Optim Eng (2018) 19:815–844
https://doi.org/10.1007/s11081-018-9378-5

Optimal subgradient methods: computational properties for large-scale linear inverse problems

Masoud Ahookhosh[1]

Received: 27 February 2016 / Revised: 21 April 2017 / Accepted: 12 February 2018 / Published online: 10 March 2018

Abstract This paper studies the computational properties of the optimal subgradient algorithm (OSGA) for applications of linear inverse problems involving high-dimensional data. First, such convex problems are formulated as a class of convex problems with multi-term composite objective functions involving linear mappings. Next, an efficient procedure for computing the first-order oracle for such problems is provided and OSGA is equipped with some prox-functions such that the OSGA subproblem is solved in a closed form. Further, a comprehensive comparison among the most popular first-order methods is given. Then, several Nesterov-type optimal methods (originally proposed for smooth problems) are adapted to solve nonsmooth problems by simply passing a subgradient instead of the gradient, where the results of these subgradient methods are competitive and totally interesting for solving nonsmooth problems. Finally, numerical results with several inverse problems (deblurring with isotropic total variation, elastic net, and ℓ_1-minimization) show the efficiency of OSGA and the adapted Nesterov-type optimal methods for large-scale problems. For the deblurring problem, the efficiency measures of the improvement on the signl-to-noise ratio and the peak signal-to-noise ratio are used. The software package implementing OSGA is publicly available.

Keywords Structured convex optimization · Linear inverse problems · Sparse nonsmooth optimization · First-order information · Subgradient methods · Optimal complexity · High-dimensional data

✉ Masoud Ahookhosh
masoud.ahookhosh@univie.ac.at

[1] Faculty of Mathematics, University of Vienna, Oskar-Morgenstern-Platz 1, 1090 Vienna, Austria

1 Introduction

In many applications, e.g., those arising in signal and image processing, machine learning, compressed sensing, geophysics and statistics, key features cannot be studied by straightforward investigations, but must be indirectly inferred from some observable quantities. Due to this feature, they are typically referred to as inverse problems. In particular, a linear relevance between the features of interest and the observed data leads to linear inverse problems. For finite-dimensional vector spaces V and U, if $y \in U$ is an indirect observation of an original object $x \in V$ and $\mathcal{A} : V \to U$ is a linear operator, then the linear inverse problem is given by

$$y = \mathcal{A}x + v, \tag{1}$$

where $v \in U$ represents an additive or impulsive noise vector about which little is known apart from qualitative knowledge.

In practice, the system (1) is typically underdetermined, rank-deficient, or ill-conditioned. The primary difficulty with linear inverse problems is that the inverse object is extremely sensitive to y because of small or zero singular values of $\mathcal{A}$, which leads to ill-conditioned systems. Indeed, in the case that $\mathcal{A}^{-1}$ for square problems or pseudo-inverse $\mathcal{A}^{\dagger} = (\mathcal{A}^{*}\mathcal{A})^{-1}\mathcal{A}^{*}$ for full rank over-determined systems exists, analyzing the singular value decomposition has shown that $\widetilde{x} = \mathcal{A}^{-1}y$ or $\widetilde{x} = \mathcal{A}^{\dagger}y$ is an inaccurate and meaningless approximation for x; see Neumaier (1998). Moreover, when the vector v is not known, one cannot solve (1) directly.

Linear inverse problems are usually underdetermined or rank-deficient, i.e., if a solution exists, then there exist infinitely many solutions. Hence, some additional information is required to determine a satisfactory solution of (1). One can find a solution by minimizing $\|\mathcal{A}x - y\|_2$, which is a linear least-squares problem; $\|\cdot\|_2$ is the Euclidean norm. Since the problem (1) is typically ill-conditioned, the solution of the linear least-squares problem is commonly improper. Tikhonov (1963) proposed the penalized minimization problem

$$\min_{x \in V} \frac{1}{2}\|\mathcal{A}x - y\|_2^2 + \frac{\lambda}{2}\|x\|_2^2, \tag{2}$$

where λ is a regularization parameter controlling the trade-off between the data fitting term and the regularization term. The problem (2) is convex and smooth, and selecting a suitable regularization parameter leads to a well-posed problem; however, in many applications, the sparsest solution of (1) among all solutions is desirable, which leads to the constrained problem

$$\begin{array}{ll} \min & \|x\|_0 \\ \text{s.t.} & \|\mathcal{A}x - y\|_2 \le \epsilon, \end{array} \tag{3}$$

where $\|x\|_0$ is the number of all nonzero elements of x and ϵ is a small nonnegative constant. Since this objective function is nonconvex, its convex relaxation (Bruckstein et al. 2009) given by

$$\begin{array}{ll} \min & \|x\|_1 \\ \text{s.t.} & \|\mathcal{A}x - y\|_2 \leq \epsilon \end{array} \tag{4}$$

or its unconstrained reformulation

$$\min_{x \in V} \frac{1}{2} \|\mathcal{A}x - y\|_2^2 + \lambda \|x\|_1, \tag{5}$$

is referred to as *basis pursuit denoising* (Chen et al. 2001) or the *least absolute shrinkage and selection operator (LASSO)* (Tibshirani 1996).

Let us consider a function $f : V \to \mathbb{R}$ given by

$$f(x) = \sum_{i=1}^{n_1} f_i(\mathcal{A}_i x) + \sum_{j=1}^{n_2} \varphi_j(\mathcal{W}_j x), \tag{6}$$

where $f_i : U_i \to \mathbb{R}$, for $i = 1, 2, \ldots, n_1$, are convex functions, $\varphi_j : V_j \to \mathbb{R}$, for $j = 1, 2, \ldots, n_2$, are smooth or nonsmooth convex functions, and $\mathcal{A}_i : V \to U_i$, for $i = 1, 2, \ldots, n_1$, and $\mathcal{W}_j : V \to V_j$, for $j = 1, 2, \ldots, n_2$, are linear operators. Here, f is called the multi-term composite function (see also He et al. 2015). In general, linear inverse problems can be modeled as a special case of the minimization problem

$$\widehat{f} := \min_{x \in V} f(x). \tag{7}$$

If the boundedness of the level set $N_f(x_0) = \{x \in V | f(x) \leq f(x_0)\}$ is assumed, problem (7) has a global minimizer denoted by $\widehat{x}$. In what follows, we will see that many well-studied structured optimization problems are special cases of (7). The objectives of the form (6) have been frequently stated in many applications, e.g., hybrid regularizations and mixed penalty functions for solving problems in fields such as signal and image processing, machine learning, geophysics, economics, and statistics. In this case, f_i $(i = 1, 2, \ldots, n_1)$ are the data fidelity while φ_j $(j = 1, 2, \ldots, n_2)$ stand for regularizers that can be smooth or nonsmooth and convex or nonconvex; however, in this paper, we consider only convex regularizers. As an example, the scaled elastic net problem

$$f(x) = \frac{1}{2} \|\mathcal{A}x - b\|_2^2 + \frac{1}{2} \lambda_1 \|\mathcal{W}_1 x\|_2^2 + \lambda_2 \|\mathcal{W}_2 x\|_1 \tag{8}$$

is of the form (7) with $f(x) = f_1(\mathcal{A}_1 x) + \varphi_1(\mathcal{W}_1 x) + \varphi_2(\mathcal{W}_2 x)$, where $f_1(\mathcal{A}_1 x) := \frac{1}{2} \|\mathcal{A}_1 x - b\|_2^2$, $\varphi_1(\mathcal{W}_1 x) := \frac{\lambda_1}{2} \|\mathcal{W}_1 x\|_2^2$, $\varphi_2(\mathcal{W}_2 x) := \lambda_2 \|\mathcal{W}_2 x\|_1$.

1.1 Examples of inverse problems

In this subsection, we describe two classical inverse problems that appear frequently in many applications (see Sect. 3 for some applications).

Example 1 (*Image restoration*) Image reconstruction, also called image restoration, is one of the classical linear inverse problems; see, e.g., Andrews and Hunt (1977) and Bertero and Boccacci (1998). The goal is to reconstruct images from some observations. Let $y \in U$ be a noisy indirect observation of an image $x \in V$ with an unknown noise $\delta \in U$, and let $\mathcal{A} : V \to U$ be a linear operator. To recover the unknown vector x, we use the linear inverse model (1). A typical model for image reconstruction involves a smooth or nonsmooth data-fidelity term penalized by some convex regularization penalties, which are selected based on expected features of the recovered image. It is typical to use the models (2), (5), or (6) to derive a solution for the corresponding inverse problem; see, e.g., Kaufman and Neumaier (1996) and Neumaier (1998).

For a known image $x \in V$, in the analysis strategy, the image is typically recovered by solving either

$$\min_{x \in V} \frac{1}{2} \|\mathcal{A}x - y\|_2^2 + \lambda\varphi(x) \tag{9}$$

or

$$\min_{x \in V} \|\mathcal{A}x - y\|_1 + \lambda\varphi(x). \tag{10}$$

Although various regularizers like the ℓ_p-norm are popular in image restoration, it is arguable that the best known and frequently employed regularizer in the analysis approach is the total variation (TV).

The pioneering work on TV was undertaken by Osher et al. (1992). Total variation regularizations are able to restore discontinuities of images and to recover the edges, and so they have been widely used in applications. TV is originally defined in an infinite-dimensional Hilbert space; however, for digital image processing a discrete version in a finite-dimensional space such as $\mathbb{R}^n$ of pixel values on a two-dimensional lattice has been used. Isotropic total variation (ITV) is popular in signal and image processing; it is given by

$$\begin{aligned} \|x\|_{ITV} \quad := \quad & \sum_i^{m-1} \sum_j^{n-1} \sqrt{(x_{i+1,j} - x_{i,j})^2 + (x_{i,j+1} - x_{i,j})^2} \\ & + \sum_i^{m-1} |x_{i+1,n} - x_{i,n}| + \sum_j^{n-1} |x_{m,j+1} - x_{m,j}|, \end{aligned} \tag{11}$$

for $x \in \mathbb{R}^{m \times n}$, see, e.g., Beck and Teboulle (2009b). Model (9) with $\varphi(\cdot) = \|\cdot\|_{ITV}$ is denoted by L22ITVR and model (10) with $\varphi(\cdot) = \|\cdot\|_{ITV}$ is denoted by L1ITVR.

Example 2 (*Compressed sensing*) Finding sparse solutions of problems using structured models has become popular in various areas of applied mathematics. In most cases, the problem involves high-dimensional data with a small number of available measurements, where the core of these problems involves an optimization problem. Due to the sparsity of the solutions and the structure of the problems, these optimization problems can be solved in a reasonable time even for extremely high-dimensional data sets. Basis pursuit, LASSO, wavelet-based deconvolution, and compressed sensing are some examples. Compressed sensing in particular has received much attention in recent years; see Candés (2006) and Donoho (2006).

Among the fields involving sparse optimization, compressed sensing is a novel sensing/sampling framework for acquiring and recovering objects such as a sparse image or signal in the most efficient way possible. Conventional processes in the image/signal acquisition from frequency data follow the Nyquist–Shannon density sampling theorem declaring that the number of samples required for reconstructing an image or a signal matches the number of pixels in the image and the bandwidth of the signal, respectively. Since most of the data we acquire can be thrown away with almost no perceptual loss, one may recover only the parts of the data that will be useful in the final reconstruction, which leads to compressed sensing theory; see, e.g., Donoho (2006) and the references therein.

