SIAM J. IMAGING SCIENCES
Vol. xx, pp. x
© xxxx Society for Industrial and Applied Mathematics

MAGMA: Multi-level accelerated gradient mirror descent algorithm for large-scale convex composite minimization

Vahan Hovhannisyan*, Panos Parpas*, and Stefanos Zafeiriou*

Abstract. Composite convex optimization models arise in several applications, and are especially prevalent in inverse problems with a sparsity inducing norm and in general convex optimization with simple constraints. The most widely used algorithms for convex composite models are accelerated first order methods, however they can take a large number of iterations to compute an acceptable solution for large-scale problems. In this paper we propose to speed up first order methods by taking advantage of the structure present in many applications and in image processing in particular. Our method is based on multi-level optimization methods and exploits the fact that many applications that give rise to large scale models can be modelled using varying degrees of fidelity. We use Nesterov's acceleration techniques together with the multi-level approach to achieve an $\mathcal{O}(1/\sqrt{\epsilon})$ convergence rate, where ϵ denotes the desired accuracy. The proposed method has a better convergence rate than any other existing multi-level method for convex problems, and in addition has the same rate as accelerated methods, which is known to be optimal for first-order methods. Moreover, as our numerical experiments show, on large-scale face recognition problems our algorithm is several times faster than the state of the art.

1. Introduction.

Composite convex optimization models consist of the optimization of a convex function with Lipschitz continuous gradient plus a non-smooth term. They arise in several applications, and are especially prevalent in inverse problems with a sparsity inducing norm and in general convex optimization with simple constraints (e.g. box or linear constraints). Applications include signal/image reconstruction from few linear measurements (also referred to as compressive or compressed sensing) [14, 17, 18, 21], image super-resolution [52], data classification [48, 49], feature selection [43], sparse matrix decomposition [17], trace norm matrix completion [16], image denoising and deblurring [23], to name a few.

Given the large number of applications it is not surprising that several specialized algorithms have been proposed for convex composite models. Second order methods outperform all other methods in terms of the number of iterations required. Interior point methods [10], Newton and truncated Newton methods [29], primal and dual Augmented Lagrangian methods [51] have all been proposed. However, unless the underlying application posses some specific sparsity pattern second order methods are too expensive to use in applications that arise in practice. As a result the most widely used methods are first order methods. Following the development of accelerated first order methods for the smooth case [37] and the adaptation of accelerated first order methods to the non-smooth case ([5, 36]) there has been a large amount of research in this area. State of the art methods use only first order information and as a result (even when accelerated) take a large number of iterations to compute an acceptable solution for large-scale problems. Applications in image processing can have millions of variables and therefore new methods that take advantage of problem structure are needed.

We propose to speed up first order methods by taking advantage of the structure present in many applications and in image processing in particular. The proposed methodology uses

*Department of Computing, Imperial College London, 180 Queen's Gate, SW7 2AZ London, UK (vh13@imperial.ac.uk, p.parpas@imperial.ac.uk, s.zafeiriou@imperial.ac.uk).

© xxxx Society for Industrial and Applied Mathematics

the fact that many applications that give rise to large scale models can be modelled using varying degrees of fidelity. The typical applications of convex composite optimization include dictionaries for learning problems, images for computer vision applications, or discretization of infinite dimensional problems appearing in image processing [3]. First order methods use a quadratic model with first order information and the Lipschitz constant to construct a search direction. We propose to use the solution of a low dimensional (not necessarily quadratic) model to compute a search direction. The low dimensional model is not just a low fidelity model of the original model but it is constructed in a way so that is consistent with the high fidelity model.

The method we propose is based on multi-level optimization methods. A multi-level method for solving convex infinite-dimensional optimization problems was introduced in [12] and later further developed by Nash in [33]. Although [33] is one of the pioneering works that uses a multi-level algorithm in an optimization context, it is rather abstract and only gives a general idea of how an optimization algorithm can be used in a multi-level way. The author proves that the algorithm converges under certain mild conditions. Extending the key ideas of Nash's multi-level method in [46], Wen and Goldfarb used it in combination with a line search algorithm for solving discretized versions of unconstrained infinite-dimensional smooth optimization problems. The main idea is to use the solution obtained from a coarse level for calculating the search direction in the fine level. On each iteration the algorithm uses either the negative gradient direction on the current level or a direction obtained from coarser level models. They prove that either search direction is a descent direction using Wolfe-Armijo and Goldstein-like stopping rules in their backtracking line search procedure. Later a multi-level algorithm using the trust region framework was proposed in [26]. In all those works the objective function is assumed to be smooth, whereas the problems addressed in this paper are not. We also note that multi-level optimization methods have been applied to PDE optimization for many years. A review of the latter literature appeared in [11].

The first multi-level optimization algorithm covering the non-smooth convex composite case was introduced in [41]. It is a multi-level version of the well-known Iterative Shrinkage-Thresholding Algorithm (ISTA) ([5], [19], [20]), called MISTA. In [41] the authors prove R-linear convergence rate of the algorithm and demonstrate in several image deblurring examples that MISTA can be up to 3-4 times faster than state of the art methods. However, MISTA's R-linear convergence rate is not optimal for first order methods and thus it has the potential for even better performance. Motivated by this possible improvement, in this paper we apply Nesterov's acceleration techniques on MISTA, and show that it is possible to achieve $\mathcal{O}(1/\sqrt{\epsilon})$ convergence rate, where ϵ denotes the desired accuracy. To the best of our knowledge this is the first multi-level method that has an optimal rate. In addition to the proof that our method is optimal, we also show in numerical experiments that, for large-scale problems it can be up to 10 times faster than the current state of the art.

One very popular recent approach for handling very large scale problems are stochastic coordinate descent methods [24, 31]. However, while being very effective for ℓ_1-regularized least squares problems in general, stochastic gradient methods are less applicable for problems with highly correlated data, such as the face recognition problem and other pattern recognition problems, as well as in dense error correction problems with highly correlated dictionaries [47]. We compare our method both with accelerated gradient methods (FISTA) as well as stochastic

© xxxx Society for Industrial and Applied Mathematics

block-coordinate methods (APCG [24]).

Our convergence proof is based on the proof of Nesterov's method given in [1], where the authors showed that optimal gradient methods can be viewed as a linear combination of primal gradient and mirror descent steps. We show that for some iterations (especially far away from the solution) it is beneficial to replace gradient steps with coarse correction steps. The coarse correction steps are computed by iterating on a carefully designed coarse model and a different step-size strategy. Since our algorithm combines concepts from multilevel optimization, gradient and mirror descent steps we call it Multilevel Accelerated Gradient Mirror Descent Algorithm (MAGMA). The proposed method converges in function value with a faster rate than any other existing multi-level method for convex problems, and in addition has the same rate as accelerated methods which is known to be optimal for first order methods [34]. However, given that we use the additional structure of problems that appear in imaging applications in practice our algorithm is several times faster than the state of the art.

The rest of this paper is structured as follows: in Section 2 we give mathematical definitions and describe the algorithms most related to MAGMA. Section 3 is devoted to presenting a multi-level model as well as our multi-level accelerated algorithm with its convergence proof. Finally, in Section 4 we present the numerical results from experiments on large-scale face recognition problems.

2. Problem Statement and Related Methods.

In this section we give a precise description of the problem class MAGMA can be applied to and discuss the main methods related to our algorithm. MAGMA uses gradient and mirror descent steps, but for some iterations it uses a smooth coarse model to compute the search direction. So in order to understand the main components of the proposed algorithm we briefly review gradient descent, mirror descent and smoothing techniques in convex optimization. We also introduce our main assumptions and some notation that will be used later on.

2.1. Convex Composite Optimization.

Let $f : \mathbb{R}^n \to (-\infty, \infty]$ be a convex, continuously differentiable function with an L_f-Lipschitz continuous gradient:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_* \leq L_f \|\mathbf{x} - \mathbf{y}\|,$$

where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$ and $g : \mathbb{R}^n \to (-\infty, \infty]$ be a closed, proper, locally Lipschitz continuous convex function, but not necessarily differentiable. Assuming that the minimizer of the following optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \{F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})\}, \tag{2.1}$$

is attained, we are concerned with solving it.

