently proposed by Auslender and Teboulle [2] for solving a class of smooth convex programming problems.

Let U be a real Hilbert space endowed with a norm $\|\cdot\|$ (not necessarily the inner product norm), and let $\mathcal{U} \subseteq U$ be a closed convex set. Assume that $f : \mathcal{U} \to \Re$ is a differentiable convex function such that for some $L \geq 0$,

$$\|\nabla f(u) - \nabla f(\tilde{u})\|^* \leq L\|u - \tilde{u}\|, \quad \forall u, \tilde{u} \in \mathcal{U}. \tag{17}$$

Our problem of interest in this section is the convex programming problem (1).

We assume throughout our discussion that the optimal value f^* of problem (1) is finite and that its set of optimal solutions is nonempty. Let $h_U : \mathcal{U} \to \Re$ be a differentiable strongly convex function with modulus $\sigma_u > 0$ with respect to $\|\cdot\|_U$, i.e.,

$$h_U(u) \geq h_U(\tilde{u}) + \langle \nabla h_U(\tilde{u}), u - \tilde{u}\rangle + \frac{\sigma_u}{2}\|u - \tilde{u}\|^2, \quad \forall u, \tilde{u} \in \mathcal{U}. \tag{18}$$

The Bregman distance function $d_{h_U} : \mathcal{U} \times \mathcal{U} \to \Re$ associated with h_U is defined as

$$d_{h_U}(u; \tilde{u}) = h_U(u) - l_{h_U}(u; \tilde{u}), \quad \forall u, \tilde{u} \in \mathcal{U}, \tag{19}$$

where $l_{h_U} : U \times \mathcal{U} \to \Re$ is the "linear approximation" of h_U defined as

$$l_{h_U}(u; \tilde{u}) = h_U(\tilde{u}) + \langle \nabla h_U(\tilde{u}), u - \tilde{u}\rangle, \quad \forall (u, \tilde{u}) \in U \times \mathcal{U}.$$

We can similarly define the function l_f by replacing h_U by f in the above identity.

We are now ready to state the VNS method proposed by Auslender and Teboulle [2] for solving (1).

Variant of Nesterov's Smooth (VNS) Method:

0) Let $\bar{u}_0 = \tilde{u}_0 \in \mathcal{U}$ be given and set $k = 0$.

1) Set $u_k = \frac{2}{k+2}\bar{u}_k + \frac{k}{k+2}\tilde{u}_k$ and compute $f(u_k)$ and $\nabla f(u_k)$.

2) Compute $(\tilde{u}_{k+1}, \bar{u}_{k+1}) \in \mathcal{U} \times \mathcal{U}$ as

$$\bar{u}_{k+1} = \operatorname{argmin}\left\{\frac{k+2}{2} l_f(u; u_k) + \frac{L}{\sigma_u} d_{h_U}(u; \bar{u}_k) : u \in \mathcal{U}\right\}, \tag{20}$$

$$\tilde{u}_{k+1} = \frac{2}{k+2}\bar{u}_{k+1} + \frac{k}{k+2}\tilde{u}_k. \tag{21}$$

3) Set $k \leftarrow k+1$ and go to step 1).

end

The main convergence result established by Auslender and Teboulle [2] regarding the above algorithm is summarized in the following theorem (see also Tseng [31]).

Theorem 2 *The sequence $\{\tilde{u}_k\}$ generated by the above VNS method satisfies*

$$f(\tilde{u}_k) - f^* \leq \frac{4L\, d_{h_U}(u^*; \tilde{u}_0)}{\sigma_u (k+1)^2}, \quad \forall k \geq 1, \tag{22}$$

where u^* is an optimal solution of (1).

We observe that the above VNS method requires solving only one (proximal) subproblem per iteration, but Nesterov's smooth algorithm [28] needs to solve two proximal subproblems or one projection subproblem plus one proximal subproblem per iteration. Moreover, it shares the same worst-case iteration complexity with Nesterov's smooth algorithm. Therefore, the VNS method is generally computationally more efficient.

4 VNS method for cone programming

In this section, we discuss the VNS method [2] described in Section 3 for solving CP problems (6) and (7). In particular, the worst-case major arithmetic operation costs of the VNS method when applied to the reformulations (13)-(16) of CP are estimated and compared.

We first present some convergence results of the VNS method when applied to (13)-(16), which are an immediate consequence of Theorem 2.

Proposition 3 *Suppose that Assumption* A.1 *holds. Let* $\{\tilde{u}_k\}$ *be the sequence generated by the VNS method when applied to (13). Given any* $\epsilon > 0$*, an iterate* $\tilde{u}_k \in \mathcal{K}$ *satisfying* $\|\mathcal{E}\tilde{u}_k - e\|_V \leq \epsilon$ *can be found in no more than*

$$\left\lceil \frac{2\sqrt{2}\|\mathcal{E}\|}{\epsilon}\sqrt{\frac{d_{h_U}(u^*;\tilde{u}_0)}{\sigma_u}} \right\rceil$$

iterations, where u^* *is a solution of (10) and* $\|\mathcal{E}\|$ *is defined in (2).*

Proof. Note that $f(u) = \|\mathcal{E}u - e\|_V^2$ for all $u \in \mathcal{K}$, $f^* = 0$, and that any solution u^* of (10) is also an optimal solution of (13). (The existence of u^* follows from Assumption A.1.) Further, in view of Proposition 1, we know that the function f is $2\|\mathcal{E}\|^2$-Lipschitz-differentiable with respect to $\|\cdot\|_U$. Using these facts and Theorem 2, we see that

$$\|\mathcal{E}\tilde{u}_k - e\|_V^2 \leq \frac{8\|\mathcal{E}\|^2\, d_{h_U}(u^*;\tilde{u}_0)}{\sigma_u\,(k+1)^2}, \quad \forall k \geq 1.$$

The conclusion then immediately follows from the above relation. ∎

We can similarly show that the following three propositions hold.

Proposition 4 *Suppose that Assumption* A.1 *holds. Let* $\{\tilde{u}_k\}$ *be the sequence generated by the VNS method when applied to (14). Given any* $\epsilon > 0$*, an iterate* $\tilde{u}_k \in U$ *satisfying* $\sqrt{(d_{\mathcal{M}}(\tilde{u}_k))^2 + (d_{\mathcal{K}}(\tilde{u}_k))^2} \leq \epsilon$ *can be found in no more than*

$$\left\lceil \frac{4}{\epsilon}\sqrt{\frac{d_{h_U}(u^*;\tilde{u}_0)}{\sigma_u}} \right\rceil$$

iterations, where u^* *is a solution of (10).*

Proposition 5 *Suppose that Assumption* A.1 *holds. Let* $\{\tilde{u}_k\}$ *be the sequence generated by the VNS method when applied to (15). Given any* $\epsilon > 0$*, an iterate* $\tilde{u}_k \in \mathcal{K}$ *satisfying* $d_{\mathcal{M}}(\tilde{u}_k) \leq \epsilon$ *can be found in no more than*

$$\left\lceil \frac{2\sqrt{2}}{\epsilon}\sqrt{\frac{d_{h_U}(u^*;\tilde{u}_0)}{\sigma_u}} \right\rceil$$

iterations, where u^* *is a solution of (10).*

Proposition 6 *Suppose that Assumption* A.1 *holds. Let* $\{\tilde{u}_k\}$ *be the sequence generated by the VNS method when applied to (16). Given any* $\epsilon > 0$*, an iterate* $\tilde{u}_k \in \mathcal{M}$ *satisfying* $d_{\mathcal{K}}(\tilde{u}_k) \leq \epsilon$ *can be found in no more than*

$$\left\lceil \frac{2\sqrt{2}}{\epsilon}\sqrt{\frac{d_{h_U}(u^*;\tilde{u}_0)}{\sigma_u}} \right\rceil$$

iterations, where u^* *is a solution of (10).*

From Propositions 3-6, we observe that the ϵ-optimal solutions $\tilde{u}_k$ obtained by the VNS method when applied to (13)-(16) all satisfy

$$u \in U, \quad \sqrt{(d_{\mathcal{M}}(u))^2 + (d_{\mathcal{K}}(u))^2} \leq \epsilon. \tag{23}$$