In compressed sensing, it is assumed that the considered object, image or signal, has a sparse representation in some bases or dictionaries, and the considered representation dictionary is incoherent with the sensing basis; see Candés (2006) and Donoho (2006). Once an object is known to be sparse in a specific dictionary, one can find a set of measurements or sensing bases supposed to be incoherent with the dictionary. An underdetermined system of linear equations of the form (1) then emerges; it is typically ill-conditioned. Hence, the main challenges are finding a measurement matrix and reformulating this inverse problem as an appropriate minimization involving regularizers.

1.2 State of the art

Over the past few decades, the solution of structured convex optimization problems of the form (7) has received much attention. Convex optimization is theoretically and computationally regarded as a mature area of optimization that is appropriate for large-scale problems appearing in applications. Since problems of the form (7) typically involve high-dimensional data, optimization schemes should avoid costly operations and large amounts of memory. Depending on the objective structure, various first-order methods have been developed to deal with such large problems. Some popular first-order methods are gradient methods (Nesterov 2013, 2015), subgradient schemes (Beck and Teboulle 2003; Nemirovsky and Yudin 1983; Nesterov 2004), bundle-type methods (Lan 2015), smoothing methods (Beck and Teboulle 2009a; BoŢ and Hendrich 2013a, 2015; Nesterov 2005) and proximal methods (Beck and Teboulle 2009a; Becker et al. 2011).

In the context of convex optimization, the analytical complexity refers to the number of calls of the first-order oracle (giving a function value and a subgradient) to achieve an ε-solution of the problem; see Nemirovsky and Yudin (1983) and Nesterov (2004). Nemirovsky and Yudin (1983) proved that the complexity bound of first-order methods for finding an ε-solution is $\Omega(\varepsilon^{-1/2})$ for smooth objectives with Lipschitz-continuous gradients and $\Omega(\varepsilon^{-2})$ for Lipschitz-continuous nonsmooth objectives. Indeed, the interesting feature of these error bounds is that they are independent of the problem dimension. For the established class of problems, an algorithm is called optimal if it can achieve these complexities. In the case of nonsmooth convex problems, the subgradient and mirror descent methods are optimal; see Nemirovsky and Yudin (1983). The seminal optimal first-order method for smooth convex problems with Lipschitz continuous gradients was introduced by

Nesterov (1983), and since then he has developed the idea of optimal methods for several classes of problems; see, e.g., Nesterov (2004, 2013, 2015). The theoretical analysis and surprising computational results of optimal methods are most interesting, especially for large-scale problems. These appealing features for large-scale convex problems have led to much interest in optimal first-order methods; see, e.g., Beck and Teboulle (2003), Becker et al. (2011), Chen et al. (2014, 2015), Gonzaga and Karas (2013), and Lan (2015).

The primary difficulty in an implementation of a first-order method is the requirement for global knowledge of the objective function, e.g., a Lipschitz constant for the function values for nonsmooth problems and a Lipschitz constant for the gradients in smooth cases. Nesterov (2015) proposed an adaptive procedure to approximate the Lipschitz constant, but it still needs a suitable initial guess. Lan (2015) proposed an optimal bundle-level method that does not need to know the global parameters. Recently, Neumaier (2016) introduced an optimal subgradient algorithm (OSGA) by incorporating a linear relaxation of the objective and a prox-function into a fractional subproblem to construct a framework that can be employed for both smooth and nonsmooth convex optimization problems at the same time without needing Lipschitz constants. OSGA achieves the optimal complexity of first-order methods and can be regarded as a fully adaptive alternative to Nesterov-type optimal methods.

1.3 Contribution

We aim to study the computational properties of OSGA to deal with large-scale linear inverse problems and to provide a software package for this method. There exist many linear inverse problems with complicated objective functions of the form (7) consisting of various combinations of linear operators and regularizers; however, they cannot be handled by proximal point methods. For example, consider the *scaled LASSO* problem, which has the form (5) when we replace $\|x\|_1$ with $\|\mathcal{W}x\|_1$, for a linear operator $\mathcal{W}$. For this problem, the proximity operator

$$\mathrm{prox}_{\lambda\|\mathcal{W}\cdot\|_1}(y) = \underset{x\in\mathbb{R}^n}{\mathrm{argmin}}\,\frac{1}{2}\|x-y\|_2^2 + \lambda\|\mathcal{W}x\|_1 \tag{12}$$

must be computed, where this auxiliary problem cannot be solved explicitly using the iterative shrinkage-thresholding except if $\mathcal{W}$ is orthonormal, i.e., proximal-based algorithms are not applicable effectively. Since OSGA has low memory requirements and needs only first-order information (function values and subgradients), we believe that it can be used efficiently to minimize such complicated objective functions. However, to apply OSGA effectively, we need suitable prox-functions such that the OSGA subproblem can be solved explicitly and an effective routine for computing the first-order information (see Sect. 2.1). Further, we aim to adapt several Nesterov-type optimal methods (originally proposed for smooth problems) to solve nonsmooth problems by passing a subgradient instead of the gradient. The surprising results show that the adapted algorithms perform competitively. Finally, the paper is accompanied with a software release.

The underlying function of the OSGA subproblem is quasi-concave and finding its solution is the most costly part of the algorithm. While it is crucial to efficiently solve this subproblem, it is not trivial to find such solutions. We consider only unconstrained convex optimization problems in which a closed-form solution for this subproblem can be found. Moreover, for unconstrained problems involving costly linear operators, a combination of a multi-dimensional subspace search technique and OSGA is studied in Ahookhosh and Neumaier (2013, 2018). In Ahookhosh and Neumaier (2017b), we gave one projection version of OSGA and provided a framework to solve the subproblem over simple convex domains or simple functional constraints. In particular, in Ahookhosh and Neumaier (2017a) we described a scheme to compute the global solution of the OSGA subproblem with bound constraints. In Ahookhosh and Neumaier (2017c), we reformulated structured nonsmooth convex problems as smooth problems with one more functional constraint and adapted a version of OSGA to handle the reformulated problem, which attains the complexity $\mathcal{O}(\varepsilon^{-1/2})$.

The remainder of this paper is structured as follows. In Sect. 2, we briefly discuss OSGA and its implementation issues for multi-term composite problems. Section 3 presents the implementation of OSGA and reports comparisons with popular first-order methods and state-of-the-art solvers for practical applications. Finally, we provide concluding remarks in Sect. 4.

2 A review of OSGA

In this section, we give a short sketch of the optimal subgradient algorithm (OSGA; see Algorithm 1). It was proposed by Neumaier (2016) for the convex constrained minimization problem

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & x \in C, \end{array} \tag{13}$$

where $f : C \to \mathbb{R}$ is a proper and convex function defined on a nonempty, closed, and convex subset C of a finite-dimensional vector space V.

OSGA is a subgradient algorithm for problem (13) that uses first-order information, i.e., function values and subgradients, to construct a sequence of iterates $\{x_k\} \subset C$ whose function values $\{f(x_k)\}$ converge to the minimum $\widehat{f} = f(\widehat{x})$ with the optimal complexity. OSGA requires no information regarding global parameters such as the Lipschitz constants of the function values and gradients. It uses a continuously differentiable prox-function $Q : V \to \mathbb{R}$, which is a strongly convex function with the convexity parameter σ satisfying

$$Q(z) \geq Q(x) + \langle g_Q(x), z - x\rangle + \frac{\sigma}{2}\|z - x\|^2, \tag{14}$$

for all $x, z \in C$ where $g_Q(x)$ denotes the gradient of Q at x and $\langle \cdot, \cdot \rangle$ stands for the inner product. We also assume that

$$Q_0 := \inf_{x \in C} Q(x) > 0, \tag{15}$$

where a point $x_0 \in C$ minimizing this problem is called the center of Q. From the definition of the center of Q, the first-order optimality condition for (15), and (14), we obtain $Q(x) \geq Q_0 + \frac{\sigma}{2}\|x - x_0\|^2$, implying $Q(x) > 0$ and $Q(x)$ is greater than the quadratic term $Q_0 + \frac{\sigma}{2}\|x - x_0\|^2$. At each iteration, OSGA satisfies the bound

$$0 \leq f(x_b) - \widehat{f} \leq \eta Q(\widehat{x}) \tag{16}$$

on the currently best function value $f(x_b)$ with a monotonically decreasing error factor η that is guaranteed to converge to zero by an appropriate step-size selection strategy (see Algorithm 2). Note that $\widehat{x}$ is not known a priori, thus the error bound is not fully constructive. However, it is sufficient to guarantee the convergence of $f(x_b)$ to $\widehat{f}$ with a predictable worst-case complexity. To maintain (16), OSGA considers linear relaxations of f at z,

$$f(z) \geq \gamma + \langle h, z\rangle \text{ for } \text{ all } z \in C, \tag{17}$$

where $\gamma \in \mathbb{R}$ and $h \in V^*$, updated using linear underestimators available from the subgradients evaluated (see Algorithm 1). For each such linear relaxation, OSGA solves a maximization problem of the form

$$\begin{aligned} E(\gamma, h) := \max\ & E_{\gamma,h}(x) \\ \text{s.t.}\ & x \in C, \end{aligned} \tag{18}$$

where

$$E_{\gamma,h}(x) := -\frac{\gamma + \langle h, x\rangle}{Q(x)}. \tag{19}$$

Let $\gamma_b := \gamma - f(x_b)$, $u := U(\gamma_b, h) \in C$ be the solution of (18), and $\eta := E(\gamma_b, h)$ be the corresponding minimum. From (17) and (19), we obtain

$$\eta = E(\gamma_b, h) \geq -\frac{\gamma - f(x_b) + \langle h, u\rangle}{Q(u)} \geq \frac{f(x_b) - \widehat{f}}{Q(u)} \geq 0, \tag{20}$$

which implies that (16) is valid. If x_b is not optimal for (13), then the rightmost inequality in (20) is strict, and since $Q(z) \geq Q_0 > 0$, we conclude that the maximum η is positive. Indeed, (16) implies that the rate of decrease in the error bound $f(x_b) - \widehat{f}$ is the same as the convergence rate of the sequence $\{\eta_k\}_{k \geq 0}$, where k is the iteration counter, which is the main motivation for the subproblem (18).

For simplicity, we set $f_{x_b} := f(x_b)$, $f_{x_b'} := f(x_b')$, $f_x := f(x)$, $f_{x'} := f(x')$, $g_{x_b} \in \partial f(x_b)$, and $g_x \in \partial f(x)$. Motivated by the above discussion and that of Sect. 2 in Neumaier (2016) about the linear relaxation constructions and the case of strongly convex functions, OSGA is presented in Algorithm 1.