An important special case of (2.1) is when $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}\|_2^2$ and $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$, where $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a full rank matrix with $m < n$, $\mathbf{b} \in \mathbb{R}^m$ is a vector and $\lambda > 0$ is a parameter. Then (2.1) becomes

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1, \tag{P_1}$$

The problem (P_1) arises from solving underdetermined linear system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$. This system has more unknowns than equations and thus has infinitely many solutions. A

© xxxx Society for Industrial and Applied Mathematics

common way of avoiding that obstacle and narrowing the set of solutions to a well-defined one is regularization via sparsity inducing norms [22]. In general, the problem of finding the sparsest decomposition of a matrix $\mathbf{A}$ with regard to data sample $\mathbf{b}$ can be written as the following non-convex minimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b}, \tag{P_0}$$

where $\|\cdot\|_0$ denotes the ℓ_0-pseudo-norm, i.e. counts the number of non-zero elements of $\mathbf{x}$. Thus the aim is to recover the sparsest $\mathbf{x}$ such that $\mathbf{A}\mathbf{x} = \mathbf{b}$. However, solving the above non-convex problem is NP-hard, even difficult to approximate [2]. Moreover, less sparse, but more stable to noise solutions are often more preferred than the sparsest one. These issues can be addressed by minimizing the more forgiving ℓ_1-norm instead of the ℓ_0-pseudo-norm:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{x}\|_1 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b}. \tag{BP}$$

It can be shown that problem (BP) (often referred to as Basis Pursuit (BP)) is convex, its solutions are gathered in a bounded convex set and at least one of them has at most m non-zeros [22]. In order to handle random noise in real world applications, it is often beneficial to allow some constraint violation. Allowing a certain error $\epsilon > 0$ to appear in the reconstruction we obtain the so-called Basis Pursuit Denoising (BPD) problem:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{x}\|_1 \quad \text{subject to} \quad \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 \leq \epsilon, \tag{BPD}$$

or using the Lagrangian dual we can equivalently rewrite it as an unconstrained, but non-smooth problem (P_1). Note that problem (P_1) is equivalent to the well known LASSO regression [44]. Often BP and BPD (in both constrained and unconstrained forms) are referred to as ℓ_1-min problems.

A relatively new, but very popular example of the BPD problem is the so-called dense error correction (DEC) [47], which studies the problem of recovering a sparse signal $\mathbf{x}$ from highly corrupted linear measurements $\mathbf{A}$. It was shown that if $\mathbf{A}$ has highly correlated columns, with an overwhelming probability the solution of (BP) is also a solution for (P_0) [47]. One example of DEC is the face recognition (FR) problem, where $\mathbf{A}$ is a dictionary of facial images, with each column being one image, $\mathbf{b}$ is a new incoming image and the non-zero elements of the sparse solution $\mathbf{x}$ identify the person in the dictionary whose image $\mathbf{b}$ is.

Throughout this section we will continue referring to the FR problem as a motivating example to explain certain details of the theory. In the DEC setting, $\mathbf{A}$ among other structural assumptions, is also assumed to have highly correlated columns; clearly, facial images are highly correlated. This assumption on one hand means that in some sense $\mathbf{A}$ contains extra information, but on the other hand, it is essential for the correct recovery. We propose to exploit the high correlation of the columns of $\mathbf{A}$ by creating coarse models that have significantly less columns and thus can be solved much faster by a multi-level algorithm.

We use the index H to indicate the coarse level of a multi-level method: $\mathbf{x}_H \in \mathbb{R}^{n_H}$ is the coarse level variable, $f_H(\cdot) : \mathbb{R}^{n_H} \to (-\infty, \infty]$ and $g_H(\cdot) : \mathbb{R}^{n_H} \to (-\infty, \infty]$ are the corresponding coarse surrogates of $f(\cdot)$ and $g(\cdot)$, respectively. In theory, these functions only need to satisfy a few very general assumptions (Assumption 1), but of course, in practice they

© xxxx Society for Industrial and Applied Mathematics

should reflect the structure of the problem, so that the coarse model is a meaningful smaller sized representation of the original problem. In the face recognition example creating coarse models for a dictionary of facial images is fairly straightforward: we take linear combinations of the columns of $\mathbf{A}$.

In our proposed method we do not use the direct prolongation of the coarse level solution to obtain a solution for the fine level problem. Instead, we use these coarse solutions to construct search directions for the fine level iterations. Therefore, in order to ensure the convergence of the algorithm it is essential to ensure that the fine and coarse level problems are coherent in terms of their gradients [33, 46]. However, in our setting the objective function is not smooth. To overcome this in a consistent way, we construct smooth coarse models, as well as use the smoothed version of the fine problem to find appropriate step-sizes[1]. Therefore, we make the following natural assumptions on the coarse model.

Assumption 1. *For each coarse model constructed from a fine level problem (2.1) it holds that*

1. *The coarse level problem is solvable, i.e. bounded from below and the minimum is attained, and*
2. *$f_H(\mathbf{x})$ and $g_H(\mathbf{x})$ are both continuous, closed, convex and differentiable with Lipschitz continuous gradients.*

On the fine level, we say that iteration k is a coarse step, if the search direction is obtained by coarse correction. To denote the j-th iteration of the coarse model, we use $\mathbf{x}_{H,j}$.

2.2. Gradient Descent Methods. Numerous methods have been proposed to solve (2.1) when g has a simple structure. By simple structure we mean that its proximal mapping, defined as

$$\text{prox}_L(\mathbf{x}) = \arg\min_{\mathbf{y}\in\mathbb{R}^n} \left\{ \frac{L}{2}\|\mathbf{y}-\mathbf{x}\|^2 + \langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + g(\mathbf{y}) \right\},$$

for some $L \in \mathbb{R}$, is given in a closed form for any f. Correspondingly, denote

$$\text{Prog}_L(\mathbf{x}) = -\min_{\mathbf{y}\in\mathbb{R}^n} \left\{ \frac{L}{2}\|\mathbf{y}-\mathbf{x}\|^2 + \langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + g(\mathbf{y}) - g(\mathbf{x}) \right\}.$$

We often use the prox and Prog operators with the Lipschitz constant L_f of ∇f. In that case we simplify the notation to $\text{prox}(\mathbf{x}) = \text{prox}_{L_f}(\mathbf{x})$ and $\text{Prog}(\mathbf{x}) = \text{Prog}_{L_f}(\mathbf{x})$ [2].

In case of the FR problem (and LASSO, in general) $\text{prox}(\mathbf{x} - \frac{1}{L}\nabla f(\mathbf{x}))$ becomes the shrinkage-thresholding operator $\mathcal{T}_L(\mathbf{x})_i = (|\mathbf{x}_i| - 1/L)_+ \text{sgn}(\mathbf{x}_i)$ and the iterative scheme is given by,

$$\mathbf{x}_{k+1} = \mathcal{T}_{\lambda/L}\left(\mathbf{x}_k - \frac{2}{L}\mathbf{A}^\top(\mathbf{A}\mathbf{x}_k - \mathbf{b})\right).$$

Then problem (P_1) can be solved by the well-known Iterative Shrinkage Thresholding Algorithm (ISTA) and its generalizations, see, e.g. [5], [19], [20], [25]. When the stepsize is fixed to

[1] We give more details about smoothing non-smooth functions in general, and $\|\cdot\|_1$ in particular, at the end of this section.

[2] These definitions without loss of generality also cover the case when the problem is constrained and smooth. Also see [35] for the original definitions for the unconstrained smooth case.

© xxxx Society for Industrial and Applied Mathematics

the Lipschitz constant of the problem, ISTA reduces to the following simple iterative scheme,

$$\mathbf{x}_{k+1} = \text{prox}(\mathbf{x}_k).$$

If $g(\cdot) \equiv 0$, the problem is smooth, $\text{prox}(\mathbf{x}) \equiv \mathbf{x}_k - \frac{1}{L_f}\nabla f(\mathbf{x}_k)$ and ISTA becomes the well-known steepest descent algorithm with

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \frac{1}{L_f}\nabla f(\mathbf{x}_k).$$

For composite optimization problems it is common to use the gradient mapping as an optimality measure [38]:

$$\text{D}(\mathbf{x}) = \mathbf{x} - \text{prox}(\mathbf{x}).$$

It is a generalization of the gradient for the composite setting and preserves its most important properties used for the convergence analysis.

We will make use of the following fundamental Lemma in our convergence analysis. The proof can be found, for instance, in [38].

Lemma 2.1 (Gradient Descent Guarantee). *For any* $\mathbf{x} \in \mathbb{R}^n$,

$$F(\text{prox}(\mathbf{x})) \leq F(\mathbf{x}) - \text{Prog}(\mathbf{x}).$$

Using Lemma 2.1, it can be shown that ISTA computes an ϵ-optimal solution in terms of function values in $\mathcal{O}(1/\epsilon)$ iterations (see [5] and references therein).

2.3. Mirror Descent Methods. Mirror Descent methods solve the general problem of minimizing a proper convex and locally Lipschitz continuous function F over a simple convex set Q by constructing lower bounds to the objective function (see for instance [34], [39]). Central to the definition of mirror descent is the concept of a distance generating function.