Thus, with respect to this accuracy criterion, they are all ϵ-optimal solutions of (10). Nevertheless, we notice that the worst-case iteration complexity of the VNS method for (13)-(16) can differ from each other. By estimating and comparing the associated worst-case major arithmetic operation costs, we next explore which one of (13)-(16) is most computationally efficient for the VNS method to find an ϵ-optimal solution of (10) satisfying (23).

To proceed, we specialize the norms $\|\cdot\|_U$ and $\|\cdot\|_V$ on the vector spaces U and V as follows:

$$\|u\|_U = \sqrt{\|u_x\|_X^2 + \|u_s\|_X^2 + \|u_y\|_Y^2}, \quad \forall u = (u_x; u_s; u_y) \in U, \tag{24}$$

$$\|v\|_V = \sqrt{\|v_p\|_Y^2 + \|v_d\|_X^2 + v_o^2}, \quad \forall v = (v_p; v_d; v_o) \in V. \tag{25}$$

And the strongly convex function $h_U(\cdot)$ for the VNS method is chosen as

$$h_U(u) = \frac{1}{2}\|u\|_U^2, \quad \forall u \in U. \tag{26}$$

Given a closed convex set $C \subseteq U$, let $\Pi_C : U \to C$ be the projection map with respect to the norm $\|\cdot\|_U$, that is,

$$\Pi_C(u) = \operatorname{argmin}\{\|u - \tilde{u}\|_U : \ \tilde{u} \in C\}, \quad \forall u \in U. \tag{27}$$

Since $\text{fps}(\mathcal{A})$ and $\text{fps}(\mathcal{A}^*)$ generally much dominate the cost of basic vector operations such as addition, subtraction, scalar multiplication and inner product, we omit for simplicity $\text{fps}(b)$, $\text{fps}(c)$ and other basic vector operation costs from all arithmetic operation costs counted in this section. The following theorem provides an estimate of the worst-case major arithmetic operation costs of the VNS method when applied to (13)-(16).

Theorem 7 *Suppose that Assumption* A.1 *holds. Let $\mathcal{E}$, $\mathcal{K}$ and $\mathcal{M}$ be defined in (11) and (12), respectively. Given any $\epsilon > 0$, the worst-case major arithmetic operation costs of the VNS method when applied to (13)-(16) for finding a pair of ϵ-optimal solutions of CP problems (6) and (7) based on the termination criterion (23) are*

$$\mathcal{C}_1(\epsilon) := [2(\text{fps}(\mathcal{A}) + \text{fps}(\mathcal{A}^*)) + \text{fps}(\Pi_{\mathcal{K}})] \left\lceil \frac{2\sqrt{2}\|\mathcal{E}\| \, \|u^* - \tilde{u}_0^1\|_U}{\epsilon} \right\rceil, \tag{28}$$

$$\mathcal{C}_2(\epsilon) := [\text{fps}(\Pi_{\mathcal{M}}) + \text{fps}(\Pi_{\mathcal{K}})] \left\lceil \frac{4\|u^* - \tilde{u}_0^2\|_U}{\epsilon} \right\rceil, \tag{29}$$

$$\mathcal{C}_3(\epsilon) := [\text{fps}(\Pi_{\mathcal{M}}) + \text{fps}(\Pi_{\mathcal{K}})] \left\lceil \frac{2\sqrt{2}\|u^* - \tilde{u}_0^3\|_U}{\epsilon} \right\rceil, \tag{30}$$

$$\mathcal{C}_4(\epsilon) := [\text{fps}(\Pi_{\mathcal{M}}) + \text{fps}(\Pi_{\mathcal{K}})] \left\lceil \frac{2\sqrt{2}\|u^* - \tilde{u}_0^4\|_U}{\epsilon} \right\rceil, \tag{31}$$

respectively, where u^* is an optimal solution of CP problems (6) and (7), and $\tilde{u}_0^1 \in \mathcal{K}$, $\tilde{u}_0^2 \in U$, $\tilde{u}_0^3 \in \mathcal{K}$ and $\tilde{u}_0^4 \in \mathcal{M}$ are the initial points of the VNS method for (13)-(16), respectively.

Proof. In view of (13)-(16), (5), (27) and Proposition 15 of [24], we have

$$\begin{array}{ll} \nabla f_1(u) = 2\mathcal{E}^*(\mathcal{E}u - e), & \nabla f_2(u) = 2(2u - \Pi_{\mathcal{M}}(u) - \Pi_{\mathcal{K}}(u)), \\ \nabla f_3(u) = 2(u - \Pi_{\mathcal{M}}(u)), & \nabla f_4(u) = 2(u - \Pi_{\mathcal{K}}(u)). \end{array}$$

Using (26) and the first relation above, we observe that when applied to (13), the worst-case major arithmetic operation cost per iteration of the VNS method includes: $2(\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*))$ for computing $\nabla f_1(\cdot)$ in step 1); and $\mathrm{fps}(\Pi_{\mathcal{K}})$ for solving the proximal subproblem in step 2) with $h_U(\cdot)$ given in (26). Together with Proposition 3, we see that the worst-case major arithmetic operation cost of the VNS method when applied to (13) for finding a pair of ϵ-optimal solutions of CP problems (6) and (7) is given by (28). Similarly, we can show the remaining conclusions hold. ∎

As observed in our computational experiment, the actual number of iterations performed by the VNS method is generally proportional to its worst-case iteration complexity given in Theorem 2. Thus, the worst-case major arithmetic operation costs estimated in Theorem 7 can be used to compare the actual performance of the VNS method for solving (13)-(16). In order to compare $\mathcal{C}_1(\epsilon)$, $\mathcal{C}_2(\epsilon)$, $\mathcal{C}_3(\epsilon)$ and $\mathcal{C}_4(\epsilon)$, we assume for the remainder of this section that

$$\tilde{u}_0^1 = \tilde{u}_0^3 = \Pi_{\mathcal{K}}(\tilde{u}_0^2), \qquad \tilde{u}_0^4 = \Pi_{\mathcal{M}}(\tilde{u}_0^1) \tag{32}$$

for some $\tilde{u}_0^2 \in U$. These together with $u^* \in \mathcal{K} \cap \mathcal{M}$ immediately imply that

$$\|u^* - \tilde{u}_0^4\|_U \;\le\; \|u^* - \tilde{u}_0^3\|_U \;=\; \|u^* - \tilde{u}_0^1\|_U \;\le\; \|u^* - \tilde{u}_0^2\|_U. \tag{33}$$

In view of this relation, (29) and (30), we conclude that $\mathcal{C}_4(\epsilon) \le \mathcal{C}_3(\epsilon) \le \mathcal{C}_2(\epsilon)$, and hence it is generally more efficient to solve (16) than (14) and (15) by the VNS method for finding a pair of ϵ-optimal solutions of CP problems (6) and (7) based on the termination criterion (23).