Algorithm 1: OSGA (Optimal SubGradient Algorithm)

```
Input: $\delta, \alpha_{\max} \in ]0,1[$,  $0 < \kappa' \leq \kappa$;  local parameters: $x_0$, $\mu \geq 0$;
Output: $x_b$,  $f_{x_b}$;
1  begin
2      $x_b = x_0$;  compute $f_{x_b}$ and $g_{x_b}$;
3      $h = g_{x_b} - \mu g_Q(x_b)$;  $\gamma = f_{x_b} - \mu Q(x_b) - \langle h, x_b \rangle$;
4      $\gamma_b = \gamma - f_{x_b}$;  $u = U(\gamma_b, h)$;  $\eta = E(\gamma_b, h) - \mu$;  $\alpha = \alpha_{\max}$;
5      while stopping criteria do not hold do
6          $x = x_b + \alpha(u - x_b)$; compute $f_x$ and $g_x$;
7          $g = g_x - \mu g_Q(x)$; $\overline{h} = h + \alpha(g - h)$; $\overline{\gamma} = \gamma + \alpha(f_x - \mu Q(x) - \langle g, x \rangle - \gamma)$;
8          $x_b' = \text{argmin}_{z \in \{x_b, x\}} f(z)$;  $f_{x_b'} = \min\{f_{x_b}, f_x\}$; $\gamma_b' = \overline{\gamma} - f_{x_b'}$;
9          $u' = U(\gamma_b', \overline{h})$; $x' = x_b + \alpha(u' - x_b)$; compute $f_{x'}$;
10         choose $\overline{x}_b$ in such a way that $f_{\overline{x}_b} \leq \min\{f_{x_b'}, f_{x'}\}$;
11         $\overline{\gamma}_b = \overline{\gamma} - f_{\overline{x}_b}$;  $\overline{u} = U(\overline{\gamma}_b, \overline{h})$;  $\overline{\eta} = E(\overline{\gamma}_b, \overline{h}) - \mu$;  $x_b = \overline{x}_b$;  $f_{x_b} = f_{\overline{x}_b}$;
12         update the parameters $\alpha$, $h$, $\gamma$, $\eta$ and $u$ using PUS;
13     end
14 end
```

In Line 12, OSGA uses the following scheme for updating the given parameters α, h, γ, η, and u, as stated in Algorithm 2 [see Section 2 of Neumaier (2016) for more information regarding step-size selection].

Algorithm 2: PUS (Parameter Updating Scheme)

```
Input: $\delta$,  $\alpha_{\max} \in ]0,1[$,  $0 < \kappa' \leq \kappa$, $\alpha$, $\eta$, $\overline{h}$, $\overline{\gamma}$, $\overline{\eta}$, $\overline{u}$;
Output: $\alpha$,  $h$,  $\gamma$,  $\eta$,  $u$;
1  begin
2      $R = (\eta - \overline{\eta})/(\delta \alpha \eta)$;
3      if $R < 1$ then
4          $\overline{\alpha} = \alpha e^{-\kappa}$;
5      else
6          $\overline{\alpha} = \min(\alpha e^{\kappa'(R-1)}, \alpha_{\max})$;
7      end
8      $\alpha = \overline{\alpha}$;
9      if $\overline{\eta} < \eta$ then
10         $h = \overline{h}$; $\gamma = \overline{\gamma}$; $\eta = \overline{\eta}$; $u = \overline{u}$;
11     end
12 end
```

Let us assume that the sublevel set $N_f(x_0) = \{x \in V \mid f(x) \leq f(x_0)\}$ is bounded for the starting point $x_0 \in C$. Since f is convex, the sublevel set $N_f(x_0)$ is closed, V is finite-dimensional, and $N_f(x_0)$ is convex and compact. From the continuity and properness of the objective f, it attains its global minimizer on the sublevel set $N_f(x_0)$. Therefore, there is at least one minimizer $\widehat{x}$.

We now consider the complexity of OSGA given by Neumaier (2016) in the subsequent theorem, which is valid for $C = V$ that is the case in the current study. In light of the complexity analysis of first-order methods given in Nemirovsky and Yudin (1983) and Nesterov (2004), the achieved complexities are optimal for Lipschitz-continuous nonsmooth problems and smooth problems with Lipschitz-continuous gradients.

Theorem 1 [Neumaier (2016), Theorem 5.1] *Suppose that $f - \mu Q$ is convex and $\mu \geq 0$. Then we have*

(i) (*Nonsmooth complexity bound*) *If the points generated by Algorithm 1 stay in a bounded region of the interior of C, or if f is Lipschitz continuous on C, the total number of iterations needed to reach a point with* $f(x) \leq f(u) + \varepsilon$ *is at most* $\mathcal{O}((\varepsilon^2 + \mu\varepsilon)^{-1})$. *Thus the asymptotic worst case complexity is* $\mathcal{O}(\varepsilon^{-2})$ *when* $\mu = 0$, *and* $\mathcal{O}(\varepsilon^{-1})$ *when* $\mu > 0$.

(ii) (*Smooth complexity bound*) *If f has Lipschitz-continuous gradients with Lipschitz constant L, the total number of iterations needed by Algorithm 1 to reach a point with* $f(x) \leq f(u) + \varepsilon$ *is at most* $\mathcal{O}(\varepsilon^{-1/2})$ if $\mu = 0$, *and at most* $\mathcal{O}(\sqrt{L/\mu}\log \varepsilon^{-1})$ *if* $\mu > 0$.

2.1 Implementation issues for multi-term composite problems

In this section, we first propose some prox-functions and then derive a closed-form solution for subproblem (18).

For $f : V \to \mathbb{R}$ and a finite-dimensional vector space V, the subdifferential of f at x is given by

$$\partial f(x) = \sum_{i=1}^{n_1} \mathcal{A}_i^* \partial f_i(\mathcal{A}_i x) + \sum_{j=1}^{n_2} \mathcal{W}_j^* \partial \varphi_j(\mathcal{W}_j x). \tag{21}$$

Note that practical problems typically involve high-dimensional data, i.e., the computation of the function values and subgradients is expensive. On the other hand, in many applications the major part of the cost of computing function values and subgradients relates to the application of forward and adjoint linear operators, arising from the presence of affine terms in (6). Therefore, the number of applications of linear operators and their adjoints should be as low as possible in each call of the first-order oracle. Considering the structure of (6), we see that affine terms $\mathcal{A}_i x$ $(i = 1, 2, \ldots, n_1)$ and $\mathcal{W}_j x$ $(j = 1, 2, \ldots, n_2)$ appear in both the function value and a subgradient of f at x. By setting $v_x^i := \mathcal{A}_i x$ $(i = 1, 2, \ldots, n_1)$ and $w_x^j := \mathcal{W}_j x$ $(j = 1, 2, \ldots, n_2)$ and using (21), we provide the first-order oracle in Algorithm 3.

Algorithm 3: NFO-FG (Nonsmooth First-order Oracle)

```
Input: $\mathcal{A}_i$ for $i = 1, \ldots, n_1$,  $\mathcal{W}_j$ for $j = 1, \ldots, n_2$,  $x$;
Output: $f_x$;  $g_x$;
1 begin
2     $v_x^i = \mathcal{A}_i x$ for $i = 1, \ldots, n_1$;
3     $w_x^j = \mathcal{W}_j x$ for $j = 1, \ldots, n_2$;
4     $f_x = \sum_{i=1}^{n_1} f_i(v_x^i) + \sum_{j=1}^{n_2} \varphi_j(w_x^j)$;
5     $g_x \in \sum_{i=1}^{n_1} \mathcal{A}_i^* \; \partial f_i(v_x^i) + \sum_{j=1}^{n_2} \mathcal{W}_i^* \; \partial \varphi_j(w_x^j)$;
6 end
```

Each call of the oracle $\mathcal{O}(x) = (f_x, g_x)$ requires $n_1 + n_2$ calls of forward and adjoint linear operators. By using this scheme, one can avoid the double application of expensive linear operators in the computation of the function values and subgradients. We also emphasize that if the total computational cost of the oracle is dominated by the application of linear mappings, the complexity of an algorithm can be measured by counting the number of forward and adjoint linear operators used to achieve an ε-solution.

Let $\|\cdot\|$ be the quadratic norm on vector space V, i.e.,

$$\|x\|_{\mathcal{D}} := \sqrt{\langle \mathcal{D}x, x\rangle} \tag{22}$$

by means of a preconditioner $\mathcal{D}$, where $\mathcal{D}$ is symmetric and positive definite. The associated dual norm on V^* is given by

$$\|h\|_{*\mathcal{D}} := \|\mathcal{D}^{-1}h\|_{\mathcal{D}} = \sqrt{\langle h, \mathcal{D}^{-1}h\rangle}, \tag{23}$$

where $\mathcal{D}^{-1}$ is the inverse of $\mathcal{D}$. We here consider the quadratic function

$$Q(x) := Q_0 + \frac{\sigma}{2}\|x - x_0\|_{\mathcal{D}}^2, \tag{24}$$

where Q_0 is a positive number and $x_0 \in V$ is the center of Q.

Let us emphasize that the efficient solution of subproblem (18) is closely related to the selection of the prox-functions. In Neumaier (2016), it is shown that with the prox-function (24) (with $\sigma = 1$), subproblem (18) can be solved explicitly.

Proposition 1 [Neumaier (2016), Section 2.3] *Suppose Q is determined by (24) and $Q_0 > 0$. Then Q is a prox-function with the center x_0 and satisfies $Q(x) \geq Q_0 + \frac{\sigma}{2}\|x - x_0\|^2$. Moreover, subproblem (18) with this Q is explicitly solved by*

$$u = x_0 - E(\gamma, h)^{-1}\sigma^{-1}\mathcal{D}^{-1}h \tag{25}$$

with

$$E(\gamma, h) = \frac{-\beta_1 + \sqrt{\beta_1^2 + 4Q_0\beta_2}}{2Q_0} = \frac{2\beta_2}{\beta_1 + \sqrt{\beta_1^2 + 4Q_0\beta_2}}, \tag{26}$$

where $\beta_1 = \gamma + \langle h, x_0\rangle$ and $\beta_2 = \left(\frac{1}{2}\sigma^{-2} - \sigma^{-1}\right)\|h\|_*^2$.

Notice that the error bound (16) is proportional to $Q(\widehat{x})$, which means that an acceptable choice for x_0 makes the term $Q(\widehat{x}) = Q_0 + \frac{1}{2}\|\widehat{x} - x_0\|_{\mathcal{D}}^2$ small. Hence, selecting a suitable starting point x_0 as close as possible to the optimizer $\widehat{x}$ has a positive effect on the convergence rate. This also suggests that a reasonable choice for Q_0 is $Q_0 \approx \frac{1}{2}\|\widehat{x} - x_0\|_{\mathcal{D}}^2$.

For simplicity, we assume that there are only two linear operators $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^{m_1}$ and $\mathcal{W} : \mathbb{R}^n \to \mathbb{R}^{m_2}$ $(m_1, m_2 \ll n)$ in problem (6), i.e.,

$$f(x) = f_1(\mathcal{A}_1 x) + \varphi_1(\mathcal{W}_1 x). \tag{27}$$

OSGA needs two function values (lines 7 and 9) and one subgradient (line 7) in each step. Therefore, if computing $f_1(\cdot)$ and $\varphi_1(\cdot)$ is negligible compared to the cost of matrix-vector products, Algorithm 3 implies that $\mathcal{O}(n^2)$ operations are needed. To implement OSGA, we require two solutions of subproblem (18) to compute u (line 6) and u' (line 9). From Proposition 1, we need $\mathcal{O}(n)$ to compute u and u'. Since the cost of the other operations in OSGA is much less than the cost of computing the first-order oracle and the subproblem solutions, OSGA requires $\mathcal{O}(n^2)$ operations at each step. Taking this into account and the bounds on the number of iterations given in Theorem 1, OSGA requires at most $\mathcal{O}(\varepsilon^2 n^2)$ operations for Lipschitz-continuous nonsmooth problems and $\mathcal{O}(\varepsilon^{1/2} n^2)$ operations for smooth problems with Lipschitz-continuous gradients.