Definition 2.2. *A function* $\omega(\mathbf{x}) : Q \to \mathbb{R}$ *is called a Distance Generating Function (DFG), if* ω *is* 1*-strongly convex with respect to a norm* $\|\cdot\|$*, that is*

$$\omega(\mathbf{z}) \geq \omega(\mathbf{x}) + \langle \nabla\omega(\mathbf{x}), \mathbf{z} - \mathbf{x}\rangle + \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|^2 \quad \forall \mathbf{x}, \mathbf{z} \in Q.$$

Accordingly, the Bregman divergence (or prox-term) is given as

$$V_{\mathbf{x}}(\mathbf{z}) = \omega(\mathbf{z}) - \langle \nabla\omega(\mathbf{x}), \mathbf{z} - \mathbf{x}\rangle - \omega(\mathbf{x}) \quad \forall \mathbf{x}, \mathbf{z} \in Q.$$

The property of DFG ensures that $V_{\mathbf{x}}(\mathbf{x}) = 0$ *and* $V_{\mathbf{x}}(\mathbf{z}) \geq \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|^2 \geq 0$.

The main step of mirror descent algorithm is given by,

$$\mathbf{x}_{k+1} = \text{Mirr}_{\mathbf{x}_k}(\nabla f(\mathbf{x}_k), \alpha), \quad \text{where} \quad \text{Mirr}_{\mathbf{x}}(\xi, \alpha) = \arg\min_{\mathbf{z}\in\mathbb{R}^n}\{V_{\mathbf{x}}(\mathbf{z}) + \langle \alpha\xi, \mathbf{z} - \mathbf{x}\rangle + \alpha g(\mathbf{z})\}.$$

Here again, we assume that Mirr can be evaluated in closed form. Note that choosing $\omega(\mathbf{z}) = 1/2\|\mathbf{z}\|_2^2$ as the DFG, which is strongly convex w.r.t. to the ℓ_2-norm over any convex set and correspondingly $V_{\mathbf{x}}(\mathbf{z}) = \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|_2^2$, the mirror step becomes exactly the proximal step of

© xxxx Society for Industrial and Applied Mathematics

ISTA. However, regardless of this similarity, the gradient and mirror descent algorithms are conceptually different and use different techniques to prove convergence. The core lemma for mirror descent convergence is the so called Mirror Descent Guarantee. For the unconstrained[3] composite setting it is given below, a proof can be found in [38].

Lemma 2.3 (Mirror Descent Guarantee). *Let* $\mathbf{x}_{k+1} = \mathrm{Mirr}_{\mathbf{x}_k}(\nabla f(\mathbf{x}_k), \alpha)$, *then* $\forall \mathbf{u} \in \mathbb{R}^n$

$$\begin{aligned}\alpha(F(\mathbf{x}_k) - F(\mathbf{u})) &\leq \alpha\langle \nabla f(\mathbf{x}_k), \mathbf{x}_k - \mathbf{u}\rangle + \alpha(g(\mathbf{x}_k) - g(\mathbf{u})) \\ &\leq \alpha^2 L_f \,\mathrm{Prog}(\mathbf{x}_k) + V_{\mathbf{x}_k}(\mathbf{u}) - V_{\mathbf{x}_{k+1}}(\mathbf{u}).\end{aligned}$$

Using the Mirror Descent Guarantee, it can be shown that the Mirror Descent algorithm converges to the minimizer $\mathbf{x}^*$ in $\mathcal{O}(L_F/\epsilon)$, where ϵ is the convergence precision [9].

2.4. Accelerated First Order Methods. A faster method for minimizing smooth convex functions with asymptotically tight $\mathcal{O}(1/\sqrt{\epsilon})$ convergence rate [34] was proposed by Nesterov in [37]. This method and its variants were later extended to solve the more general problem (2.1) (e.g. [5], [36], [1]), see [45] for full review and analysis of accelerated first order methods. The main idea behind the accelerated methods is to update the iterate based on a linear combination of previous iterates rather than only using the current one as in gradient or mirror descent algorithms. The first and most popular method that achieved an optimal convergence rate for composite problems is FISTA [5]. At each iteration it performs one proximal operation, then uses a linear combination of its result and the previous iterator for the next iteration.

Data: FISTA($f(\cdot)$, $g(\cdot)$, $\mathbf{x}_0$)
Choose $\epsilon > 0$.
Set $\mathbf{y}_1 = \mathbf{x}_0$ and $t_1 = 1$.
for $k = 0, 1, 2, \ldots$ **do**
  Set $\mathbf{x}_{k+1} = \mathrm{prox}(\mathbf{x}_k)$.
  Set $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.
  Set $\mathbf{y}_{k+1} = \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}_k - \mathbf{x}_{k-1})$.
  if $\|D(\mathbf{x}_k)\|_* < \epsilon$ **then**
    return $\mathbf{x}_k$.
  end
end

Algorithm 1: FISTA

Alternatively, Nesterov's accelerated scheme can be modified to use two proximal operations at each iteration and linearly combine their results as the current iteration point [35]. In a recent work this approach was reinterpreted as a combination of gradient descent and

[3] Simple constraints can be included by defining g as an indicator function.

© xxxx Society for Industrial and Applied Mathematics

mirror descent algorithms [1]. The algorithm is given next.

Data: AGM($f(\cdot)$, $g(\cdot)$, $\mathbf{x}_0$)
Choose $\epsilon > 0$.
Choose DGF ω and set $V_{\mathbf{x}}(\mathbf{y}) = \omega(\mathbf{y}) - \langle \nabla\omega(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle - \omega(\mathbf{x})$.
Set $\mathbf{y}_0 = \mathbf{z}_0 = \mathbf{x}_0$.
for $k = 0, 1, 2, \ldots$ **do**
  Set $\alpha_{k+1} = \frac{k+2}{2L}$ and $t_k = \frac{2}{k+2}$.
  Set $\mathbf{x}_k = t_k\mathbf{z}_k + (1 - t_k)\mathbf{y}_k$.
  if $\|D(\mathbf{x}_k)\|_* < \epsilon$ **then**
    return $\mathbf{x}_k$.
  end
  Set $\mathbf{y}_{k+1} = \mathrm{prox}(\mathbf{x}_k)$.
  Set $\mathbf{z}_{k+1} = \mathrm{Mirr}_{\mathbf{z}_k}(\nabla f(\mathbf{x}_k), \alpha_{k+1})$.
end

Algorithm 2: AGM

The convergence of Algorithm 2 relies on the following lemmas.

Lemma 2.4. *If* $t_k = \frac{1}{\alpha_{k+1}L_f}$ *then it satisfies that for every* $\mathbf{u} \in \mathbb{R}^n$,

$$\begin{aligned}
&\alpha_{k+1}\langle \nabla f(\mathbf{x}_k), \mathbf{z}_k - \mathbf{u}\rangle + \alpha_{k+1}(g(\mathbf{z}_k) - g(\mathbf{u})) \\
&\le \alpha_{k+1}^2 L_f \,\mathrm{Prog}(\mathbf{x}_k) + V_{\mathbf{z}_k}(\mathbf{u}) - V_{\mathbf{z}_{k+1}}(\mathbf{u}) \\
&\le \alpha_{k+1}^2 L_f (F(\mathbf{x}_k) - F(\mathbf{y}_{k+1})) + V_{\mathbf{z}_k}(\mathbf{u}) - V_{\mathbf{z}_{k+1}}(\mathbf{u}).
\end{aligned}$$

Lemma 2.5 (Coupling). *For any* $\mathbf{u} \in \mathbb{R}^n$

$$\alpha_{k+1}^2 L_f F(\mathbf{y}_{k+1}) - (\alpha_{k+1}^2 L_f - \alpha_{k+1})F(\mathbf{y}_k) + (V_{\mathbf{z}_{k+1}}(\mathbf{u}) - V_{\mathbf{z}_k}(\mathbf{u})) \le \alpha_{k+1}F(\mathbf{u}).$$

Theorem 2.6 (Convergence). *Algorithm 2 ensures that*

$$F(\mathbf{y}_T) - F(\mathbf{x}^*) \le \frac{4\Theta L_f}{T^2},$$

where Θ *is any upper bound on* $V_{\mathbf{x}_0}(\mathbf{x}^*)$.