For the rest of this section, our aim is to compare $\mathcal{C}_4(\epsilon)$ with $\mathcal{C}_1(\epsilon)$. To proceed, we now establish some useful properties for the operators $\mathcal{A}\mathcal{A}^*$ and $\mathcal{E}\mathcal{E}^*$.

Proposition 8 *Let $\mathcal{E}$ be defined in (11). Under Assumption* A.2*, the following statements hold:*

i) The operator $\mathcal{A}\mathcal{A}^*$ is invertible;

ii) The operator $\mathcal{E}\mathcal{E}^*$ is invertible, and moreover,

$$(\mathcal{E}\mathcal{E}^*)^{-1} = \mathcal{H}^*\mathcal{G}\mathcal{H}, \tag{34}$$

where

$$\mathcal{H} = \begin{pmatrix} \mathcal{I} & 0 & 0 \\ 0 & \mathcal{I} & 0 \\ -c\mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1} & b\mathcal{A}(I+\mathcal{A}^*\mathcal{A})^{-1} & \mathcal{I} \end{pmatrix}, \tag{35}$$

$$\mathcal{G} = \begin{pmatrix} (\mathcal{A}\mathcal{A}^*)^{-1} & 0 & 0 \\ 0 & (\mathcal{I}+\mathcal{A}^*\mathcal{A})^{-1} & 0 \\ 0 & 0 & \xi^{-1} \end{pmatrix}, \tag{36}$$

$$\xi = cc^* + bb^* - c\mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}c^* - b\mathcal{A}(I+\mathcal{A}^*\mathcal{A})^{-1}\mathcal{A}^*b^*. \tag{37}$$

Proof. Let $y \in Y$ be such that $\mathcal{A}\mathcal{A}^*y = 0$. Then we have

$$\|\mathcal{A}^*y\|_X^2 = \langle \mathcal{A}^*y,\ \mathcal{A}^*y\rangle_X = \langle y,\ \mathcal{A}\mathcal{A}^*y\rangle_Y = 0,$$

and hence $\mathcal{A}^*y = 0$. It leads to

$$\langle \mathcal{A}x,\ y\rangle_Y = \langle x,\ \mathcal{A}^*y\rangle_X = 0, \quad \forall x \in X,$$

which together with $\mathrm{Im}(\mathcal{A}) = Y$ (see Assumption A.2), implies that $\langle y,\ y\rangle_Y = 0$, and so $y = 0$. It follows that $\mathcal{A}\mathcal{A}^*$ is injective. Now let $\{y^i\}_{i=1}^m$ be a basis for the space Y. As shown above, $\mathcal{A}\mathcal{A}^*(\sum_{i=1}^m \alpha_i y^i) = 0$ for some $\{\alpha_i\}_{i=1}^m$ yields $\sum_{i=1}^m \alpha_i y^i = 0$. The latter relation implies $\alpha_i = 0$ for $i = 1, \ldots, m$. Thus, we immediately see that $\{\mathcal{A}\mathcal{A}^*y^i\}_{i=1}^m$ is linearly independent, and $\{\mathcal{A}\mathcal{A}^*y^i\}_{i=1}^m$ is a basis for Y, which leads to $\mathcal{A}\mathcal{A}^*Y = Y$, and hence $\mathcal{A}\mathcal{A}^*$ is surjective. The statement i) immediately follows.

In view of the definition of $\mathcal{E}$ (see (11)), we obtain that

$$\mathcal{E}u = \begin{pmatrix} \mathcal{A}u_x \\ u_s + \mathcal{A}^*u_y \\ cu_x - bu_y \end{pmatrix}, \quad \forall u = \begin{pmatrix} u_x \\ u_s \\ u_y \end{pmatrix} \in U.$$

The above relation together with Assumption A.2 immediately implies that $\mathrm{Im}(\mathcal{E}) = V$. Using this result and following a similar proof as for statement i), we conclude that $\mathcal{E}\mathcal{E}^*$ is invertible.

Further, we see from (11) that

$$\mathcal{E}^* = \begin{pmatrix} \mathcal{A}^* & 0 & c^* \\ 0 & \mathcal{I} & 0 \\ 0 & \mathcal{A} & -b^* \end{pmatrix}, \quad \mathcal{E}\mathcal{E}^* = \begin{pmatrix} \mathcal{A}\mathcal{A}^* & 0 & \mathcal{A}c^* \\ 0 & \mathcal{I}+\mathcal{A}^*\mathcal{A} & -\mathcal{A}^*b^* \\ c\mathcal{A}^* & -b\mathcal{A} & cc^*+bb^* \end{pmatrix}. \tag{38}$$

In view of this relation and (35), it is easy to verify that

$$\mathcal{H}\mathcal{E}\mathcal{E}^*\mathcal{H}^* = \begin{pmatrix} \mathcal{A}\mathcal{A}^* & 0 & 0 \\ 0 & \mathcal{I}+\mathcal{A}^*\mathcal{A} & 0 \\ 0 & 0 & \xi \end{pmatrix}. \tag{39}$$

Noticing that $\mathcal{H}$, $\mathcal{H}^*$ and $\mathcal{E}\mathcal{E}^*$ are invertible, we conclude from (39) that ξ^{-1} exists. Upon taking inverse on both sides of (39), we immediately see that (34) holds. ∎

To compare $\mathcal{C}_4(\epsilon)$ with $\mathcal{C}_1(\epsilon)$, it is clear from (28) and (31) that we need to relate $\mathrm{fps}(\Pi_{\mathcal{M}})$ with $\mathrm{fps}(\mathcal{A})$ and $\mathrm{fps}(\mathcal{A}^*)$. We now aim to build a relation between them. In view of (27), (12) and (24), we can easily show that

$$\Pi_{\mathcal{M}}(u) = u + \mathcal{E}^*(\mathcal{E}\mathcal{E}^*)^{-1}(e - \mathcal{E}u),$$

which together with the definition of $\mathcal{E}$ implies that

$$\mathrm{fps}(\Pi_{\mathcal{M}}) = 2\,(\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)) + \mathrm{fps}((\mathcal{E}\mathcal{E}^*)^{-1}). \tag{40}$$

We next need to estimate $\mathrm{fps}((\mathcal{E}\mathcal{E}^*)^{-1})$. First, we note from (37) that ξ only depends on $\mathcal{A}$, b, c and their adjoint operators, and thus it only needs to be computed once. Evidently, the major work for evaluating ξ is to compute:

$$\tau := (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}c^*, \quad \delta := (\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}\mathcal{A}^*b^*, \tag{41}$$

$\mathcal{A}^*\tau$ and $\mathcal{A}\delta$. It is easy to observe that the major arithmetic operation cost for computing ξ, τ and δ is

$$2\,(\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)) + \mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1}) + \mathrm{fps}((\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}).$$

From now on, we assume that ξ, τ and δ are computed beforehand as above. We are now ready to estimate $\mathrm{fps}((\mathcal{E}\mathcal{E}^*)^{-1})$ as follows.