3 Comparisons of first-order methods

This section describes extensive numerical experiments and comparisons of OSGA and state-of-the-art first-order methods to for several applications of inverse problems. We start with image deblurring, adapt some Nesterov-type optimal (for smooth problems) methods, and apply them and some state-of-the-art first-order methods to elastic net and ℓ_1-minimization problems.

The OSGA software package (implemented in MATLAB) for unconstrained convex optimization problems is publicly available at http://homepage.univie.ac.at/masoud.ahookhosh/.

Some examples are available to show how to apply OSGA. The interface to each subprogram in the package is fully documented in the associated file. Moreover, the OSGA user's manual Ahookhosh (2015b) describes the design of the codes and how to solve problems. We use the prox-function (24) with the identity matrix $\mathcal{D} = I$ and $Q_0 = \frac{1}{2}\|x_0\|_2 + \epsilon$, where ϵ is the machine precision. We also use the parameters

$$\delta = 0.9, \alpha_{max} = 0.7, \kappa = \kappa' = 0.5. \tag{28}$$

In our comparison, we use the codes of the authors where they are available. Otherwise, the codes are written in MATLAB, and the default parameter values for the algorithms and packages are used. All implementations are executed on a Dell Precision Tower 7000 Series 7810 (Dual Intel Xeon Processor E5-2620 v4 with 32 GB RAM).

To illustrate the results, we used the Dolan and Moré performance profile Dolan and Moré (2002). In this procedure, the performance of each algorithm is measured by the ratio of its computational outcome versus the best numerical outcome of all the algorithms. This performance profile offers a tool for statistically comparing algorithm performance. Let $\mathcal{S}$ be a set of algorithms and $\mathcal{P}$ be a set of test problems. For each problem p and algorithm s, $t_{p,s}$ denotes the computational outcome with respect to the performance index, which is used in the definition of the performance ratio

$$r_{p,s} := \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}}. \tag{29}$$

If an algorithm s fails to solve a problem p, the procedure sets $r_{p,s} := r_{\text{failed}}$, where

r_{failed} should be strictly larger than any performance ratio (29). Let n_p be the number of problems in the experiment. For any factor $\tau \in \mathbb{R}$, the overall performance of an algorithm s is given by

$$\rho_s(\tau) := \frac{1}{n_p}\text{size}\{\mathrm{p} \in \mathcal{P} : \mathrm{r_{p,s}} \leq \tau\}. \tag{30}$$

Here, $\rho_s(\tau)$ is the probability that a performance ratio $r_{p,s}$ of an algorithm $s \in \mathcal{S}$ is within a factor τ of the best possible ratio. The function $\rho_s(\tau)$ is a distribution function for the performance ratio. In particular, $\rho_s(1)$ is the probability that an algorithm s wins over all other algorithms, and $\lim_{\tau \to r_{\text{failed}}} \rho_s(\tau)$ is the probability that the algorithm s solves all the problems considered. Therefore, this performance profile can be employed as a measure of efficiency among all the algorithms. In the performance profiles of Figs. 1 and 3 in the next subsection, the number τ is represented in the x-axis, while $P(r_{p,s} \leq \tau : 1 \leq s \leq n_s)$ is shown in the y-axis.

3.1 Image deblurring with isotropic total variation

Image blur is one of the most common problems in photography and can often ruin photographs. Hence, it has been an important problem in digital imaging, which is an inherently ill-conditioned inverse problem of the form (1) leading to the optimization problem of the form (6).

Let $\widehat{x}$ and $\widehat{f}$ be a minimizer and the corresponding minimum of the problems (9) or (10), and also let x and f be the current iteration point and function value of the algorithms, respectively. Let x_0 be a $m \times n$ clean image. The performances are measured by the relative errors

$$\delta_1 := \frac{\|x - \widehat{x}\|_2}{\|\widehat{x}\|_2} \quad \text{and} \quad \delta_2 := \frac{f - \widehat{f}}{f_0 - \widehat{f}}, \tag{31}$$

for iteration points and function values with $f_0 := f(x_0)$, and the so-called improvement in the signal-to-noise ratio (ISNR)

$$\text{ISNR} := 20\log_{10}\left(\frac{\|y - x_0\|_F}{\|x - x_0\|_F}\right) \tag{32}$$

and the peak signal-to-noise ratio (PSNR) defined by

$$\text{PSNR} := 20\log_{10}\left(\frac{255\sqrt{mn}}{\|x - x_0\|_F}\right), \tag{33}$$

where the pixel values are in $\{0, \ldots, 255\}$ and

$$\text{PSNR} := 20\log_{10}\left(\frac{\sqrt{mn}}{\|x - x_0\|_F}\right) \tag{34}$$

in which the pixel values are in [0, 1]. This definition indicates that a larger PSNR value means a smaller term $\|x - x_0\|_F$, i.e., a better quality for the restored image.

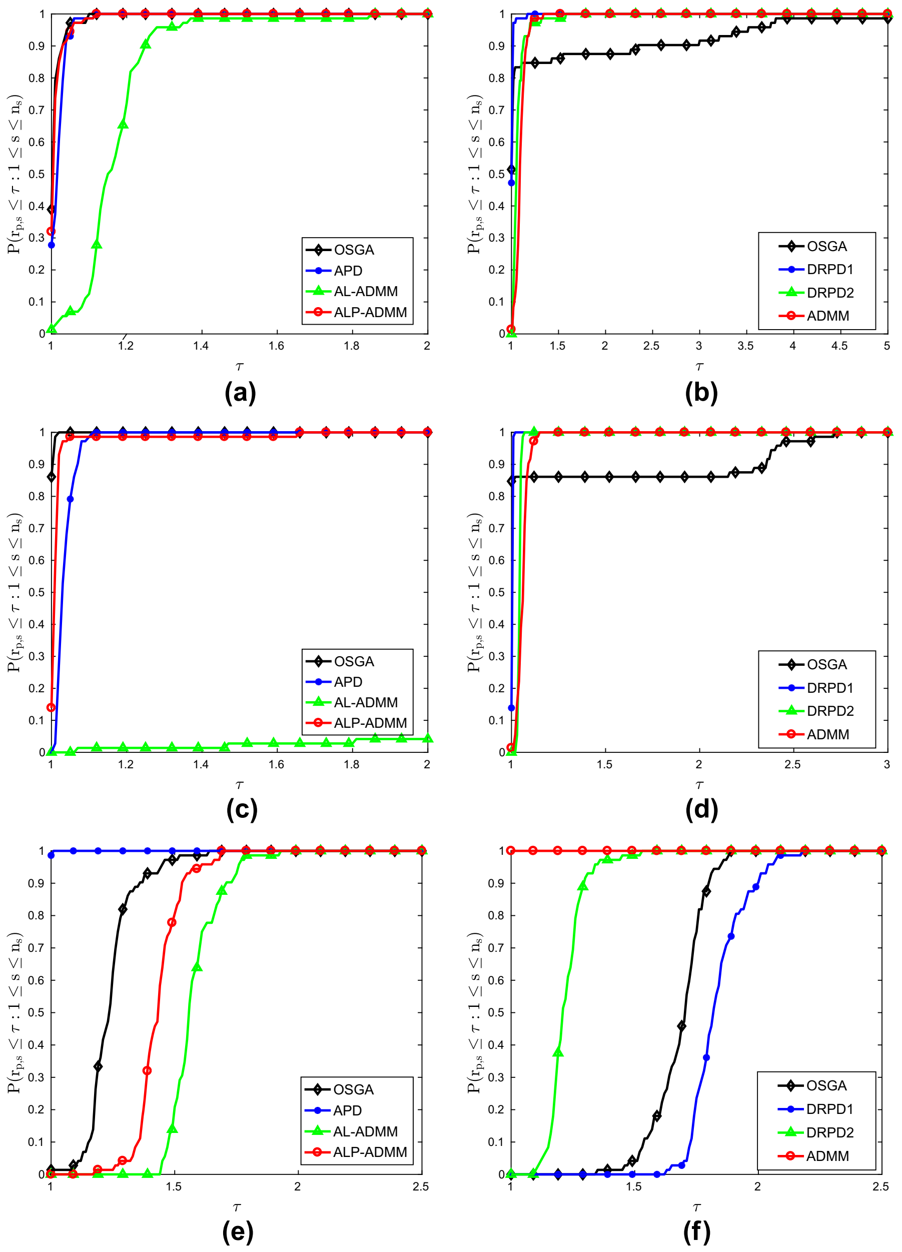

Fig. 1 Performance profiles for the PSNR, the best function value ($\widehat{f}$), and the running time (T) after 100 iterations of each algorithm: **a**, **c**, and **e** show APD, AL-ADMM, ALP-ADMM, and OSGA while **b**, **d**, and **f** show DRPD1, DRPD2, ADMM, and OSGA. **a** Performance profile for PSNR. **b** Performance profile for PSNR. **c** Performance profile for function values. **d** Performance profile for function values. **e** Performance profile for the running time. **f** Performance profile for the running time

Conversely, a smaller PSNR value indicates a larger term $\|x - x_0\|_F$ leading to a worse image quality.

We here study the deblurring problem using the optimization models L22ITVR and L1ITVR (see Example 1), where they are applied to problems involving additive and impulsive noise, respectively; see Nikolova (2004, 2002). We first consider the deblurring of a set of 72 images (see Table 5) using the L22ITVR model with $\lambda = 5 \times 10^{-2}$. We use APD [an accelerated primal-dual method proposed by Chen et al. (2014)], AL-ADMM and ALP-ADMM [two accelerated ADMM methods proposed by Chen et al. (2015)], and OSGA. We stop the algorithms after 100 iterations. We display the results of these reconstructions via performance profiles in subfigures (a), (c), and (e) of Fig. 1. Next, we consider the deblurring of a set of 72 images (see Table 5) using the L1ITVR model with $\lambda = 5 \times 10^{-2}$. We use DRPD1, DRPD2 [two Douglas–Rachford primal-dual methods proposed by BoŢ and Hendrich (2013b)], ADMM [an alternating direction method proposed by Chan et al. (2013)], and OSGA. In this case, the blurred/noisy image is generated by a 7×7 Gaussian kernel with standard deviation 5 and salt-and-pepper impulsive noise with noise density 0.4. The algorithms are stopped after 100 iterations. We illustrate the results of these reconstructions via performance profiles in subfigures (b), (d), and (f) of Fig. 1.