Remark 1. *In [1] the analysis of the algorithm is done for smooth constrained problems, however it can be set to work for the unconstrained composite setting as well.*

Remark 2. *Whenever* L_f *is not known or is expensive to calculate, we can use backtracking line search with the following stopping condition*

$$F(\mathbf{y}_{k+1}) \le F(\mathbf{x}_k) - \mathrm{Prog}_L(\mathbf{x}_k), \tag{2.2}$$

where $L > 0$ *is the smallest number for which the condition (2.2) holds. The convergence analysis of the algorithm can be extended to cover this case in a straightforward way (see [5]).*

© xxxx Society for Industrial and Applied Mathematics

2.5. Smoothing and First Order Methods. A more general approach of minimizing non-smooth problems is to replace the original problem by a sequence of smooth problems. The smooth problems can be solved more efficiently than by using subgradient type methods on the original problem. In [35] Nesterov proposed a first order smoothing method, where the objective function is of special max-type. Then Beck and Teboulle extended this method to a more general framework for the class of so-called *smoothable* functions [6]. Both methods are proven to converge to an ϵ-optimal solution in $\mathcal{O}(1/\epsilon)$ iterations.

Definition 2.7. *Let $g : \mathbb{E} \to (-\infty, \infty]$ be a closed and proper convex function and let $\mathbf{x} \subseteq \operatorname{dom} g$ be a closed convex set. The function g is called "(α, β, K)-smoothable" over X, if there exist β_1, β_2 satisfying $\beta_1 + \beta_2 = \beta > 0$ such that for every $\mu > 0$ there exists a continuously differentiable convex function $g_\mu : \mathbb{E} \to (-\infty, \infty)$ such that the following hold:*

1. *$g(\mathbf{x}) - \beta_1\mu \leq g_\mu(\mathbf{x}) \leq g(\mathbf{x}) + \beta_2\mu$ for every $\mathbf{x} \in X$.*
2. *The function g_μ has a Lipschitz gradient over X with a Lipschitz constant such that there exists $K > 0$, $\alpha > 0$ such that*

$$\|\nabla g_\mu(\mathbf{x}) - \nabla g_\mu(\mathbf{y})\|_* \leq \left(K + \frac{\alpha}{\mu}\right) \|\mathbf{x} - \mathbf{y}\| \text{ for every } \mathbf{x}, \mathbf{y} \in X.$$

We often use $L_\mu = K + \frac{\alpha}{\mu}$.

It was shown in [6] that with an appropriate choice of the smoothing parameter μ a solution obtained by smoothing the original non-smooth problem and solving it by a method with $\mathcal{O}(1/\sqrt{\epsilon})$ convergence rate finds an ϵ-optimal solution in $\mathcal{O}(1/\epsilon)$ iterations. When the proximal step computation is tractable, accelerated first order methods are superior both in theory and in practice. However, for the purposes of establishing a step-size strategy for MAGMA it is convenient to work with smooth problems. We combine the theoretical superiority of non-smooth methods with the rich mathematical structure of smooth models to derive a step size strategy that guarantees convergence. MAGMA does not use smoothing in order to solve the original problem. Instead, it uses a smoothed version of the original model to compute a step size when the search direction is given by the coarse model (coarse correction step).

For the FR problem (and (P_1) in general), where $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$, it can be shown [6] that

$$g_\mu(\mathbf{x}) = \lambda \sum_{j=1}^{n} \sqrt{\mu^2 + \mathbf{x}_j^2} \tag{2.3}$$

is a μ-smooth approximation of $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ with parameters $(\lambda, \lambda n, 0)$.

3. Multi-level Accelerated Proximal Algorithm. In this section we formally present our proposed algorithm within the multi-level optimization setting, together with the proof of its convergence with $\mathcal{O}(1/\sqrt{\epsilon})$ rate, where ϵ is the convergence precision.

3.1. Model Construction. First we present the notation and construct a mathematical model, which will later be used in our algorithm for solving (2.1). A well-defined and converging multi-level algorithm requires appropriate information transfer mechanisms in between levels and a well-defined and coherent coarse model. These aspects of our model are presented next.

© xxxx Society for Industrial and Applied Mathematics

3.1.1. Information Transfer. In order to transfer information between levels, we use linear restriction and prolongation operators $\mathbf{R} : \mathbb{R}^{n \times n_H}$ and $\mathbf{P} : \mathbb{R}^{n_H \times n}$ respectively, where $n_H < n$ is the size of the coarse level variable. The restriction operator $\mathbf{R}$ transfers the current fine level point to the coarse level and the prolongation operator $\mathbf{P}$ constructs a search direction for the fine level from the coarse level solution. The techniques we use are standard (see [13]) so we keep this section brief.

There are many ways to design $\mathbf{R}$ and $\mathbf{P}$ operators, but they should satisfy the following condition to guarantee the convergence of the algorithm,

$$\sigma \mathbf{P} = \mathbf{R}^\top,$$

with some scalar $\sigma > 0$. Without loss of generality we assume $\sigma = 1$, which is a standard assumption in the multi-level literature [13], [33].

In our applications, when the problem is given as (P_1), we use a simple interpolation operator given in the form of (4.3) as the restriction operator, which essentially constructs linear combinations of the columns of dictionary $\mathbf{A}$. We give more details on using $\mathbf{R}$ for our particular problem in Section 4.

3.1.2. Coarse Model. A key property of the coarse model is that at its initial point $\mathbf{x}_{H,0}$ the optimality conditions of the two models match. This is achieved by adding a linear term to the coarse objective function:

$$F_H(\mathbf{x}_H) = f_H(\mathbf{x}_H) + g_H(\mathbf{x}_H) + \langle \mathbf{v}_H, \mathbf{x}_H \rangle, \tag{3.1}$$

where the vector $\mathbf{v}_H \in \mathbb{R}^{n_H}$ is defined so that the fine and coarse level problems are first order coherent, that is, their first order optimality conditions coincide. In this paper we have a non-smooth objective function and assume the coarse model is smooth [4] with L_H-Lipschitz continuous gradient. Furthermore, we construct $\mathbf{v}_H$ such that the gradient of the coarse model is equal to the gradient of the smoothed fine model's gradient:

$$\nabla F_H(\mathbf{R}\mathbf{x}) = \mathbf{R} \nabla F_\mu(\mathbf{x}), \tag{3.2}$$

where $F_\mu(\mathbf{x})$ is a μ-smooth approximation of $F(\mathbf{x})$ with parameters (a, β, K). Note that for the composite problem (2.1) $F_\mu(\mathbf{x}) = f(\mathbf{x}) + g_\mu(\mathbf{x})$, where g_μ is a μ-smooth approximation of $g(\mathbf{x})$. In our experiments for (P_1) we use $g_\mu(\mathbf{x})$ as given in (2.3). The next lemma gives a choice for $\mathbf{v}_H$, such that (3.2) is satisfied.

Lemma 3.1 (Lemma 3.1 of [46]). *Let F_H be a Lipschitz continuous function with L_H Lipschitz constant, then for*

$$\mathbf{v}_H = \mathbf{R} \nabla F_\mu(\mathbf{x}) - (\nabla f_H(\mathbf{R}\mathbf{x}) + \nabla g_H(\mathbf{R}\mathbf{x})) \tag{3.3}$$

we have

$$\nabla F_H(\mathbf{R}\mathbf{x}) = \mathbf{R} \nabla F_\mu(\mathbf{x}).$$

The condition (3.2) is referred to as *first order coherence*. It ensures that if $\mathbf{x}$ is optimal for the smoothed fine level problem, then $\mathbf{R}\mathbf{x}$ is also optimal in the coarse level.

[4] This can be done by smoothing the non-smooth coarse term.

© xxxx Society for Industrial and Applied Mathematics

While in practice it is often beneficial to use more than one levels, for the theoretical analysis of our algorithm without loss of generality we assume that there is only one coarse level. Indeed, assume that $\mathbf{I}_h^H \in \mathbb{R}^{H\times h}$ is a linear operator that transfers $\mathbf{x}$ from $\mathbf{R}^h$ to $\mathbf{R}^H$. Now, we can define our restriction operator as $\mathbf{R} = \prod_{i=1}^{l} \mathbf{I}_{h_i}^{H_i}$, where $h_1 = n$, $h_{i+1} = H_i$ and $H_l = n_H$. Accordingly, the prolongation operator will be $\mathbf{P} = \prod_{i=l}^{1} \mathbf{I}_{H_i}^{h_i}$, where $\sigma_i \mathbf{I}_{H_i}^{h_i} = (\mathbf{I}_{h_i}^{H_i})^\top$. Note that this construction satisfies the assumption that $\sigma\mathbf{P} = \mathbf{R}^\top$, with $\sigma = \prod_{i=1}^{l} \sigma_i$.

3.2. MAGMA. In this subsection we describe our proposed multi-level accelerated proximal algorithm for solving (2.1). We call it MAGMA for Multi-level Accelerated Gradient Mirror descent Algorithm. As in [35] and [1], at each iteration MAGMA performs both gradient and mirror descent steps, then uses a convex combination of their results to compute the next iterate. The crucial difference here is that whenever a coarse iteration is performed, our algorithm uses the coarse direction

$$\mathbf{y}_{k+1} = \mathbf{x}_k + s_k \mathbf{d}_k(\mathbf{x}_k),$$

instead of the gradient, where $\mathbf{d}_k(\cdot)$ is the search direction and s_k is an appropriately chosen step-size. Next we describe how each of these terms is constructed.