Lemma 9 *Under Assumption* A.2*, the following holds:*

$$\mathrm{fps}((\mathcal{E}\mathcal{E}^*)^{-1}) = \mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*) + \mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1}) + \mathrm{fps}((\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}). \tag{42}$$

Proof. In view of (34), we observe that for any $v = (v_p; v_d; v_o) \in V$, $z = (\mathcal{E}\mathcal{E}^*)^{-1}v$ can be computed according to the following steps: 1) compute $h = \mathcal{H}v$; 2) compute $w = \mathcal{G}h$; and 3) compute $z = \mathcal{H}^*w$. We now analyze their main computation in details. For step 1), using (35) and the relation $h = \mathcal{H}v$, we obtain $h = (v_p; v_d; h_o)$, where

$$h_o = -c\mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}v_p + b\mathcal{A}(\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}v_d + v_o.$$

We clearly see that the main computation for step 1) is to evaluate:

$$\varrho := (\mathcal{A}\mathcal{A}^*)^{-1}v_p, \quad \vartheta := (\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}v_d, \tag{43}$$

$\mathcal{A}^*\varrho$ and $\mathcal{A}\vartheta$. Hence, the worst-case major arithmetic operation cost for step 1) is

$$\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*) + \mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1}) + \mathrm{fps}((\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}).$$

For step 2), using (36), (43) and the relations $h = (v_p; v_d; h_o)$ and $w = \mathcal{G}h$, we obtain $w = (w_p; w_d; w_o)$, where

$$w_p = (\mathcal{A}\mathcal{A}^*)^{-1}v_p = \varrho, \quad w_d = (\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}v_d = \vartheta, \quad w_o = \xi^{-1}h_o.$$

Since ϱ, ϑ and h_o are already computed in step 1), w is readily available and so almost there is no additional computational cost for step 2). Finally, for step 3), using (35), (41) and the relation $z = \mathcal{H}^* w$, we obtain $z = (z_p; z_d; w_o)$, where

$$\begin{aligned} z_p &= w_p - (\mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A}c^* w_o = w_p - \tau w_o, \\ z_d &= w_d + (\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}\mathcal{A}^* b^* w_o = w_d + \delta w_o. \end{aligned}$$

Noticing that w_p, w_d and w_o are already obtained in step 2), so almost there is no extra computational cost for step 3). Summing up all major computational costs for steps 1)-3), we immediately see the conclusion holds. ∎

The following result provides an estimate of $\mathrm{fps}((\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1})$.

Lemma 10 *The following holds:*

$$\mathrm{fps}((\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1}) = \mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*) + \mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1}). \tag{44}$$

Proof. By applying Sherman-Morrison-Woodbury formula in the context of operators, we have

$$(\mathcal{I} + \mathcal{A}^*\mathcal{A})^{-1} = \mathcal{I} - \mathcal{A}^*(\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1}\mathcal{A},$$

which immediately leads to the conclusion. ∎

In view of (40), (42) and (44), we now obtain an estimate of $\mathrm{fps}(\Pi_{\mathcal{M}})$ in terms of $\mathcal{A}$ and $\mathcal{A}^*$ as follows.

Lemma 11 *Under Assumption* A.2*, the following holds:*

$$\mathrm{fps}(\Pi_{\mathcal{M}}) = 4(\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)) + \mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1}) + \mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1}). \tag{45}$$

Using (31) and (45), we finally obtain an estimate of $\mathcal{C}_4(\epsilon)$ as follows.

Theorem 12 *Let $\mathcal{C}_4(\epsilon)$ be defined in (31). Under Assumption* A.2*, the following holds:*

$$\mathcal{C}_4(\epsilon) = [4(\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)) + \mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1}) + \mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1}) + \mathrm{fps}(\Pi_{\mathcal{K}})] \left\lceil \frac{2\sqrt{2}\|u^* - \tilde{u}_0^4\|_U}{\epsilon} \right\rceil \tag{46}$$

for some $\tilde{u}_0^4 \in \mathcal{M}$.

We are now ready to compare $\mathcal{C}_1(\epsilon)$ and $\mathcal{C}_4(\epsilon)$ that are estimated in (28) and (46), respectively. First, by the choice of the initial points specified in (32), we know from (33) that $\|u^* - \tilde{u}_0^4\|_U \leq \|u^* - \tilde{u}_0^1\|_U$. In addition, using the second identity of (38) and the relation $\|\mathcal{A}\| = \|\mathcal{A}^*\|$, it is not hard to see that

$$\|\mathcal{E}\| = \sqrt{\|\mathcal{E}\mathcal{E}^*\|} \;\geq\; \sqrt{\max\left(\|\mathcal{I} + \mathcal{A}^*\mathcal{A}\|, \|cc^* + bb^*\|\right)} \;\geq\; \max\left(\sqrt{1 + \|\mathcal{A}\|^2}, \|b\|, \|c\|\right).$$

Thus, $\|\mathcal{E}\|$ is bounded below by the quantity on the right-hand side above, which can be much larger than one. Using this observation and Assumption A.3, we conclude from (28) and (46) that $\mathcal{C}_4(\epsilon)$ can be much less than $\mathcal{C}_1(\epsilon)$.

In summary, assuming the initial points are chosen according to (32), the VNS method [2] is generally more efficient when applied to (16) than (13)-(15) for finding a pair of ϵ-optimal solutions of CP problems (6) and (7). We shall mention that for similar reasons, this conclusion also holds for the other first-order methods, for example, Nesterov's smooth method [28] and another variant of Nesterov's smooth method proposed by Tseng [31].

5 Computational results

In this section, we justify our theoretical prediction on the behavior of the VNS method when applied to formulations (13)-(16) by conducting numerical experiments on several CP problems, that is, Dantzig selector, basis pursuit de-noising, MAXCUT SDP relaxation and Lovász capacity problems on a set of randomly generated instances.

The above CP problems are either in the form of (6) or (7) of which X and Y are finite dimensional vector spaces. We set their inner products to the standard ones, and set all norms used by the VNS method to the inner product norms. In addition, we choose $\tilde{u}_0^1 = \tilde{u}_0^2 = \tilde{u}_0^3 := (x_0; s_0; y_0) = (0; 0; 0)$ and $\tilde{u}_0^4 = \Pi_{\mathcal{M}}(\tilde{u}_0^1)$ as the initial points for the VNS method when applied to (13)-(16), respectively. For convenience of presentation, we refer to the VNS method for (13)-(16) as VNS1, VNS2, VNS3 and VNS4, respectively. The following termination criterion is used by the VNS method to solve CP problems (6) and (7). Given a tolerance $\epsilon > 0$, an approximate primal-dual solution $(x, s, y) \in \mathcal{L} \times \mathcal{L}^* \times Y$ is found such that

$$\max\{\|\mathcal{A}^* y + s - c\|_X,\ \|\mathcal{A}x - b\|_Y,\ |\langle c, x\rangle - \langle b, y\rangle|\} \leq \epsilon. \tag{47}$$

Furthermore, suppose that $\tilde{u}^k$ is an approximate solution obtained at the kth iteration by these four approaches. $\tilde{u}^k$ is then used to check the termination criterion (47) for VNS1 and VNS3 while $u_k = \Pi_{\mathcal{K}}(\tilde{u}^k)$ is used for VNS2 and VNS4. The codes for the VNS method are written in Matlab, which are available online at www.math.sfu.ca/∼zhaosong. All computations in this section are performed on an Intel Xeon 5410 CPU (2.33GHz) and 8GB RAM running Red Hat Enterprise Linux 4 (kernel 2.6.18).

5.1 Dantzig selector problem

In this subsection, we aim to compare the performance of the VNS method for solving the Dantzig selector (DS) problem when applied to formulations (13)-(16).