The results of subfigure (a) of Fig. 1 show that APD, ALP-ADMM, and OSGA are competitive regarding PSNR; however, OSGA slightly outperforms the others. Subfigure (c) shows that OSGA behaves considerably better than the others in terms of attaining the best function value. In subfigure (c), APD requires less time than AL-ADMM, ALP-ADMM, and OSGA, but OSGA is in second place. The results of subfigure (b) of Fig. 1 show that DRPD1 and OSGA are comparable with respect to PSNR, but OSGA attains slightly better results for 85% of the problems; however, it performs poorly for the remaining 15%. It can be seen from subfigure (d) that OSGA outperforms the others in terms of the best function value. In subfigure (f), while ADMM and DRPD2 have better running times among the others, they are not competitive with OSGA with respect to PSNR and the best function value.

To visualize the results of OSGA with L22ITVR, we consider the 512×512 blurred/noisy fingerprint image, reconstructing it using L22ITVR with APD, AL-ADMM, ALP-ADMM, and OSGA. We consider three different values for the regularization parameter $\lambda = 5 \times 10^{-1}, 10^{-1}, 5 \times 10^{-2}$, and we stop after 100 iterations. We report the results in Table 1. The blurred/noisy image is constructed following the above procedure for L22ITVR, and the results are summarized in subfigures (a)–(f) of Fig. 2 for $\lambda = 5 \times 10^{-2}$. The quality of the recovered image by OSGA seems acceptable (PSNR = 32.51). Its δ_1 evolution is comparable with AL-ADMM, while the δ_2 and ISNR evolution are competitive with ALP-ADMM.

To visualize the results of OSGA with L1ITVR, we consider the deblurring of the 1024×1024 blurred/noisy pirate image and restore it using the L1ITVR model with DRPD1, DRPD2, ADMM, and OSGA. We consider three different values for the regularization parameter $\lambda = 5 \times 10^{-1}, 10^{-1}, 5 \times 10^{-2}$, and we stop after 100 iterations. We report the results in Table 2. The blurred/noisy image is constructed following the above procedure for L1ITVR, and the results are summarized in

Table 1 Deblurring the 512 × 512 fingerprint image with APD, AL-ADMM, ALP-ADMM, and OSGA using L22ITVR for several regularization parameters

λ	APD			AL-ADMM			ALP-ADMM			OSGA		
	$\widehat{f}$	T	$PSNR$	$\widehat{f}$	T	$PSNR$	$\widehat{f}$	T	$PSNR$	$\widehat{f}$	T	$PSNR$
5e−1	2,802,704	**3.51**	20.80	15,775,179	5.33	16.54	2,020,669	4.68	22.77	**1,932,494**	3.98	**22.94**
1e−1	577,770	**3.69**	23.88	7,748,366	5.35	17.78	459,964	4.14	**25.53**	**459,672**	4.30	25.32
5e−2	294,584	**3.46**	25.09	4,620,987	5.26	18.76	**246,560**	4.67	**26.50**	246,778	4.18	**26.50**

T and $\widehat{f}$ are the running time and the best function value found, respectively, PSNR is given by (33), and the best results are displayed in bold

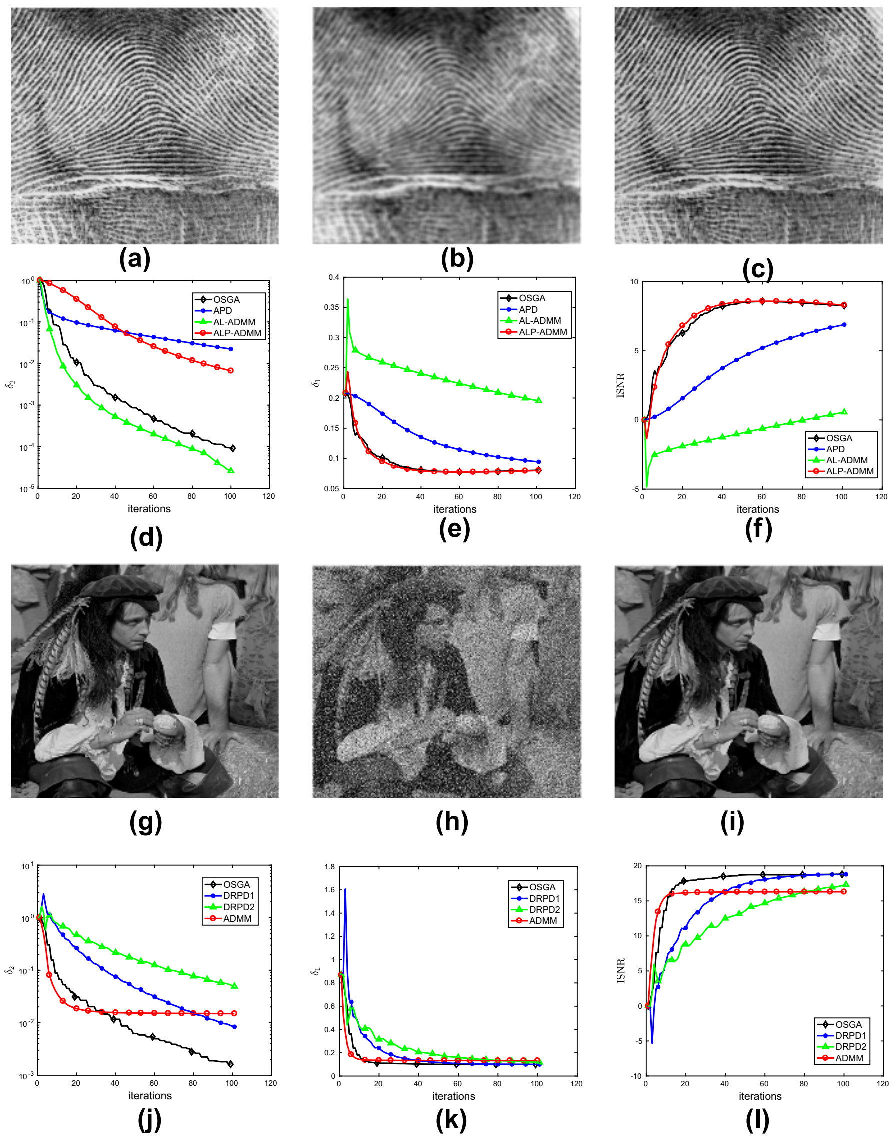

Fig. 2 **a–f** give a comparison of APD, AL-ADMM, ALP-ADMM, and OSGA for the deblurring of the 512×512 fingerprint image with 9×9 uniform blur and Gaussian noise with SNR $= 40$ dB. We set $\lambda = 10^{-2}$, and all algorithms are stopped after 100 iterations. **g–l** give a comparison of DRPD1, DRPD2, ADMM, and OSGA for the deblurring of the 1024×1024 pirate image with the 7×7 Gaussian kernel with standard deviation 5 and salt-and-pepper impulsive noise with noise density 0.4. We set $\lambda = 5 \times 10^{-2}$, and all algorithms are stopped after 100 iterations. **a** Clean image. **b** Blurred/noisy image. **c** OSGA: PSNR $= 32.51$, $T = 6.73$. **d** δ_2 versus iterations. **e** δ_1 versus iterations. **f** ISNR versus iterations. **g** Clean image. **h** Blurred/noisy image. **i** OSGA: PSNR $= 27.66$, $T = 13.82$. **j** δ_2 versus iterations. **k** δ_1 versus iterations. **l** ISNR versus iterations

Table 2 Deblurring the 1024×1024 pirate image with DRPD1, DRPD2, ADMM, and OSGA using L1ITVR for several regularization parameters

λ	DRPD1			DRPD2			ADMM			OSGA		
	$\widehat{f}$	T	$PSNR$	$\widehat{f}$	T	$PSNR$	$\widehat{f}$	T	$PSNR$	$\widehat{f}$	T	$PSNR$
5e−1	221,001	13.18	27.64	229,169	8.34	26.14	233,304	**7.36**	25.11	**219,797**	13.68	**27.65**
1e−1	221,207	13.57	27.63	229,396	8.19	26.13	233,494	**7.20**	25.10	**220,036**	14.32	**27.68**
5e−2	221,060	13.88	27.63	229,172	8.65	26.15	233,394	**7.15**	25.13	**219,922**	13.82	**27.66**

T and $\widehat{f}$ are the running time and the best function value found, respectively, PSNR is given by (33), and the best results are displayed in bold subfigures (g)–(l) of Fig. 2. This figure shows that the best δ_1 and δ_2 evolution are attained by OSGA.

3.2 Elastic net minimization

We now give a comparison of some first-order algorithms for the elastic net problem, where the subgradient-type methods use the nonsmooth first-order oracle and the proximal-type methods employ the smooth first-order oracle and a proximity operator. We compare OSGA with PGA (proximal gradient algorithm; Parikh and Boyd 2013), NSDSG (nonsummable diminishing subgradient algorithm; Boyd et al. 2003), FISTA (Beck and Tebolle's fast proximal gradient algorithm; Beck and Teboulle 2009a), NESCO (Nesterov's composite optimal algorithm; Nesterov 2013), NESUN (Nesterov's universal gradient algorithm; Nesterov 2015), NES83 (Nesterov's 1983 optimal algorithm; Nesterov 1983), NESCS (Nesterov's constant step optimal algorithm; Nesterov 2004), and NES05 (Nesterov's 2005 optimal algorithm; Nesterov 2005). The codes are written in MATLAB, and the original parameters of the related literature are used.

The algorithms NES83, NESCS, and NES05 were originally proposed to solve smooth convex problems, where they use the smooth first-order oracle and attain the optimal complexity $\mathcal{O}(\varepsilon^{-1/2})$. Although obtaining the optimal complexity bound of smooth problems is computationally interesting, problem (6) is typically nonsmooth. Recently, Lewis and Overton (2013) investigated the behavior of the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method for nonsmooth nonconvex problems, and their results are interesting. In addition, Nesterov (2013) showed that subgradients of composite functions preserve the most important features of gradients of smooth convex functions. These facts motivate our computational investigation of the behavior of optimal smooth first-order methods by passing the nonsmooth first-order oracle, providing a function value and a subgradient. In particular, we consider Nesterov's 1983 algorithm (see Nesterov 1983) and adapt it to solve (6) by simply passing a subgradient of the function as described in Algorithm 4.

Algorithm 4: NES83 (Nesterov's 1983 algorithm for multi-term composite functions)

```
Input: select $z$ and $y_0$ such that $z \neq y_0$ and $g_{y_0} \neq g_z$, $\rho \in ]0,1[$, $\varepsilon > 0$;
Output: $x_k$, $f_{x_k}$;
begin
    $a_0 = 0$;  $x_{-1} = y_0$; compute $g_{y_0}$ and $g_z$; $\alpha_{-1} = \|y_0 - z\|/\|g_{y_0} - g_z\|$;
    while stopping criteria do not hold do
        $\hat{\alpha}_k = \alpha_{k-1}$; $\hat{x}_k = y_k - \hat{\alpha}_k g_{y_k}$;
        compute $f_{\hat{x}_k}$;
        while $f_{\hat{x}_k} < f_{y_k} - \frac{1}{2}\hat{\alpha}_k \|g_{y_k}\|^2$ do
            $\hat{\alpha}_k = \rho\hat{\alpha}_k$; $\hat{x}_k = y_k - \hat{\alpha}_k g_{y_k}$; compute $f_{\hat{x}_k}$;
        end
        $x_k = \hat{x}_k$; $f_{x_k} = f_{\hat{x}_k}$; $\alpha_k = \hat{\alpha}_k$; $a_{k+1} = \left(1 + \sqrt{4a_k^2 + 1}\right)/2$;
        $y_{k+1} = x_k + (a_k - 1)(x_k - x_{k-1})/a_{k+1}$; compute $g_{y_{k+1}}$; $k = k + 1$
    end
end
```

Similarly to NES83, the methods NESCS and NES05 are adapted from Nesterov's constant step (Nesterov 2004) and 2005 algorithms (Nesterov 2005), respectively. The adapted versions of NES83, NESCS, and NES05 are able to deal with nonsmooth problems as well; however, there is to date no theory to support their convergence.