At each iteration k the algorithm first decides whether to use the gradient or the coarse model to construct a search direction for the gradient descent step. This decision is based on the optimality condition at the current iterate $\mathbf{x}_k$: we do not want to use the coarse direction when it is not helpful, i.e. when

- the first order optimality conditions are almost satisfied, or
- the current point $\mathbf{x}_k$ is very close to the point $\tilde{\mathbf{x}}$, where a coarse iteration was last performed, as long as the algorithm has not performed too many gradient correction steps.

More formally, we choose the coarse direction, whenever both of the following conditions are satisfied:

$$\begin{aligned} \|\mathbf{R}\nabla F_{\mu_k}(\mathbf{x}_k)\|_* &> \kappa \|\nabla F_{\mu_k}(\mathbf{x}_k)\|_*, \\ \|\mathbf{x}_k - \tilde{\mathbf{x}}\| &> \vartheta \|\tilde{\mathbf{x}}\| \quad \text{or} \quad q < K_d, \end{aligned} \tag{3.4}$$

where $\kappa \in (0, \min(1, \min\|\mathbf{R}\|))$, $\vartheta > 0$ and K_d are predefined constants, and q is the number of consecutive gradient correction steps [46], [41].

If a coarse direction is chosen, a coarse model is constructed as described in (3.1). In order to satisfy the coherence property (3.2), we start the coarse iterations with $\mathbf{x}_{H,0} = \mathbf{R}\mathbf{x}_k$, then solve the coarse model by a first order method and construct a search direction:

$$\mathbf{d}_k(\mathbf{x}_k) = \mathbf{P}(\mathbf{x}_{H,N_H} - \mathbf{R}\mathbf{x}_k), \tag{3.5}$$

where $\mathbf{x}_{H,N_H} \in \mathbb{R}^{n_H}$ is an approximate solution of the coarse model after N_H iterations. Note that in practice we do not find the exact solution, but rather run the algorithm for N_H iterations, until we achieve a certain acceptable $\mathbf{x}_{H,N_H}$. In our experiments we used the monotone version of FISTA [4], however in theory any monotone algorithm will ensure convergence.

We next show that any coarse direction defined in (3.5) is a descent direction for the smoothed fine problem.

© xxxx Society for Industrial and Applied Mathematics

Lemma 3.2 (Descent Direction). *If at iteration k a coarse step is performed and suppose that $F_H(\mathbf{x}_{H,N_H}) < F_H(\mathbf{x}_{H,1})$, then for any $\mathbf{x}_k$ it holds that*

$$\langle \mathbf{d}_k(\mathbf{x}_k), \nabla F_{\mu_k}(\mathbf{x}_k)\rangle < -\frac{\kappa^2}{2L_H}\|\nabla F_{\mu_k}(\mathbf{x}_k)\|_*^2 \leq 0.$$

Proof. Using the definition of coarse direction $\mathbf{d}(\mathbf{x}_k)$, linearity of $\mathbf{R}$ and Lemma 3.1 we obtain

$$\begin{aligned}\langle \mathbf{d}_k(\mathbf{x}_k), \nabla F_{\mu_k}(\mathbf{x}_k)\rangle &= \langle \mathbf{R}^\top(\mathbf{x}_{H,N_H} - \mathbf{x}_{H,0}), \nabla F_{\mu_k}(\mathbf{x}_k)\rangle \\ &= \langle \mathbf{x}_{H,N_H} - \mathbf{x}_{H,0}, \mathbf{R}\nabla F_{\mu_k}(\mathbf{x}_k)\rangle \\ &= \langle \mathbf{x}_{H,N_H} - \mathbf{x}_{H,0}, \nabla F_H(\mathbf{x}_{H,0})\rangle \\ &\leq F_H(\mathbf{x}_{H,N_H}) - F_H(\mathbf{x}_{H,0}),\end{aligned} \tag{3.6}$$

where for the last inequality we used the convexity of F_H. On the other hand using the monotonicity assumption on the coarse level algorithm and lemma (2.1) we derive:

$$F_H(\mathbf{x}_{H,N_H}) - F_H(\mathbf{x}_{H,0}) < F_H(\mathbf{x}_{H,1}) - F_H(\mathbf{x}_{H,0}) \leq -\frac{1}{2L_H}\|\nabla F_H(\mathbf{x}_{H,0})\|_*^2. \tag{3.7}$$

Now using the choice $\mathbf{x}_{H,0} = \mathbf{R}\mathbf{x}_k$ and (3.2) we obtain:

$$\nabla F_H(\mathbf{x}_{H,0}) = \nabla F_H(\mathbf{R}\mathbf{x}_k) = \mathbf{R}\nabla F_{\mu_k}(\mathbf{x}_k). \tag{3.8}$$

Then combining (3.7), (3.8) and condition (3.4) we obtain

$$F_H(\mathbf{x}_{H,N_H}) - F_H(\mathbf{x}_{H,0}) < -\frac{1}{2L_H}\|\nabla \mathbf{R} F_{\mu_k}(\mathbf{x}_k)\|^2 \leq -\frac{\kappa^2}{2L_H}\|\nabla F_{\mu_k}(\mathbf{x}_k)\|^2. \tag{3.9}$$

Finally, combining (3.6) and (3.9) we derive the desired result. ∎

After constructing a descent search direction, we find suitable step sizes for both gradient and mirror steps at each iteration. In [1] a fixed step size of $\alpha_{k+1} = \frac{1}{L_f}$ is used for gradient steps. However, in our algorithm we do not always use the steepest descent direction for gradient steps, and therefore another step size strategy is required. We show that step sizes obtained from Armijo-type backtracking line search on the smoothed (fine) problem ensures that the algorithm converges with optimal rate. Starting from a predefined $s_k = s^0$, we reduce it by a factor of constant $\tau \in (0,1)$ until

$$F_{\mu_k}(\mathbf{y}_{k+1}) \leq F_{\mu_k}(\mathbf{x}_k) + cs_k\langle \mathbf{d}_k(\mathbf{x}_k), \nabla F_{\mu_k}(\mathbf{x}_k)\rangle \tag{3.10}$$

is satisfied for some predefined constant $c \in (0,1)$.

For mirror steps we always use the current gradient as search direction, as they have significant contribution to the convergence only when the gradient is small enough, meaning we are close to the minimizer, and in those cases we do not perform coarse iterations. However,

© xxxx Society for Industrial and Applied Mathematics

we need to adjust the step size α_k for mirror steps in order to ensure convergence. Next we formally present the proposed algorithm.

Data: MAGMA($f(\cdot)$, $g(\cdot)$, $\mathbf{x}_0$, T)
Set $\mathbf{y}_0 = \mathbf{z}_0 = \mathbf{x}_0$ and $\alpha_0 = 0$
for $k = 0, 1, 2, ..., T$ **do**
  Choose η_{k+1} and α_{k+1} according to (3.18) and (3.19).
  Set $t_k = \frac{1}{\alpha_{k+1}\eta_{k+1}}$.
  Set $\mathbf{x}_k = t_k\mathbf{z}_k + (1 - t_k)\mathbf{y}_k$.
  if *Conditions (3.4) are satisfied* **then**
    Perform N_H coarse iterations.
    Set $\mathbf{d}_k = \mathbf{P}(\mathbf{x}_{H,N_H} - \mathbf{R}\mathbf{x}_k)$.
    Set $\mathbf{y}_{k+1} = \mathbf{x}_k + s_k\mathbf{d}(\mathbf{x}_k)$, with s_k satisfying (3.10).
  end
  else
    Set $\mathbf{y}_{k+1} = \text{prox}(\mathbf{x}_k)$.
  end
  Set $\mathbf{z}_{k+1} = \text{Mirr}_{\mathbf{z}_k}(\nabla f(\mathbf{x}_k), \alpha_{k+1})$.
end

Algorithm 3: MAGMA

Now we show the convergence of Algorithm 3 with $\mathcal{O}(1/\sqrt{\epsilon})$ rate. First we prove an analogue of Lemma 2.4.