The DS problem is a model recently proposed by Candès and Tao [13] for recovering large sparse signal using a relatively small number of linear measurements or observations. Given the data consisting of a matrix $A \in \Re^{m\times n}$ and a vector $\mathtt{b}$, the DS problem is in the form of

$$\min_x\{\|x\|_1 :\ \|A^T(Ax - \mathtt{b})\|_\infty \leq \lambda\}, \tag{48}$$

where λ is a nonnegative parameter for controlling the residual. This model is capable of producing a sparse vector x^* such that the residual $Ax^* - \mathtt{b}$ is not too correlated with any of the columns of A. Recently, the DS problem has also found numerous applications in variable selection and model fitting in the context of linear regression, where it performs especially well when the number of predictors is large relative to the number of observations (see [5] and the references therein).

In the context of compressed sensing, the data matrix A is usually either a Gaussian random matrix (that is, its elements are independently drawn from the standard normal distribution) or has all rows randomly chosen from an orthonormal matrix such as the fast Fourier transform (FFT), discrete cosine transform (DCT) or wavelets transform matrix. It is often large-scale, fully dense and has full row rank. Moreover, $\kappa(AA^T)$ is usually small. Indeed, when A is a Gaussian random matrix, it follows from a well-known random matrix theory (see, for example, [16]) that A has full row rank with probability one and

$$n\left(1-\sqrt{\frac{m}{n}}\right)^2 \le \lambda_i(AA^T) \le n\left(1+\sqrt{\frac{m}{n}}\right)^2, \quad i=1,\ldots,m.$$

Hence,

$$\kappa(AA^T) = \frac{\lambda_{\max}(AA^T)}{\lambda_{\min}(AA^T)} \le \left(\frac{1+\sqrt{m/n}}{1-\sqrt{m/n}}\right)^2$$

with prevailing probability for large n. Noticing that $m \ll n$ usually holds, $\kappa(AA^T)$ is thus often nearly one. On the other hand, when A is a partial orthonormal matrix mentioned above, we have $AA^T = I$ and hence $\kappa(AA^T) = 1$.

We next apply the VNS method to solve problem (48). In particular, we first reformulate it into a CP problem in the form of (7) and then apply the VNS method to solve the latter problem based on formulations (13)-(16). It is easy to observe that problem (48) can be reformulated as

$$\begin{array}{lrcrcl} \max\limits_{x^+,x^-} & -\mathbf{1}^T x^+ & - & \mathbf{1}^T x^- & & \\ \text{s.t.} & A^TAx^+ & - & A^TAx^- & \le & \lambda\mathbf{1} + A^T\mathtt{b}, \\ & -A^TAx^+ & + & A^TAx^- & \le & \lambda\mathbf{1} - A^T\mathtt{b}, \\ & -x^+ & & & \le & 0, \\ & & & -x^- & \le & 0. \end{array} \tag{49}$$

Clearly, (49) is a CP problem in the form of (7) with $\mathcal{L} = \Re^{4n}_+$, $X = \Re^{4n}$, $Y = \Re^{2n}$, and

$$\mathcal{A} = \begin{bmatrix} A^TA & -A^TA & -I & 0 \\ -A^TA & A^TA & 0 & -I \end{bmatrix}, \qquad b = \begin{pmatrix} -\mathbf{1} \\ -\mathbf{1} \end{pmatrix}, \quad c = (\lambda\mathbf{1} + A^T\mathtt{b}; \lambda\mathbf{1} - A^T\mathtt{b}; 0; 0).$$

We now show that CP problem (49) satisfies Assumption A.3 by exploring how the linear systems (8) can be efficiently solved. It is clear to see that

$$\mathcal{A}\mathcal{A}^* = I + \begin{bmatrix} A^TA & -A^TA \\ -A^TA & A^TA \end{bmatrix}^2 = I + 2\begin{bmatrix} A^TAA^T \\ -A^TAA^T \end{bmatrix}[A \quad -A].$$

By applying Sherman-Morrison-Woodbury formula to the above identity, we obtain that

$$\begin{aligned}(\mathcal{A}\mathcal{A}^*)^{-1} &= I - 2\begin{bmatrix} A^TAA^T \\ -A^TAA^T \end{bmatrix}\left(I + 2[A \quad -A]\begin{bmatrix} A^TAA^T \\ -A^TAA^T \end{bmatrix}\right)^{-1}[A \quad -A], \\ &= I - 2\begin{bmatrix} A^TAA^T \\ -A^TAA^T \end{bmatrix}\left(I + 4(AA^T)^2\right)^{-1}[A \quad -A]. \end{aligned} \tag{50}$$

Similarly, we have

$$(\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1} = \frac{1}{2}I - \frac{1}{2}\begin{bmatrix} A^TAA^T \\ -A^TAA^T \end{bmatrix}\left(I + 2(AA^T)^2\right)^{-1}[A \quad -A]. \tag{51}$$

When the matrix A has all rows randomly chosen from an orthonormal matrix such as FFT, DCT or wavelets transform matrix, we have $AA^T = I$. Using this relation, (50) and (51), we see that

$$\begin{aligned}(\mathcal{A}\mathcal{A}^*)^{-1} &= I - \frac{2}{5}\begin{bmatrix} A^T \\ -A^T \end{bmatrix}[A \quad -A], \\ (\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1} &= \frac{1}{2}I - \frac{1}{6}\begin{bmatrix} A^T \\ -A^T \end{bmatrix}[A \quad -A]. \end{aligned}$$

It follows that the linear systems (8) can be trivially solved and $\mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1})$ are comparable to $\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)$. When A is a Gaussian random matrix, we know from above that $\kappa(AA^T)$ is often nearly one, and so are $\kappa(I + 4(AA^T)^2)$ and $\kappa(I + 2(AA^T)^2)$. Using this fact, (50) and (51), we observe that the linear systems (8) can be reduced to the ones with coefficient matrices $I + 4(AA^T)^2$ and $I + 2(AA^T)^2$, and the latter linear systems can be efficiently solved by CG method whose number of iterations is reasonably small. Additionally, the arithmetic operation cost of CG method per iteration is $\mathcal{O}(\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*))$. Hence, $\mathrm{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\mathrm{fps}((\mathcal{I} + \mathcal{A}\mathcal{A}^*)^{-1})$ are comparable to $\mathrm{fps}(\mathcal{A}) + \mathrm{fps}(\mathcal{A}^*)$. We thus conclude that CP problem (49) satisfies Assumption A.3. In addition, it evidently satisfies Assumptions A.1 and A.2.

We next compare the performance of the VNS method for solving DS problem (48) (or, equivalently, (49)) when applied to formulations (13)-(16). All instances for DS problem (48) are randomly generated in the same manner as described in l_1-magic [7]. In particular, given $\sigma > 0$ and positive integers m, n, T with $m < n$ and $T < n$, we first generate a matrix $W \in \Re^{n\times m}$ with entries randomly chosen from a normal distribution with mean zero, variance one and standard deviation one. Then we compute an orthonormal basis, denoted by B, for the range space of W, and set $A = B^T$. We also randomly generate a vector $\tilde{x} \in \Re^n$ with only T nonzero components that are ± 1, and generate a vector $v \in \Re^m$ with entries randomly chosen from a normal distribution with mean zero, variance one and standard deviation one. Finally, set $\mathtt{b} = A\tilde{x} + \sigma v$. Especially, we choose $\sigma = 0.005$ for all instances.