We compare OSGA with the subgradient-type methods (NSDSG, NES83, NESCS, NES05) and the proximal-type methods (PGA, NSDSG, FISTA, NESCO, NESUN) in separate comparisons. Most of these methods need to know about Lipschitz constants to determine a step-size. While NESCS, NES05, PGA, and FISTA use the constant L to determine a step-size, NESCO and NESUN start with a lower estimation of L and adapt it by a backtracking line search; see Nesterov (2013, 2015) for more details. In our implementation, similarly to that of Becker et al. (2011), NESCO and NESUN use the initial estimate

$$L_0 := \frac{\|g_{x_0} - g_{z_0}\|_*}{\|x_0 - z_0\|} \tag{35}$$

for the Lipschitz constant of gradients ($x_0 \neq z_0$ and $g_{x_0} \neq g_{z_0}$ in which g_{x_0} and g_{z_0} are gradients of f at arbitrary points x_0 and z_0). In Nesterov (1983), Nesterov used a similar term to determine an initial step-size for NES83. For NSDSG, the step-size is computed by $\alpha_0(k)^{-1/2}$, where k is the iteration counter and $\alpha_0 > 0$ is a constant that should be as large as possible such that the algorithm is not divergent; see Boyd et al. (2003).

Let us consider an underdetermined system $Ax = y$ with a $m \times n$ random matrix A and a random m-vector y. This problem arises in many applications in which the goal is to determine x. Considering the ill-conditioned nature of this problem, the most popular optimization models are (2), (5), and (8), where (2) is smooth and (5) and (8) are nonsmooth. We are particularly interested in the multi-term problem (8) with both dense and sparse data. For (8), we set $\mathcal{W} = I$, where I is the identity operator. For $m = 2000$ and $n = 5000$, the dense data (A, y, x_0) is randomly generated by the "rand" function in MATLAB, and the sparse data (A, y, x_0) by the "sprand" function. For both cases, we solve the problem by 1000 iterations of OSGA, save the best function value f_b, and stop the other methods as soon as a function value less than or equal to f_b is attained or if the maximum number of iterations (10,000) is reached. In our comparison, we use $\widehat{L} := \max_{1 \leq i \leq n} \|a_i\|^2$, where a_i $(i = 1, 2, \ldots, n)$ is the ith column of A. In the implementation, NESCS, NES05, PGA, and FISTA use $L = 10^4 \widehat{L}$ and $L = 10^2 \widehat{L}$ for dense and sparse problems, respectively. Moreover, NSDSG employs $\alpha_0 = 10^{-7}$ and $\alpha_0 = 10^{-5}$ for the dense and sparse problems, respectively. We consider three levels of regularization parameters. We perform 50 runs for each parameter setting and report the averages in Tables 3 and 4. We also display the performance profiles of the running time for all 150 runs in Fig. 3.

In Table 3, the best running time is given by OSGA, while the running times of NES83, NES05, and OSGA are compared in Table 4. The performance profile of the running time in subfigures (a) and (b) of Fig. 3 shows that OSGA outperforms the

Table 3 Averages of N_i and T for PGA, FISTA, NESCO, NESUN, and OSGA for the elastic net problem (8) with several regularization parameters, where N_i and T are the number of iterations and the running time, respectively

Data	λ	OSGA		PGA		FISTA		NESCO		NESUN	
		N_i	T	N_i	T	N_i	T	N_i	T	N_i	T
Dense	10^{-1}	1000.00	**11.27**	10,000.00	69.59	1848.98	19.24	**771.86**	23.03	868.60	17.89
	10^{-2}	1000.00	**11.02**	10,000.00	62.36	18.73.56	17.17	**792.10**	20.30	923.64	16.41
	10^{-4}	1000.00	**10.88**	10,000.00	62.38	1856.20	16.74	**784.98**	20.14	916.66	16.33
Sparse	10^{-1}	**1000.00**	**2.52**	10,000.00	15.16	8965.36	18.29	10,000.00	267.52	10,000.00	253.08
	10^{-2}	**1000.00**	**2.58**	10,000.00	14.77	10,000.00	20.21	10,000.00	269.62	1119.86	29.12
	10^{-4}	1000.00	**2.58**	10,000.00	14.67	9960.10	20.05	**654.80**	19.04	677.58	18.67

Each row of this table is the average of results for 50 runs of the algorithms with random data, and the best results are displayed in bold

Table 4 Averages of N_i and T for NSDSG, NES83, NESCS, NES05, and OSGA for the scaled elastic net problem (8) with several regularization parameters, where N_i and T are the number of iterations and the running time, respectively

Data	λ	OSGA		NSDSG		NES83		NESCS		NES05	
		N_i	T	N_i	T	N_i	T	N_i	T	N_i	T
Dense	10^{-1}	**1000.00**	**11.12**	10,000.00	64.16	1430.40	13.67	6754.78	62.04	1856.82	11.20
	10^{-2}	**1000.00**	11.02	10,000.00	62.98	1449.30	13.42	8376.04	74.50	1881.88	**10.64**
	10^{-4}	**1000.00**	10.89	10,000.00	61.18	1434.72	12.87	8402.48	69.68	1864.50	**9.79**
Sparse	10^{-1}	1000.00	2.65	10,000.00	13.02	**845.18**	**1.76**	4277.86	7.97	8974.74	12.34
	10^{-2}	**1000.00**	**2.67**	10,000.00	12.99	1593.70	3.19	10,000.00	18.41	10,000.00	13.58
	10^{-4}	**1000.00**	2.58	10,000.00	12.38	1144.84	**2.28**	10,000.00	17.86	9964.28	13.07

Each row of this table is the average of results for 50 runs of the algorithms with random data, and the best results are displayed in bold

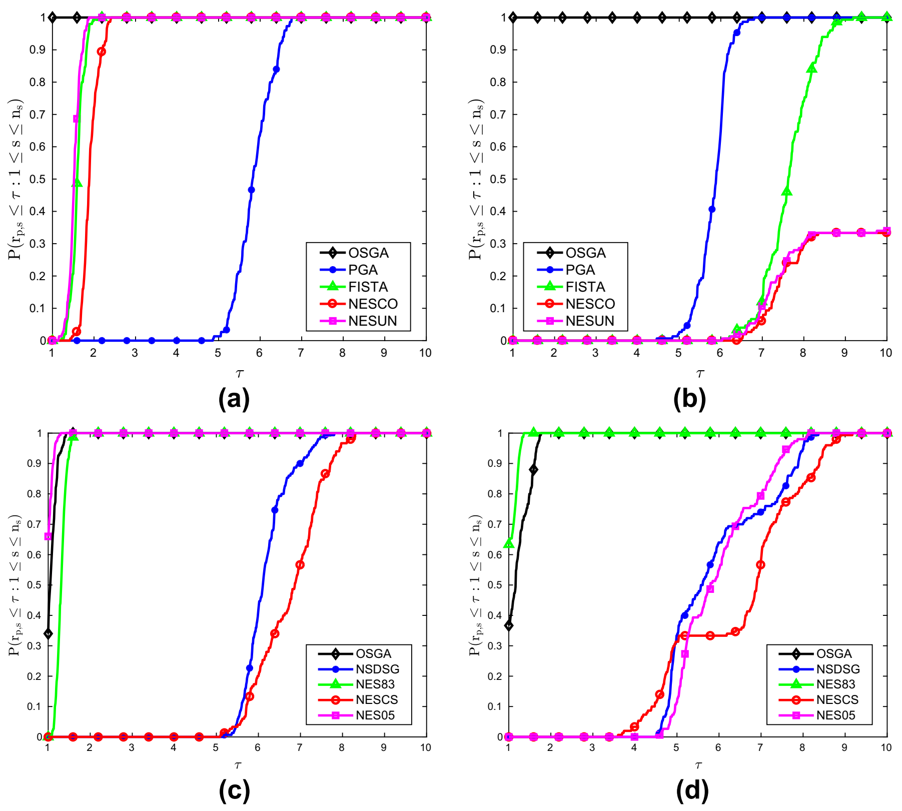

Fig. 3 Performance profiles for the running time (T) for the elastic net problem (8): **a**, **b** show the proximal-type methods PGA, FISTA, NESCO, NESUN, and OSGA with dense and sparse data, respectively; **c**, **d** show the subgradient-type methods NSDSG, NES83, NESCS, NES05, and OSGA with dense and sparse data, respectively. **a** Performance profile for the running time, dense data. **b** Performance profile for the running time, sparse data. **c** Performance profile for the running time, dense data. **d** Performance profile for the running time, sparse data

others substantially; but in subfigure (c), NES83, NES05, and OSGA are competitive and in subfigure (d), NES83 and OSGA have the best running time.

To visualize the function value evolution of the methods, we display function values versus iterations for both dense and sparse data with $\lambda_1 = \lambda_2 = 10^{-1}, 10^{-2}, 10^{-4}$. In subfigures (a)–(e) of Fig. 4, for dense data, it can be seen that NESCO and NESUN are comparable with OSGA. It is notable that FISTA, NESCO, NESUN, and OSGA have much better results than PGA, which supports the theoretical results about the complexity of these methods. Subfigures (g)–(l) of Fig. 4 show that the adapted algorithms NES83, NESCS, and NES05 are surprisingly comparable with OSGA and their results are much better than those of NSDSG. Among the adapted algorithms, NES83 performs better than the others.