Lemma 3.3. *For every* $\mathbf{u} \in \mathbb{R}^n$ *and* $\eta_k > 0$ *and*

$$\eta_{k+1} = \begin{cases} L_f, & \text{when } k+1 \text{ is a gradient correction step} \\ \max\left\{\frac{1}{4\alpha_k^2\eta_k}, \frac{L_H}{cs_k\kappa^2}\right\}, & \text{otherwise} \end{cases}$$

it holds that

$$\begin{aligned} &\alpha_{k+1}\langle\nabla f(\mathbf{x}_k), \mathbf{z}_k - \mathbf{u}\rangle + \alpha_{k+1}(g(\mathbf{z}_k) - g(\mathbf{u})) \\ \leq &\alpha_{k+1}^2 L_f \,\text{Prog}(\mathbf{x}_k) + V_{\mathbf{z}_k}(\mathbf{u}) - V_{\mathbf{z}_{k+1}}(\mathbf{u}) \\ \leq &\alpha_{k+1}^2[\eta_{k+1}(F_{\mu_k}(\mathbf{x}_k) - F_{\mu_k}(\mathbf{y}_{k+1})) + L_f\beta\mu_k] + V_{\mathbf{z}_k}(\mathbf{u}) - V_{\mathbf{z}_{k+1}}(\mathbf{u}) \end{aligned}$$

Proof. First note that the proof of the first inequality follows directly from Lemma 4.2 of [1]. Now assume k is a coarse step, then the first inequality follows from Lemma 2.3. To show

© xxxx Society for Industrial and Applied Mathematics

the second one we first note that for $\tilde{\mathbf{x}} = \text{prox}(\mathbf{x}_k)$,

$$\begin{aligned}
&\text{Prog}(\mathbf{x}_k) - \frac{1}{2L_f}\|\nabla F_{\mu_k}(\mathbf{x}_k)\|^2 \\
=& -\frac{L_f}{2}\|\tilde{\mathbf{x}} - \mathbf{x}_k\|^2 - \langle \nabla f(\mathbf{x}_k), \tilde{\mathbf{x}} - \mathbf{x}_k\rangle - g(\tilde{\mathbf{x}}) + g(\mathbf{x}_k) - \frac{1}{2L_f}\|\nabla F_{\mu_k}(\mathbf{x}_k)\|^2 \\
\leq& -\frac{L_f}{2}\|\tilde{\mathbf{x}} - \mathbf{x}_k\|^2 - \langle \nabla f(\mathbf{x}_k), \tilde{\mathbf{x}} - \mathbf{x}_k\rangle - \langle \tilde{\mathbf{x}} - \mathbf{x}_k, \nabla g_{\mu_k}(\mathbf{x}_k)\rangle + \beta\mu_k - \frac{1}{2L_f}\|\nabla F_{\mu_k}(\mathbf{x}_k)\|^2 \\
=& -\frac{L_f}{2}(\|\tilde{\mathbf{x}} - \mathbf{x}_k\|^2 + 2\langle \tilde{\mathbf{x}} - \mathbf{x}_k, \frac{1}{L_f}\nabla F_{\mu_k}(\mathbf{x}_k)\rangle + \|\frac{1}{L_f}\nabla F_{\mu_k}(\mathbf{x}_k)\|^2) + \beta\mu_k \\
\leq& \beta\mu_k.
\end{aligned}$$

Here we used the definitions of Prog, prox and g_μ, as well as the convexity of g_μ. Therefore from Lemma 3.2 and backtracking condition (3.10) we obtain that if k is a coarse correction step, then

$$\begin{aligned}
L_f \,\text{Prog}(\mathbf{x}_k) &\leq \frac{1}{2}\|\nabla F_{\mu_k}(\mathbf{x}_k)\|^2 + L_f\beta\mu_k \\
&\leq -\frac{L_H}{\kappa^2}\langle \mathbf{d}_k(\mathbf{x}_k), \nabla F_{\mu_k}(\mathbf{x}_k)\rangle + L_f\beta\mu_k \\
&\leq \frac{L_H}{cs_k\kappa^2}(F_{\mu_k}(\mathbf{x}_k) - F_{\mu_k}(\mathbf{y}_{k+1})) + L_f\beta\mu_k.,
\end{aligned}$$

Otherwise, if k is a gradient correction step, then from Lemma 2.1

$$L_f \,\text{Prog}(\mathbf{x}_k) \leq L_f(F(\mathbf{x}_k) - F(\mathbf{y}_{k+1})) \leq L_f(F_{\mu_k}(\mathbf{x}_k) - F_{\mu_k}(\mathbf{y}_{k+1})) + L_f\beta\mu_k.$$

Now choosing

$$\eta_{k+1} = \begin{cases} L_f, & \text{when } k+1 \text{ is a gradient correction step} \\ \max\left\{\frac{1}{4\alpha_k^2\eta_k}, \frac{L_H}{cs_k\kappa^2}\right\}, & \text{otherwise} \end{cases}$$

we obtain the desired result. ■

Remark 3. *The recurrent choice of η_{k+1} may seem strange at this point, as we could simply set it to $\max\{\frac{L_H}{cs_k\kappa^2}, L_f\}$, however forcing $\eta_{k+1} \geq \frac{1}{4\alpha_k^2\eta_k}$ helps us ensure that $t_k \in (0, 1]$ later.*

Lemma 3.4 (Coupling). *For any $\mathbf{u} \in \mathbb{R}^n$ and $t_k = 1/\alpha_{k+1}\eta_{k+1}$, where η_{k+1} is defined as in Lemma 3.3, it holds that*

$$\begin{aligned}
\alpha_{k+1}^2\eta_{k+1}F(\mathbf{y}_{k+1}) - (\alpha_{k+1}^2\eta_{k+1} - \alpha_{k+1})F(\mathbf{y}_k) + (V_{\mathbf{z}_{k+1}}(\mathbf{u}) - V_{\mathbf{z}_k}(\mathbf{u})) \\
\leq \alpha_{k+1}F(\mathbf{u}) + (L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k. \quad (3.11)
\end{aligned}$$

Proof. First note that, if k is a gradient correction step, then the proof follows from Lemma 2.5. Now assume k is a coarse correction step, then using the convexity of f we obtain

$$\begin{aligned}
F(\mathbf{x}_k) - F(\mathbf{u}) &= f(\mathbf{x}_k) - f(\mathbf{u}) + g(\mathbf{x}_k) - g(\mathbf{u}) \\
&\leq \langle \nabla f(\mathbf{x}_k), \mathbf{x}_k - \mathbf{u}\rangle + g(\mathbf{x}_k) - g(\mathbf{u}).
\end{aligned} \quad (3.12)$$

© xxxx Society for Industrial and Applied Mathematics

On the other hand, from the definition of g_{μ_k} and its convexity we have

$$\begin{aligned} g(\mathbf{x}_k) - g(\mathbf{z}_k) &\leq g_{\mu_k}(\mathbf{x}_k) + \beta_1\mu_k - g_{\mu_k}(\mathbf{z}_k) + \beta_2\mu_k \\ &\leq \langle \nabla g_{\mu_k}(\mathbf{x}_k), \mathbf{x}_k - \mathbf{z}_k\rangle + \beta\mu_k. \end{aligned} \tag{3.13}$$

Then using (3.13), we can rewrite $g(\mathbf{x}_k) - g(\mathbf{u})$ as

$$\begin{aligned} g(\mathbf{x}_k) - g(\mathbf{u}) &= (g(\mathbf{x}_k) - g(\mathbf{z}_k)) + (g(\mathbf{z}_k) - g(\mathbf{u})) \\ &\leq \langle \nabla g_{\mu_k}(\mathbf{x}_k), \mathbf{x}_k - \mathbf{z}_k\rangle + \beta\mu_k + (g(\mathbf{z}_k) - g(\mathbf{u})). \end{aligned} \tag{3.14}$$

Now rewriting $\langle \nabla f(\mathbf{x}_k), \mathbf{x}_k - \mathbf{u}\rangle = \langle \nabla f(\mathbf{x}_k), \mathbf{x}_k - \mathbf{z}_k\rangle + \langle \nabla f(\mathbf{x}_k), \mathbf{z}_k - \mathbf{u}\rangle$ in (3.12) and using (3.14) we obtain

$$F(\mathbf{x}_k) - F(\mathbf{u}) \leq \langle \nabla F_{\mu_k}(\mathbf{x}_k), \mathbf{x}_k - \mathbf{z}_k\rangle + \beta\mu_k + \langle \nabla f(\mathbf{x}_k), \mathbf{z}_k - \mathbf{u}\rangle + (g(\mathbf{z}_k) - g(\mathbf{u})), \tag{3.15}$$

where we used that $\nabla F_{\mu_k}(\mathbf{x}_k) = \nabla f(\mathbf{x}_k) + \nabla g_{\mu_k}(\mathbf{x}_k)$.