As in [7, 18], we set the noise level $\lambda = 3e-3$ for DS problem (48). The termination criterion (47) with $\epsilon = 0.1$ is used for VNS1-VNS4. The performance of these methods for the above randomly generated instances is presented in Table 1. In detail, the parameters m, n and T

Table 1: Comparison of VNS1–VNS4 for DS problem

Problem			Iteration				Time			
m	n	T	VNS1	VNS2	VNS3	VNS4	VNS1	VNS2	VNS3	VNS4
120	512	20	3958	227	121	109	4.4	0.5	0.3	0.2
240	1024	40	6570	234	142	113	14.3	1.0	0.7	0.5
360	1536	60	9712	291	172	139	46.1	3.2	2.1	1.4
480	2048	80	12650	337	194	160	212.9	13.7	9.4	5.9
600	2560	100	14499	333	199	160	410.0	23.1	16.5	10.1
720	3072	120	18282	387	229	184	724.5	37.7	26.6	16.3
840	3584	140	18477	359	214	168	1002.6	47.8	34.1	20.3
960	4096	160	20669	356	224	168	1455.1	61.7	46.4	26.4
1080	4608	180	21827	363	222	169	1872.9	77.3	56.4	32.5
1200	5120	200	25621	402	248	189	2731.4	105.9	78.0	44.9

of each instance are listed in columns one to three, respectively. The number of iterations of VNS1-VNS4 is given in columns four to seven, and CPU times (in seconds) are given in the last four columns, respectively. We observe from Table 1 that VNS4 substantially outperforms the other three approaches for solving (49) (or, equivalently, (48)).

Since DS problem (48) can be formulated as an LP, the well-known methods such as simplex and IP methods are suitable for solving it. Recently, Candès and Romberg [7] implemented an IP method for this problem based on reformulation (49). Subsequently, Friedlander and Saunders [18] applied CPLEX dual simplex solver [15] and the IP solvers such as PDCO [29] and CPLEX barrier IP solver to several LP reformulations of (48). The computational results reported in [18] demonstrate that the IP method [7] generally outperforms the other methods. In addition, we already see from above that VNS4 typically outperforms VNS1-VNS3. In next experiment we aim to compare the performance of VNS4 with the IP method [7] (labeled as IP) on the instances randomly generated in the same manner as above.

We choose the initial point for VNS4 as the one mentioned in the beginning of Section 5 while the initial point for the IP method [7] is chosen by default. The termination criterion (47) with $\epsilon = 0.1$ is used for both methods. The codes for both methods are written in Matlab. In addition, the IP method [7] uses the conjugate gradient method to approximately solve the associated Newton systems. The performance of these methods on the randomly generated instances are presented in Tables 2-4 for noise levels $\lambda = 3e-2$, $3e-3$ and $3e-4$, respectively. (It shall be mentioned that $\lambda = 3e-3$ is the default noise level used in the IP method [7].) In each of these tables, the parameters m, n and T of the instances are listed in columns one to three, respectively. The numbers of iterations of VNS4 and IP are given in columns four to five, and CPU times (in seconds) are given in the last two columns, respectively. From Tables 2-4, we conclude that when low-accuracy solutions are sought, the method VNS4, that is, the VNS method applied to formulation (16) substantially outperforms the IP method [7] for solving problem (49) (or, equivalently, (48)). We shall also mention that the former method requires much less memory than the latter one. In addition, as the noise level λ decreases, the performance of IP method [7] stays almost same, but the performance of VNS4 clearly improves in terms of

Table 2: Comparison of VNS4 and IP for DS problem with $\lambda = 3e - 2$

Problem			Iteration		Time	
m	n	T	VNS4	IP	VNS4	IP
1560	6656	260	329	26	129.7	2322.2
1680	7168	280	339	26	153.3	2836.4
1800	7680	300	356	27	188.2	3596.0
1920	8192	320	365	27	222.1	4298.4
2040	8704	340	394	27	269.7	5107.6
2160	9216	360	397	26	292.6	5780.6
2280	9728	380	388	27	318.0	7001.3
2400	10240	400	432	27	393.9	8091.3
3600	15360	600	531	31	1100.9	30279.3
4800	20480	800	626	31	2299.3	71384.1

Table 3: Comparison of VNS4 and IP for DS problem with $\lambda = 3e - 3$

Problem			Iteration		Time	
m	n	T	VNS4	IP	VNS4	IP
1560	6656	260	198	23	78.7	2019.2
1680	7168	280	197	22	89.6	2350.5
1800	7680	300	210	23	111.4	3018.0
1920	8192	320	215	24	131.9	3751.8
2040	8704	340	219	23	150.6	4286.9
2160	9216	360	217	24	160.8	5231.4
2280	9728	380	213	24	175.4	6123.6
2400	10240	400	227	25	208.3	7385.8
3600	15360	600	246	26	516.6	25244.5
4800	20480	800	282	27	1046.4	62164.1

both CPU times and number of iterations. A possible interpretation for the latter phenomenon is that, as λ decreases, the distance between the initial point specifically chosen above and the optimal solution set of the corresponding CP problems gets smaller, and it thus follows from Theorem 7 that the computational cost for finding an ϵ-optimal solution becomes cheaper.

5.2 Basis pursuit de-noising problem

In this subsection, we aim to compare the performance of the VNS method for solving the basis pursuit de-noising (BPDN) problem when applied to formulations (13)-(16).

The BPDN problem is a model proposed by Chen et al. [14] for recovering large-scale sparse signal from highly incomplete information. Given the data consisting of a matrix $A \in \Re^{m\times n}$ and a vector $\mathtt{b}$, the BPDN problem is in the form of

$$\min_x \lambda\|x\|_1 + \frac{1}{2}\|Ax - \mathtt{b}\|_2^2, \tag{52}$$

Table 4: Comparison of VNS4 and IP for DS problem with $\lambda = 3e - 4$

Problem			Iteration		Time	
m	n	T	VNS4	IP	VNS4	IP
1560	6656	260	193	23	76.6	2015.2
1680	7168	280	192	22	87.1	2354.3
1800	7680	300	203	23	107.6	3018.9
1920	8192	320	206	25	125.9	3898.4
2040	8704	340	212	23	145.8	4271.2
2160	9216	360	208	23	153.3	5029.2
2280	9728	380	208	24	170.9	6126.9
2400	10240	400	219	24	200.5	7086.7
3600	15360	600	238	26	496.1	25123.4
4800	20480	800	258	28	954.0	64246.4

where λ is a nonnegative parameter for controlling the sparsity of solutions.