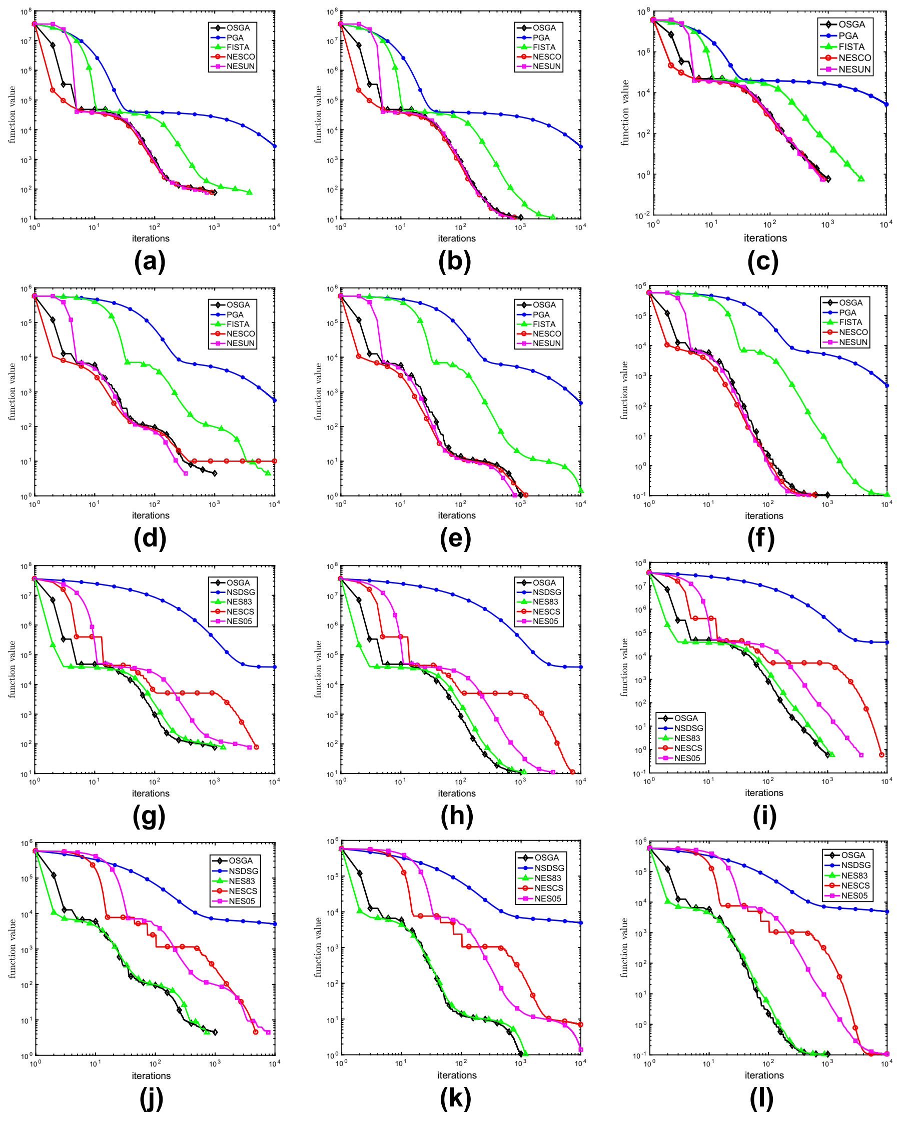

Fig. 4 Function values versus iterations for the elastic net problem (8) (single instance) with the regularization parameters $\lambda = 10^{-}, 10^{-2}, 10^{-4}$: **a–f** show the proximal-type methods PGA, FISTA, NESCO, NESUN, and OSGA, and **g–l** show the subgradient-type methods NSDSG, NES83, NESCS, NES05, and OSGA. **a** Dense data ($\lambda = 10^{-1}$). **b** Dense data ($\lambda = 10^{-2}$). **c** Dense data ($\lambda = 10^{-4}$). **d** Sparse data ($\lambda = 10^{-1}$). **e** Sparse data ($\lambda = 10^{-2}$). **f** Sparse data ($\lambda = 10^{-4}$). **g** Dense data ($\lambda = 10^{-1}$). **h** Dense data ($\lambda = 10^{-2}$). **i** Dense data ($\lambda = 10^{-4}$). **j** Sparse data ($\lambda = 10^{-1}$). **k** Sparse data ($\lambda = 10^{-2}$). **l** Sparse data ($\lambda = 10^{-4}$)

3.3 Sparse recovery with ℓ_1-minimization

Here, we consider the inverse problem (1) with $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, and $y \in \mathbb{R}^m$, where the observation is deficient, $m<n$, as a common interest in compressed sensing. Hence, the object x cannot be recovered from the observation y directly, even in the noiseless system $Ax = y$, unless there are additional assumptions such as sparsity of x. We consider the recovery of a sparse signal $x \in \mathbb{R}^n$ from $y \in \mathbb{R}^m$ with A obtaining by first filling it with independent samples from the standard Gaussian distribution and then orthonormalizing the rows; see Figueiredo et al. (2007). In our experiment, we consider $m = 2000$ and $n = 4000$. The data is generated as explained in the script “demo_continuation.m” of the gradient projection for sparse reconstruction (GPSR) (Figueiredo et al. 2007) package available at http://www.lx.it.pt/~mtf/GPSR/, where the original signal x consists of 300 randomly placed ± 1 spikes. We therefore solve (5) with the regularization parameters $\lambda = 10^{-1}\|Ay\|_\infty, 10^{-2}\|Ay\|_\infty, 10^{-3}\|Ay\|_\infty$. For the original signal x_0, the mean squared error (MSE) is used as a measure of the quality of a recovered signal, i.e.,

$$\mathrm{MSE} = \frac{1}{n}\|x - x_0\|_2^2. \tag{36}$$

We solve the problem by 100 iterations of OSGA, save the best function value f_b, and stop the other methods as soon as a function value less than or equal to f_b is attained or if the maximum number of iterations (10,000) is reached. The results are given in Fig. 5.

Figure 5 shows a comparison of OSGA and the proximal-type methods (PGA, FISTA, NESCO, NESUN) and the subgradient-type methods (NSDSG, NES83, NESCS, NES05) for the sparse signal recovery with ℓ_1-minimization, where we compare the algorithms with respect to function values and MSE. Subfigures (a)–(s) show that NESUN and OSGA are substantially better than the others; however, NESUN takes far more iterations. Moreover, NESCO and sometimes NESUN do not have good accuracy because the backtracking line searches are not terminated with a reasonable approximation of the Lipschitz constant. Subfigures (d)–(f) show that OSGA has the best MSE evolution. From subfigures (g)–(l), it can be seen that NES83 and OSGA are comparable but much better than the others with respect to function values and MSE evolution; however, OSGA requires fewer iterations for the same accuracy.

3.4 Efficiency of OSGA and adapted Nesterov-type optimal methods

As discussed in Sect. 1, OSGA does not need global information (e.g., Lipschitz constants for function values and gradients) of the underlying function of minimization, except for the strong convexity parameter if it is available. The parameters (28) of OSGA have been set so that they work well for a range of convex problems in the fields of signal and image processing, machine learning, and statistics; see Ahookhosh (2015a). Moreover, OSGA can be applied to any type of convex problem without knowledge of the problem structure, which is not the case

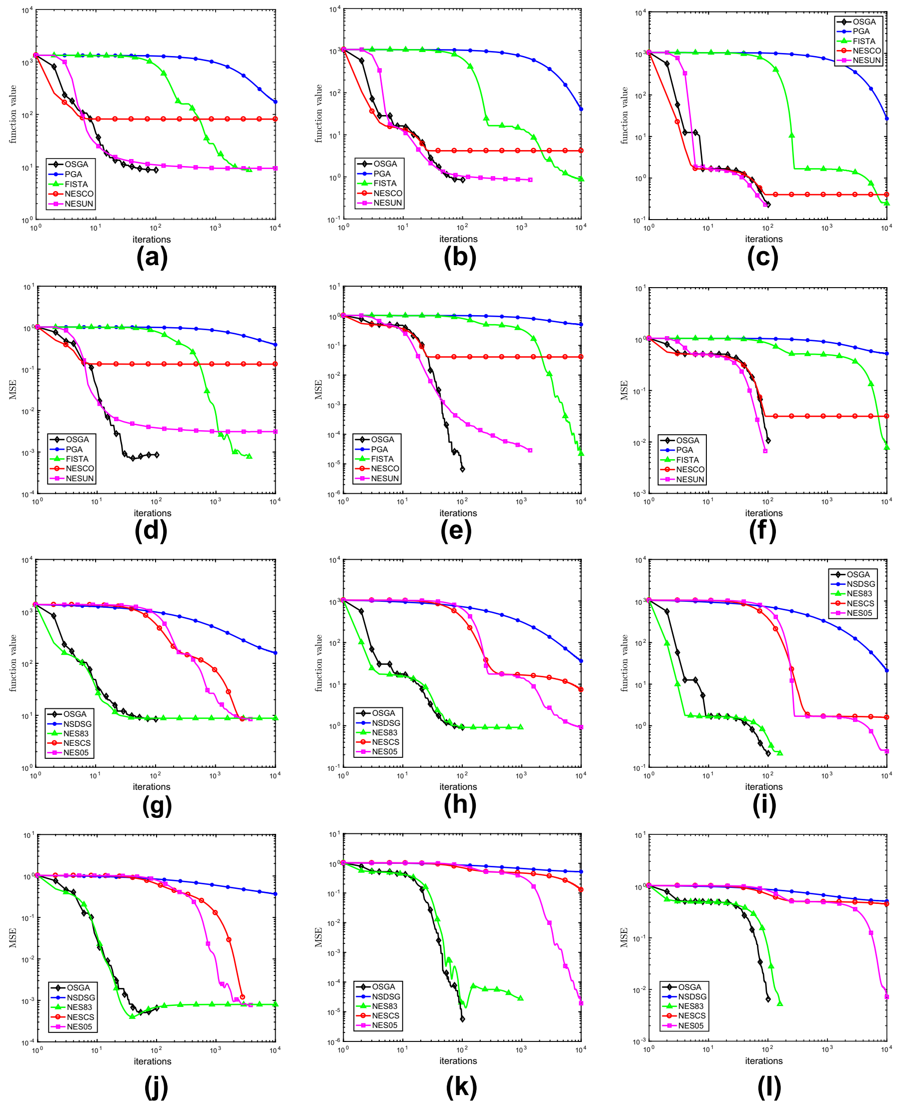

Fig. 5 Function values and MSE (36) versus iterations for a sparse recovery using the ℓ_1-minimization problem (5) (single instance) with three levels of regularization parameters $\lambda = 10^{-1}\|\mathcal{A}y\|_\infty, 10^{-2}\|\mathcal{A}y\|_\infty, 10^{-3}\|\mathcal{A}y\|_\infty$. **a**–**c** and **d**–**f** show the function values and MSE versus iterations of the proximal-type methods PGA, FISTA, NESCO, NESUN, and OSGA, respectively. **g**–**i** and **j**–**l** show the function values and MSE versus iterations of the subgradient-type methods NSDSG, NES83, NESCS, NES05, and OSGA, respectively. **a** Function values ($\lambda = 10^{-1}\|\mathcal{A}y\|_\infty$). **b** Function values ($\lambda = 10^{-2}\|\mathcal{A}y\|_\infty$). **c** Function values ($\lambda = 10^{-3}\|\mathcal{A}y\|_\infty$). **d** MSE ($\lambda = 10^{-1}\|\mathcal{A}y\|_\infty$). **e** MSE ($\lambda = 10^{-2}\|\mathcal{A}y\|_\infty$). **f** MSE ($\lambda = 10^{-3}\|\mathcal{A}y\|_\infty$). **g** Function values ($\lambda = 10^{-1}\|\mathcal{A}y\|_\infty$). **h** Function values ($\lambda = 10^{-2}\|\mathcal{A}y\|_\infty$). **i** Function values ($\lambda = 10^{-3}\|\mathcal{A}y\|_\infty$). **j** MSE ($\lambda = 10^{-1}\|\mathcal{A}y\|_\infty$). **k** MSE ($\lambda = 10^{-2}\|\mathcal{A}y\|_\infty$). **l** MSE ($\lambda = 10^{-3}\|\mathcal{A}y\|_\infty$)

for Nesterov-type optimal methods. For example, OSGA is applied to L22ITVR and L1ITVR successfully, while Nesterov-type optimal methods need a proximity operator for the isotropic total variation, which is only iteratively available; this requirement makes them expensive for such problems. Therefore, if an approximation of Lipschitz constants is not available or a closed-form solution for the proximal mapping of a function is not available, OSGA can be applied successfully. Further, our results show that if we have a good approximation of the Lipschitz constants and a closed-form solution for the proximal mapping of a function (e.g., the elastic net and ℓ-minimization), OSGA is competitive with the state-of-the-art solvers.