From the choice $t_k(\mathbf{x}_k - \mathbf{z}_k) = (1 - t_k)(\mathbf{y}_k - \mathbf{x}_k)$ and convexity of F_{μ_k} we have

$$\begin{aligned} \langle \nabla F_{\mu_k}(\mathbf{x}_k), \mathbf{x}_k - \mathbf{z}_k\rangle &= \frac{(1-t_k)}{t_k}\langle \nabla F_{\mu_k}(\mathbf{x}_k), \mathbf{y}_k - \mathbf{x}_k\rangle \\ &\leq \frac{(1-t_k)}{t_k}(F_{\mu_k}(\mathbf{y}_k) - F_{\mu_k}(\mathbf{x}_k)) \\ &\leq \frac{(1-t_k)}{t_k}(F(\mathbf{y}_k) - F(\mathbf{x}_k)) + \frac{(1-t_k)}{t_k}\beta\mu_k, \end{aligned}$$

where for the last inequality we used that F_{μ_k} is a μ_k-smooth approximation of F. Then choosing $t_k = 1/(\alpha_{k+1}\eta_{k+1})$, where α_{k+1} is a step-size for the Mirror Descent step and η_{k+1} is defined in Lemma 3.3, we have

$$\langle \nabla F_{\mu_k}(\mathbf{x}_k), \mathbf{x}_k - \mathbf{z}_k\rangle \leq (\alpha_{k+1}\eta_{k+1} - 1)(F(\mathbf{y}_k) - F(\mathbf{x}_k)) + (\alpha_{k+1}\eta_{k+1} - 1)\beta\mu_k. \tag{3.16}$$

Plugging (3.16) into (3.15) we obtain

$$\begin{aligned} \alpha_{k+1}(F(\mathbf{x}_k) - F(\mathbf{u})) \leq (\alpha_{k+1}^2\eta_{k+1} - \alpha_{k+1})(F(\mathbf{y}_k) - F(\mathbf{x}_k)) + \alpha_{k+1}^2\eta_{k+1}\beta\mu_k \\ + \alpha_{k+1}\langle \nabla f(\mathbf{x}_k), \mathbf{z}_k - \mathbf{u}\rangle + \alpha_{k+1}(g(\mathbf{z}_k) - g(\mathbf{u})). \end{aligned} \tag{3.17}$$

Finally, we apply Lemma 3.3 on (3.17) and obtain the desired result after simplifications:

$$\begin{aligned} \alpha_{k+1}(F(\mathbf{x}_k) - F(\mathbf{u})) \leq (\alpha_{k+1}^2\eta_{k+1} - \alpha_{k+1})(F(\mathbf{y}_k) - F(\mathbf{x}_k)) \\ + \alpha_{k+1}^2[\eta_{k+1}(F(\mathbf{x}_k) - F(\mathbf{y}_{k+1})) + (L_f + \eta_{k+1})\beta\mu_k] + V_{\mathbf{z}_k}(\mathbf{u}) - V_{\mathbf{z}_{k+1}}(\mathbf{u}). \end{aligned}$$

Theorem 3.5 (Convergence). *After T iterations of Algorithm 3, without loss of generality assuming that the last iteration is a gradient correction step for*

$$\eta_{k+1} = \begin{cases} L_f, & \text{when } k+1 \text{ is a gradient correction step} \\ \max\left\{\frac{1}{4\alpha_k^2\eta_k}, \frac{L_H}{cs_k\kappa^2}\right\}, & \text{otherwise} \end{cases} \tag{3.18}$$

© xxxx Society for Industrial and Applied Mathematics

and

$$\alpha_{k+1} = \begin{cases} \frac{k+2}{2L_f}, & \textit{when } k+1 \textit{ is a gradient correction step} \\ \frac{1}{2\eta_{k+1}} + \alpha_k \sqrt{\frac{\eta_k}{\eta_{k+1}}}, & \textit{otherwise} \end{cases} \tag{3.19}$$

it holds that

$$F(\mathbf{y}_T) - F(\mathbf{x}^*) \leq \mathcal{O}\left(\frac{L_f}{T^2}\right),$$

Proof. By choosing η_{k+1} and α_{k+1} according to (3.18) and (3.19), we ensure that $\alpha_k^2\eta_k = \alpha_{k+1}^2\eta_{k+1} - \alpha_{k+1} + \frac{1}{4\eta_{k+1}}$ and $t_k = 1/(\alpha_{k+1}\eta_{k+1}) \in (0,1]$. Then telescoping Lemma 3.4 with $k = 0, 1, \ldots, T-1$ we obtain

$$\sum_{k=0}^{T-1}[\alpha_{k+1}^2\eta_{k+1}F(\mathbf{y}_{k+1}) - \alpha_k^2\eta_k F(\mathbf{y}_k) + \frac{1}{4\eta_{k+1}}F(\mathbf{y}_k)] + V_{\mathbf{z}_T}(\mathbf{u}) - V_{\mathbf{z}_0}(\mathbf{u}) \leq \sum_{k=0}^{T-1}[\alpha_{k+1}F(\mathbf{u}) + (L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k].$$

Or equivalently,

$$\alpha_T^2\eta_T F(\mathbf{y}_T) - \alpha_0^2\eta_0 F(\mathbf{y}_0) + \sum_{k=0}^{T-1}\frac{1}{4\eta_{k+1}}F(\mathbf{y}_k) + V_{\mathbf{z}_T}(\mathbf{u}) - V_{\mathbf{z}_0}(\mathbf{u}) \leq \sum_{k=0}^{T-1}\left[\alpha_{k+1}F(\mathbf{u}) + (L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k\right],$$

Then choosing $\mathbf{u} = \mathbf{x}^*$ and noticing that $F(\mathbf{y}_k) \geq F(\mathbf{x}^*)$ and $V_{\mathbf{z}_T}(\mathbf{u}) \geq 0$, for an upper bound $\Theta \geq V_{\mathbf{z}_0}(\mathbf{x}^*)$, we obtain

$$\alpha_T^2\eta_T F(\mathbf{y}_T) - \alpha_0^2\eta_0 F(\mathbf{y}_0) \leq \Theta + \sum_{k=0}^{T-1}\left[\left(\alpha_{k+1} - \frac{1}{4\eta_{k+1}}\right)F(\mathbf{x}^*) + (L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k\right].$$

Now using the fact that $\alpha_{k+1} - \frac{1}{4\eta_{k+1}} = \alpha_{k+1}^2\eta_{k+1} - \alpha_k^2\eta_k$, for $k = 0, 1, \ldots, T-1$ we simplify to

$$\alpha_T^2\eta_T F(\mathbf{y}_T) - \alpha_0^2\eta_0 F(\mathbf{y}_0) \leq \Theta + \alpha_T^2\eta_T F(\mathbf{x}^*) - \alpha_0^2\eta_0 F(\mathbf{x}^*) + \sum_{k=0}^{T-1}(L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k.$$

Then defining $\alpha_0 = 0$ and $\eta_0 = L_f$ and using the property of Bregman distances we obtain

$$\begin{aligned}\alpha_T^2\eta_T(F(\mathbf{y}_T) - F(\mathbf{x}^*)) &\leq \Theta + \alpha_0^2\eta_0(F(\mathbf{y}_0) - F(\mathbf{x}^*)) + \sum_{k=0}^{T-1}(L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k \\ &\leq \Theta + \frac{1}{L_f} + \sum_{k=0}^{T-1}(L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k.\end{aligned}$$

© xxxx Society for Industrial and Applied Mathematics

Then assuming that the last iteration T is a gradient correction step[5] and using the definitions of α_T and η_T for the left hand side, we can further simplify to

$$\frac{(T+1)^2}{4L_f}(F(\mathbf{y}_T) - F(\mathbf{x}^*)) \leq \Theta + \sum_{k=0}^{T-1}(L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k,$$

or equivalently

$$F(\mathbf{y}_T) - F(\mathbf{x}^*) \leq 4L_f\frac{\Theta + \sum_{k=0}^{T-1}(L_f + \eta_{k+1})\alpha_{k+1}^2\beta\mu_k}{(T+1)^2}.$$

Finally, choosing $\mu_k \in \left(0, \frac{\zeta}{(L_f+\eta_{k+1})\alpha_{k+1}^2\beta T}\right]$ for a small predefined $\zeta \in (0,1]$, we obtain:

$$F(\mathbf{y}_T) - F(\mathbf{x}^*) \leq \frac{4L_f(\Theta+\zeta)}{(T+1)^2}.$$

■

Remark 4. *Clearly, the constant factor of our Algorithm's worst case convergence rate is not better than that of AGM, however, as we show in the next section, in practice MAGMA can be several times faster than AGM.*

4. Numerical Experiments.

In this section we demonstrate the empirical performance of our algorithm compared to FISTA [5] and a variant of MISTA [41] on several large-scale face recognition problems. We chose those two particular algorithms, as MISTA is the only multi-level algorithm that can solve non-smooth composite problems and FISTA is the only other algorithm that was able to solve our large-scale problems in reasonable times. The source code and data we used in the numerical experiments is available from the webpage of the second author, www.doc.ic.ac.uk/∼pp500/magma.html. In Section 4.4 we also compare FISTA and MAGMA to two recent proximal stochastic algorithms: (APCG [31]) and SVRG [50]. We chose Face Recognition (FR) as a demonstrative application since (a) FR using large scale dictionaries is a relatively unexplored problem [6] and (b) the performance of large scale face recognition depends on the face resolution.