We observe that problem (52) can be reformulated as

$$\begin{array}{ll} \min\limits_{x^+,x^-,t_1,t_2,u} & \lambda \mathbf{1}^T x^+ + \lambda \mathbf{1}^T x^- + 2t_1 \\ \text{s.t.} & t_1 - t_2 \;=\; 2, \\ & Ax^+ - Ax^- - u \;=\; \mathtt{b}, \\ & x^+ \geq 0,\; x^- \geq 0, \quad (t_1; t_2; u) \in \mathcal{Q}^{m+2}. \end{array} \tag{53}$$

Clearly, (53) is a CP problem in the form of (6) with $\mathcal{L} = \Re^{2n}_+ \times \mathcal{Q}^{m+2}$, $X = \Re^{m+2n+2}$, $Y = \Re^{m+1}$, and

$$\mathcal{A} = \begin{bmatrix} 0 & 0 & 1 & -1 & 0 \\ A & -A & 0 & 0 & -I \end{bmatrix}, \quad b = \begin{pmatrix} 2 \\ \mathtt{b} \end{pmatrix}, \quad c = (\lambda \mathbf{1}; \lambda \mathbf{1}; 2; 0; 0).$$

Note that $\mathcal{K} = \mathcal{L} \times \mathcal{L}^* \times Y = \Re^{2n}_+ \times \mathcal{Q}^{m+2} \times \Re^{2n}_+ \times \mathcal{Q}^{m+2} \times \Re^{m+1}$. Hence, the projection of a point into $\mathcal{K}$ can be reduced to the projection of a point into $\mathcal{Q}^{m+2}$, which can be cheaply computed according to Fact 2.3 of [3]. We also see that

$$\mathcal{A}\mathcal{A}^* = \begin{bmatrix} 2 & 0 \\ 0 & I + 2AA^T \end{bmatrix}.$$

Thus, when the matrix A has all rows randomly chosen from an orthonormal matrix such as FFT, DCT or wavelets transform matrix, $\mathcal{A}\mathcal{A}^*$ is a diagonal operator. On the other hand, when A is a Gaussian random matrix, we know from Subsection 5.1 that $\kappa(AA^T)$ is often nearly one and so $\mathcal{A}\mathcal{A}^*$ is a block diagonal operator consisting of one diagonal block and another block with a small conditional number. Therefore, the linear systems (8) can often be either trivially solved by direct method or efficiently solved by CG method whose number of iterations is reasonably small. Additionally, the arithmetic operation cost of CG method per iteration is $\mathcal{O}(\text{fps}(\mathcal{A}) + \text{fps}(\mathcal{A}^*))$. Hence, $\text{fps}((\mathcal{A}\mathcal{A}^*)^{-1})$ and $\text{fps}((\mathcal{I}+\mathcal{A}\mathcal{A}^*)^{-1})$ are comparable to $\text{fps}(\mathcal{A})+\text{fps}(\mathcal{A}^*)$. We thus conclude that CP problem (53) satisfies Assumption A.3. In addition, it evidently satisfies Assumptions A.1 and A.2.

Table 5: Comparison of VNS1–VNS4 for BPDN problem

Problem			Iteration				Time			
m	n	T	VNS1	VNS2	VNS3	VNS4	VNS1	VNS2	VNS3	VNS4
1320	5632	220	236	51	29	25	10.9	6.6	2.6	2.3
1440	6144	240	252	53	30	26	13.8	8.0	3.1	2.5
1560	6656	260	267	54	31	26	16.8	9.2	3.8	3.0
1680	7168	280	298	57	33	28	21.7	11.5	4.6	3.7
1800	7680	300	311	58	34	28	25.8	13.2	5.4	4.2
1920	8192	320	318	59	34	29	29.6	15.8	6.2	4.9
2040	8704	340	342	61	36	30	36.2	17.2	7.4	5.7
2160	9216	360	366	63	37	31	43.0	21.0	8.5	6.5
2280	9728	380	411	66	40	33	53.3	22.7	10.2	7.7
2400	10240	400	442	69	42	34	62.3	26.5	11.8	8.7

We next compare the performance of the VNS method for solving BPDN problem (52) (or, equivalently, (53)) when applied to formulations (13)-(16). All instances for BPDN problem (48) are randomly generated in the same manner as described in Subsection 5.1. As in [18], we set $\lambda = 3e-3$ for BPDN problem (52). The termination criterion (47) with $\epsilon = 0.1$ is used for VNS1-VNS4. The performance of these methods is presented in Table 5, whose columns have similar meaning as those of Table 1. We observe from Table 5 that VNS4 substantially outperforms the other three approaches for solving (53) (or, equivalently, (52)).

In literature, there are numerous efficient methods for solving the BPDN problem (see, for example, [21, 17, 32, 35, 4, 37, 1, 36]). We observed in our experiments that the VNS method is usually not competitive with those efficient methods. This phenomenon is actually not surprising as the VNS method is a generic method for solving a class of convex programming, but those efficient methods are tailored for the BPDN problem, which considerably utilizes the special structure of the problem. In addition, when applied to the primal-dual CP problems (6) and (7), the performance of the VNS method strongly depends on the choice of the primal-dual initial points (see Theorem 7). For the BPDN problem, it seems not easy to choose a good initial point for the dual problem.

5.3 MAXCUT SDP relaxation problem

In this subsection, we aim to compare the performance of the VNS method for solving the MAXCUT SDP relaxation [19] when applied to formulations (13)-(16). Let G be an undirected graph with vertex set $\{1, \ldots, n\}$ and edge set E whose elements are unordered pairs of distinct vertices denoted by (i, j). Let $W \in \mathcal{S}^n$ be a matrix of nonnegative weights such that $W_{ij} = W_{ji} = 0$ whenever $(i, j) \notin E$. Consider the MAXCUT SDP relaxation

$$\max_{Z} \{\mathrm{Tr}(CZ) : Z \succeq 0, Z_{ii} = 1 \ \forall i\}, \tag{54}$$

Table 6: Comparison of VNS1–VNS4 for MAXCUT SDP relaxation

Problem	Iteration				Time			
Nodes	VNS1	VNS2	VNS3	VNS4	VNS1	VNS2	VNS3	VNS4
50	6713	1129	651	567	26.0	4.7	2.9	2.4
100	33135	4094	2533	2059	545.4	75.2	47.3	37.7
150	80672	7522	5068	3734	3444.4	360.1	245.4	177.0
200	154354	16158	10491	8115	13201.9	1549.5	1006.2	777.5
250	259318	18825	11605	9396	39024.0	3169.7	1981.7	1581.8
300	407375	25355	16284	12452	98502.5	6775.6	4459.5	3318.2
350	582214	32208	21208	16074	208360.3	12927.0	8565.6	6444.2
400	827348	40495	26629	20160	425426.4	23470.9	15532.3	11578.6

where $C \in \mathcal{S}^n$ is defined as

$$C_{ij} = \begin{cases} \sum_{k=1}^{n} W_{ik}/4 & \text{if } i = j; \\ -W_{ij}/4 & \text{otherwise} \end{cases} \qquad \forall ij. \tag{55}$$

Clearly, problem (54) is equivalent to

$$\min_{Z} \{\mathrm{Tr}(-CZ) : Z \succeq 0, Z_{ii} = 1 \ \forall i\}, \tag{56}$$

which is in the form of (6) and satisfies Assumptions A.1-A.3 due to the fact that $\mathcal{A}\mathcal{A}^*$ is the identity operator.

All instances in this test are randomly generated. In particular, we first generate $W \in \mathcal{S}^n$ according to the uniform distribution on $(0, 1)$ and set its diagonal to zero, and then compute C according to (55). The termination criterion (47) with $\epsilon = 0.1$ is used for VNS1-VNS4. The performance of these methods are presented in Table 6. In particular, column one lists the number of nodes of each instance. The number of iterations of VNS1-VNS4 is given in columns two to five, and CPU times (in seconds) are given in the last four columns, respectively. We see from Table 6 that VNS4 substantially outperforms the other three approaches for solving (56) (or, equivalently, (54)).

Though VNS4 outperforms VNS1-VNS3, we observed in our experiment that it is usually not competitive with some existing efficient methods in literature for solving the MAXCUT SDP relaxation (see, for example, [6, 22, 38, 33, 34]) as the performance of the VNS method strongly depends on the choice of the primal-dual initial points and generally it is not easy to choose good initial points.