The adapted Nesterov-type optimal methods (NES83, NESCS, and NES05) applied to the elastic net and ℓ-minimization with their standard parameters have performed well for both problems. Surprisingly, NES83 outperforms the other Nesterov-type optimal methods and is competitive with OSGA, which shows its potential for nonsmooth problems.

4 Final remarks

We have studied the computational properties of OSGA for convex optimization problems with multi-term composite functions. OSGA has a simple structure, and it is flexible enough to handle general convex optimization with the optimal complexity for Lipschitz-continuous nonsmooth problems and smooth problems with Lipschitz-continuous gradients. It does not need to know global parameters such as Lipschitz constants except for the strong convexity parameter, which can be set to zero if it is unknown.

The main objective of this paper is the study of the computational behavior of OSGA for large-scale structured convex problems. We also adapt some first-order methods (originally proposed for smooth convex optimization) to nonsmooth problems: we simply pass a subgradient of the nonsmooth function to these methods in place of the gradient. We consider several examples of linear inverse problems and report extensive numerical results by comparing OSGA with the adapted methods and some state-of-the-art first-order methods. By these comparisons, it turns out that OSGA is efficient (in the sense of PSNR, ISNR, the number of function evaluation, and the running time) for such problems: its fast rate of convergence is often beyond the theoretical rate. While OSGA attains the complexity $\mathcal{O}(\varepsilon^{-2})$, surprisingly its computational performance is competitive with optimal first-order methods (FISTA, NESCO, and NESUN) with the complexity $\mathcal{O}(\varepsilon^{-1/2})$. Moreover, a promising performance of the adapted Nesterov-type optimal methods is observed, especially for NES83 that is comparable with OSGA for the elastic net and ℓ_1-minimization problems possibly because of its simple structure (it does not need to solve a subproblem, so requires less running time); however, the performance is not to date supported by theoretical results, so this might be an interesting topic for future research.

Acknowledgements Open access funding provided by University of Vienna. We would like to thank Arnold Neumaier for a careful reading of the manuscript and his insightful comments and feedback. We are grateful to Radu I. Boţ (Boţ and Hendrich 2013b), Guanghui Lan (Chen et al. 2014, 2015), Yuyuan Ouyang and Min Tao (Chan et al. 2013) for making the codes of the papers available. We are also grateful to the referees, associate editors, and editor in chief who helped us to considerably improve the quality of this paper.

Appendix

The test images and corresponding spatial resolutions for the deblurring problems L22ITVR and L1ITVR are given in Table 5.

Table 5 List of images and spatial resolutions for deblurring problems L22ITVR and L1ITVR

Image	Spatial res.	Image	Spatial res.	Image	Spatial res.
Aerial	256 × 256	Crowd 1	512 × 512	Tank 1	512 × 512
Airplane	256 × 256	Crowd 2	512 × 512	Tank 2	512 × 512
Cameraman	256 × 256	Darkhair woman	512 × 512	Tank 3	512 × 512
Chemical plant	256 × 256	Dollar	512 × 512	Truck	512 × 512
Clock	256 × 256	Elaine	512 × 512	Truck APCs 1	512 × 512
Fingerprint 1	256 × 256	Fingerprint	512 × 512	Truck APCs 2	512 × 512
Lena	256 × 256	Flintstones	512 × 512	Washington 1	512 × 512
Moon surface	256 × 256	Girlface	512 × 512	Washington 2	512 × 512
Pattern 1	256 × 256	Goldhill	512 × 512	Washington 3	512 × 512
Pattern 2	256 × 256	Head CT	512 × 512	Washington 4	512 × 512
Star	256 × 256	Houses	512 × 512	Zelda	600 × 600
Aerial	512 × 512	Kiel	512 × 512	Dark blobs 1	600 × 600
Airfield	512 × 512	Lake	512 × 512	Dark blobs 2	600 × 600
Airplane 1	512 × 512	Lena	512 × 512	House	600 × 600
Airplane 1	512 × 512	Lena numbers	512 × 512	Ordered matches	600 × 600
APC	512 × 512	Liftingbody	512 × 512	Random matches	600 × 600
Barbara	512 × 512	Lighthouse	512 × 512	Rice	600 × 600
Blobs	512 × 512	Livingroom	512 × 512	Shepp-logan phantom	600 × 600
Blonde woman	512 × 512	Mandril	512 × 512	Airport	1024 × 1024
Boat	512 × 512	MRI spine	512 × 512	Pentagon	1024 × 1024
Cameraman	512 × 512	Peppers	512 × 512	Pirate	1024 × 1024
Car APCs 1	512 × 512	Pirate	512 × 512	Rose	1024 × 1024
Car APCs 2	512 × 512	Smiling woman	512 × 512	Testpat	1024 × 1024
Clown	512 × 512	Squares	512 × 512	Washington	1024 × 1024

References

Ahookhosh M (2015a) High-dimensional nonsmooth convex optimization via optimal subgradient methods. PhD Thesis, Faculty of Mathematics, University of Vienna

Ahookhosh M (2015b) User's manual for OSGA (Optimal SubGradient Algorithm). http://homepage.univie.ac.at/masoud.ahookhosh/uploads/User's_manual_for_OSGA.pdf

Ahookhosh M, Neumaier A (2013) High-dimensional convex optimization via optimal affine subgradient algorithms. In: ROKS workshop, pp 83–84

Ahookhosh M, Neumaier A (2017a) An optimal subgradient algorithms for large-scale bound-constrained convex optimization. Math Methods Oper Res 86(1):123–147

Ahookhosh M, Neumaier A (2017b) Optimal subgradient algorithms for large-scale convex optimization in simple domains. Numer Algorithms 76(4):1071–1097

Ahookhosh M, Neumaier A (2017c) Solving structured nonsmooth convex optimization with complexity $O(\varepsilon ^{-1/2})$. TOP. https://doi.org/10.1007/s11750-017-0462-3

Ahookhosh M, Neumaier A (2018) An optimal subgradient algorithm with subspace search for costly convex optimization problems. http://www.optimization-online.org/DB_HTML/2015/04/4852.html **(submitted)**

Andrews H, Hunt B (1977) Digital image restoration. Englewood Cliffs, Prentice-Hall

Beck A, Teboulle M (2003) Mirror descent and nonlinear projected subgradient methods for convex optimization. Oper Res Lett 31:167–175

Beck A, Teboulle M (2009a) A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J Imaging Sci 2:183–202

Beck A, Teboulle M (2009b) Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. IEEE Trans Image Process 18(11):2419–2434

Becker SR, Candès EJ, Grant MC (2011) Templates for convex cone problems with applications to sparse signal recovery. Math Program Comput 3:165–218

Bertero M, Boccacci P (1998) Introduction to inverse problems in imaging. IOP, Bristol

Bot RI, Hendrich C (2013a) A double smoothing technique for solving unconstrained nondifferentiable convex optimization problems. Comput Optim Appl 54(2):239–262

Bot RI, Hendrich C (2013b) A Douglas–Rachford type primal-dual method for solving inclusions with mixtures of composite and parallel-sum type monotone operators. SIAM J Optim 23(4):2541–2565

Bot RI, Hendrich C (2015) On the acceleration of the double smoothing technique for unconstrained convex optimization problems. Optimization 64(2):265–288

Boyd S, Xiao L, Mutapcic A (2003) Subgradient methods. http://www.stanford.edu/class/ee392o/subgrad_method.pdf

Bruckstein AM, Donoho DL, Elad M (2009) From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM Rev 51(1):34–81

Candés E (2006) Compressive sampling. In: Proceedings of international congress of mathematics, vol 3, Madrid, Spain, pp 1433–1452

Chan RH, Tao M, Yuan X (2013) Constrained total variation deblurring models and fast algorithms based on alternating direction method of multipliers. SIAM J Imaging Sci 6(1):680–697

Chen S, Donoho DL, Saunders MA (2001) Atomic decomposition by basis pursuit. SIAM Rev 43(1):129–159

Chen Y, Lan G, Ouyang Y (2014) Optimal primal-dual methods for a class of saddle point problems. SIAM J Optim 24(4):1779–1814

Chen Y, Lan G, Ouyang Y (2015) An accelerated linearized alternating direction method of multipliers. SIAM J Imaging Sci 8(1):644–681

Dolan E, Moré JJ (2002) Benchmarking optimization software with performance profiles. Math Program 91:201–213

Donoho DL (2006) Compressed sensing. IEEE Trans Inf Theory 52(4):1289–1306

Figueiredo MAT, Nowak RD, Wright SJ (2007) Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. IEEE J Sel Top Signal Proces 1(4):586–597

Gonzaga CC, Karas EW (2013) Fine tuning Nesterov's steepest descent algorithm for differentiable convex programming. Math Program 138:141–166

He N, Juditsky A, Nemirovski A (2015) Mirror Prox algorithm for multi-term composite minimization and semi-separable problems. Comput Optim Appl 61:275–319

Kaufman L, Neumaier A (1996) PET regularization by envelope guided conjugate gradients. IEEE Trans Med Imaging 15:385–389

Lan G (2015) Bundle-level type methods uniformly optimal for smooth and nonsmooth convex optimization. Math Program 149:1–45

Lewis AS, Overton ML (2013) Nonsmooth optimization via quasi-Newton methods. Math Program 141:135–163

Nemirovsky AS, Yudin DB (1983) Problem complexity and method efficiency in optimization. Wiley, New York

Nesterov Y (1983) A method of solving a convex programming problem with convergence rate $O(1/k^2)$. Dokl AN SSSR 269: 543–547 **(in Russian)**. English translation: Sov Math Dokl 27:372–376

Nesterov Y (2004) Introductory lectures on convex optimization: a basic course. Kluwer, Dordrecht

Nesterov Y (2005) Smooth minimization of non-smooth functions. Math Program 103:127–152

Nesterov Y (2013) Gradient methods for minimizing composite objective function. Math Program 140:125–161

Nesterov Y (2015) Universal gradient methods for convex optimization problems. Math Program 152(1):381–404

Neumaier A (1998) Solving ill-conditioned and singular linear systems: a tutorial on regularization. SIAM Rev 40(3):636–666

Neumaier A (2016) OSGA: a fast subgradient algorithm with optimal complexity. Math Program 158(1):1–21

Nikolova M (2002) Minimizers of cost-functions involving nonsmooth data-fidelity terms. SIAM J Numer Anal 40(3):965–994

Nikolova M (2004) A variational approach to remove outliers and impulse noise. J Math Imaging Vis 20:99–120

Osher S, Rudin L, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. Physica 60:259–268

Parikh N, Boyd S (2013) Proximal algorithms. Found Trends Optim 1(3):123–231

Tibshirani R (1996) Regression shrinkage and selection via the Lasso. J R Stat Soc 58(1):267–288

Tikhonov AN (1963) Solution of incorrectly formulated problems and the regularization method. Sov Math Dokl 4:1035–1038