5.4 Lovász capacity problem

In this subsection, we aim to compare the performance of the VNS method for solving the Lovász capacity problem introduced in [25] when applied to formulations (13)-(16). Let G be an undirected graph with vertex set $\{1, \ldots, n\}$ and edge set E. Consider the Lovász capacity problem

$$\min_{Z,t} \left\{t : tI - \mathbf{1}\mathbf{1}^T - Z \succeq 0, Z_{ij} = 0 \ \forall (i, j) \notin E\right\}. \tag{57}$$

Table 7: Comparison of VNS1–VNS4 for Lovász capacity problem

Problem		Iteration				Time			
Nodes	Edges	VNS1	VNS2	VNS3	VNS4	VNS1	VNS2	VNS3	VNS4
50	621	2406	2246	823	768	10.4	10.2	3.7	3.4
100	2475	8410	6791	2505	2369	148.8	128.7	44.9	42.3
150	5636	16799	12522	4506	4286	769.4	600.6	208.7	198.4
200	10060	27487	19351	6956	6612	2476.1	1861.5	594.7	564.9
250	15799	40611	27012	9650	9212	6251.4	4461.1	1415.1	1349.1
300	22637	55360	36109	12920	12333	13536.3	9439.8	3007.9	2859.2
350	30860	71313	45757	16337	15596	25559.8	17667.7	5492.7	5236.9
400	40011	90311	56063	19989	19035	46720.1	30100.9	9483.7	8994.3

Clearly, problem (57) is equivalent to

$$\max_{Z,t} \left\{ -t : tI - \mathbf{1}\mathbf{1}^T - Z \succeq 0, Z_{ij} = 0 \ \forall (i,j) \notin E \right\}, \tag{58}$$

which is in the form of (7) and satisfies Assumptions A.1-A.3 due to the fact that $\mathcal{A}\mathcal{A}^*$ is a diagonal operator.

All graphs in this experiment are randomly generated with density around 50%. The termination criterion (47) with $\epsilon = 0.1$ is used for VNS1-VNS4. The performance of these methods are presented in Table 7. The number of nodes and edges of each graph is listed in columns one and two, respectively. The number of iterations of VNS1-VNS4 is given in columns three to six, and CPU times (in seconds) are given in the last four columns, respectively. We observe from Table 7 that VNS4 is the most efficient one among these four approaches for solving (58) (or, equivalently, (57)).

In addition, we observed in our experiment that the VNS method is usually not competitive with some existing efficient methods in literature for solving the Lovász capacity problem (see, for example, [6, 22, 38, 33, 34]) as the performance of the VNS method strongly depends on the choice of the primal-dual initial points and generally it is not easy to choose good initial points.

References

[1] M. Afonso, J. Bioucas-Dias, and M. Figueiredo. An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems. Submitted to *IEEE Transactions on Image Processing*, 2009.

[2] A. Auslender and M. Teboulle. Interior gradient and proximal methods for convex and conic optimization. *SIAM Journal on Optimization*, 16(3):697–725, 2006.

[3] H. Bauschke and S. Kruk. The method of reflection-projection for convex feasibility problems with an obtuse cone. *Journal of Optimization Theory and Applications*, 120(3):503–531, 2004.

[4] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[5] R. A. Berk. *Statistical Learning from a Regression Perspective*. Springer, first edition, 2008.

[6] S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming, Series B*, 95:329–357, 2003.

[7] E. Candès and J. Romberg. 1_1-magic : Recovery of sparse signals via convex programming. User's guide, Applied & Computational Mathematics, California Institute of Technology, Pasadena, CA 91125, USA, October 2005. Available at www.l1-magic.org.

[8] E. Candes and J. Romberg. Quantitative robust uncertainty principles and optimally sparse decompositions. *Foundations of Computational Mathematics*, 6(2):227–254, 2006.

[9] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[10] E. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

[11] E. Candès and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

[12] E. Candès and T. Tao. Near-optimal signal recovery from random projections and universal encoding strategies. *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[13] E. Candès and T. Tao. The Dantzig selector: statistical estimation when p is much smaller than n. *Annals of Statistics*, 35(6):2313–2351, 2007.

[14] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

[15] ILOG CPLEX. Mathematical programming system, 2007. Available at www.cplex.com.

[16] A. Edelman. Eigenvalues and condition numbers of random matrices. *SIAM Journal on Matrix Analysis and Applications*, 9(4):543–560, 1988.

[17] M. A. T. Figueiredo, R. D. Nowak and S. J. Wright. Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. *IEEE J. Sel. Top. Signa.: Special Issue on Convex Optimization Methods for Signal Processing*, 1(4):586-598, 2007.

[18] M. P. Friedlander and M. A. Saunders. Discussion: The dantzig selector: Statistical estimation when p is much larger than n. *Annals of Statistics*, 35(6):2385–2391, 2007.

[19] M. X. Goemans and D. P. Williamson. .878-approximation algorithms for MAX CUT and MAX 2SAT. *Proceedings of the Symposium of Theoretical Computer Science*, pages 422–431, 1994.

[20] G. H. Golub and C. E. Van Loan. *Matrix Computations: Second Edition*. The John Hopkins University Press, Baltimore, MA 21211, 1989.

[21] E. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for l_1-minimization: methodology and convergence. *SIAM Journal on Optimization*, 19(3):1107–1130, 2008.

[22] C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673–696, 1999.

[23] F. Jarre and F. Rendl. An augmented primal-dual method for linear conic programs. *SIAM Journal on Optimization*, 19(2):808–823, 2008.

[24] G. Lan, Z. Lu, and R. D. C. Monteiro. Primal-dual first-order methods with $o(1/\epsilon)$ iteration iteration-complexity for cone programming. *Mathematical Programming, Series A*, 2009. DOI 10.1007/s10107-008-0261-6.

[25] L. Lovász. On the Shannon Capacity of a graph. *IEEE Transactions of Information Theory*, IT-25(1):1–7, January 1979.

[26] A. Nemirovski. Prox-method with rate of convergence $o(1/t)$ for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2005.

[27] Y. E. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. *Doklady AN SSSR*, 269(3):543–547, 1983. translated as Soviet Math. Docl.

[28] Y. E. Nesterov. Smooth minimization of nonsmooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

[29] M. A. Saunders. Pdco - matlab solver for convex optimization, 2005. Available at www.stanford.edu/group/SOL/software/pdco.html.

[30] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistics Society B*, 58(1):267–288, 1996.

[31] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. Manuscript, Department of Mathematics, University of Washington, Seattle, WA, 98195, USA, May 2008. Submitted to *SIAM Journal on Optimization*.

[32] E. Van den Berg and M. P. Friedlander. Probing the Pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890-912, 2008.

[33] Z. Wen, D. Goldfarb, and K. Scheinberg. Row by row methods for semidefinite programming. Technical report, 2009.

[34] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented Lagrangian methods for semidefinite programming. Technical report, 2009.

[35] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

[36] J. Yang and Y. Zhang. Alternating direction algorithms for L_1-problems in compressive sensing. Technical report, Rice University, 2009.

[37] S. Yun and K.-C. Toh. A coordinate gradient descent method for l_1-regularized convex minimization. To appear in *Computational Optimization and Applications*, May 2008.

[38] X. Zhao, D. Sun, and K.-C. Toh. A Newton-CG augmented Lagrangian method for semidefinite programming. *SIAM Journal on Optimization*, 20(4): 1737–1765, 2010.