4.1. Robust Face Recognition.

In [47] it was shown that a sparse signal $\mathbf{x} \in \mathbb{R}^n$ from highly corrupted linear measurements $\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{e} \in \mathbb{R}^n$, where $\mathbf{e}$ is an unknown error and $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a highly correlated dictionary, can be recovered by solving the following ℓ_1-minimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{e}\in\mathbb{R}^m} \|\mathbf{x}\|_1 + \|\mathbf{e}\|_1 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} + \mathbf{e} = \mathbf{b}.$$

It was shown that accurate recovery of sparse signals is possible and computationally feasible even for images with nearly 100% of the observations corrupted.

[5] This assumption does not lose the generality, as we can always perform one extra gradient correction iteration

[6] FR using large scale dictionaries is an unexplored problem in ℓ_1 optimization literature due to the complexity of the (P_1) problem.

© xxxx Society for Industrial and Applied Mathematics

We introduce the new variable $\mathbf{w} = [\mathbf{x}^\top, \mathbf{e}^\top]^\top \in \mathbb{R}^{n+m}$ and matrix $\mathbf{B} = [\mathbf{A}, \mathbf{I}] \in \mathbb{R}^{m\times(m+n)}$, where $\mathbf{I} \in \mathbb{R}^{m\times m}$ is the identity matrix. The updated (P_1) problem can be written as the following optimization problem:

$$\min_{\mathbf{w}\in\mathbb{R}^{m+n}} \frac{1}{2}\|\mathbf{B}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{w}\|_1. \tag{4.1}$$

A popular application of dense error correction in computer vision, is the face recognition problem. There $\mathbf{A}$ is a dictionary of facial images stacked as column vectors[7], $\mathbf{b}$ is an incoming image of a person who we aim to identify in the dictionary and the non-zero entries of the sparse solution $\mathbf{x}^*$ (obtained from the solution of (4.1) $\mathbf{w}^* = [\mathbf{x}^{*\top}, \mathbf{e}^{*\top}]^\top$) indicate the images that belong to the same person as $\mathbf{b}$. As stated above, $\mathbf{b}$ can be highly corrupted, e.g. with noise and/or occlusion.

In a recent survey Yang et. al. [51] compared a number of algorithms solving the face recognition problem with regard to their efficiency and recognition rates. Their experiments showed that the best algorithms were the Dual Proximal Augmented Lagrangian Method (DALM), primal dual interior point method (PDIPA) and L1LS [29], however, the images in their experiments were of relatively small dimension - 40×30 pixels. Consequently the problems were solved within a few seconds. It is important to notice that both DALM and PDIPA are unable to solve large-scale (both large number of images in $\mathbf{A}$ and large image dimensions) problems, as the former requires inverting $\mathbf{A}\mathbf{A}^\top$ and the later uses Newton update steps. On the other hand L1LS is designed to take advantage of sparse problems structures, however it performs significantly worse whenever the problem is not very sparse.

4.2. MAGMA for Solving Robust Face Recognition. In order to be able to use the multilevel algorithm, we need to define a coarse model and restriction and prolongation operators appropriate for the given problem. In this subsection we describe those constructions for the face recognition problem used in our numerical experiments.

Creating a coarse model requires decreasing the size of the original fine level problem. In our experiments we only reduce n, as we are trying to improve over the performance of first order methods, whose complexity per iteration is $\mathcal{O}(n^2)$. Also the columns of $\mathbf{A}$ are highly correlated, which means that reducing n loses little information about the original problem, whereas it is not necessarily true for the rows of $\mathbf{A}$. Therefore, we introduce the dictionary $\mathbf{A}_H \in \mathbb{R}^{m\times n_H}$ with $n_H < n$. More precisely, we set $\mathbf{A}_H = \mathbf{A}\mathbf{R}_x^\top$, with $\mathbf{R}_x$ defined in (4.3). With the given restriction operator we are ready to construct a coarse model for the problem (4.1):

$$\min_{\mathbf{w}_H\in\mathbb{R}^{m+n_H}} \frac{1}{2}\| \begin{bmatrix}\mathbf{A}_H & \mathbf{I}\end{bmatrix} \mathbf{w}_H - \mathbf{b}\|_2^2 + \lambda \sum_{j=1}^{m+n_H} \sqrt{\mu_H^2 + \mathbf{w}_{H,j}^2} + \langle \mathbf{v}_H, \mathbf{w}_H\rangle, \tag{4.2}$$

where $\mathbf{w}_H = [\mathbf{x}_H^\top, \mathbf{e}^\top]^\top$ and $\mathbf{v}_H$ is defined in (3.3). It is easy to check that the coarse objective function and gradient can be evaluated using the following equations:

$$F_H(\mathbf{w}_H) = \frac{1}{2}\|\mathbf{A}_H\mathbf{x}_H + \mathbf{e} - \mathbf{b}\|_2^2 + \lambda \sum_{j=1}^{m+n_H} \sqrt{\mu_H^2 + \mathbf{w}_{H,j}^2} + \langle \mathbf{v}_H, \mathbf{w}_H\rangle,$$

[7] Facial images are, indeed, highly correlated.

© xxxx Society for Industrial and Applied Mathematics

$$\nabla F_H(\mathbf{w}_H) = \begin{bmatrix} \mathbf{A}_H^\top \mathbf{A}_H & \mathbf{A}_H^\top \\ \mathbf{A}_H & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}_H \\ \mathbf{e} \end{bmatrix} - \begin{bmatrix} \mathbf{A}_H^\top \mathbf{b} \\ \mathbf{b} \end{bmatrix} + \nabla g_H(\mathbf{w}_H) + \mathbf{v}_H,$$

where $\nabla g_H(\mathbf{w}_H)$ is the gradient of g_H defined in (2.3) and is given elementwise as follows:

$$\frac{\lambda \mathbf{w}_{H,j}}{\sqrt{\mu_H^2 + \mathbf{w}_{H,j}^2}}, \forall j = 1, 2, \ldots, n_H.$$

Hence we do not multiply matrices of size $m \times (m + n_H)$, but only $m \times n_H$.

We use a standard full weighting operator [13] as a restriction operator,

$$\mathbf{R}_x = \frac{1}{4} \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & \ldots & 0 \\ & & & \ldots & & & & 0 \\ 0 & \ldots & & 0 & 0 & 1 & 2 & 1 \end{bmatrix} \in \mathbb{R}^{\frac{n_i}{2} \times n_i} \tag{4.3}$$

and its transpose for the prolongation operator at each level $i \in \{1, ..., H-1\}$, where $n_1 = n$, $n_{i+1} = \frac{n_i}{2}$ and H is the depth of each V-cycle. However, we do not apply those operators on the whole variable $\mathbf{w}$ of the model (4.1), as this would imply also reducing m. We rather apply the operators only on part $\mathbf{x}$ of $\mathbf{w} = [\mathbf{x}^\top, \mathbf{e}^\top]^\top$, therefore our restriction and prolongation operators are of the following forms: $\mathbf{R} = [\mathbf{R}_x, \mathbf{I}]$ and $\mathbf{P} = \mathbf{R}^\top = [\mathbf{R}_x^\top, \mathbf{I}]^\top$.

In all our experiments, for the Mirr operator we chose the standard Euclidean norm $\|\cdot\|_2$ and accordingly $\frac{1}{2}\|\cdot\|_2^2$ as a Bregman divergence.

4.3. Numerical Results. The proposed algorithm has been implemented in MatLab® and tested on a PC with Intel® Core™ i7 CPU (3.4GHz×8) and 15.6GB memory.

In order to create a large scale FR setting we created a dictionary $\mathbf{A}$ of up to $8,824$ images from more than $4,000$ different people [8] by merging images from several facial databases captured in controlled and in uncontrolled conditions. In particular, we used images from MultiPIE [27], XM2VTS [32] and many in-the-wild databases such as LFW [28], LFPW [8] and HELEN [30]. In [51] the dictionary was small hence only very low-resolution images were considered. However, in large scale FR having higher resolution images is very beneficial (e.g., from a random cohort of 300 selected images, low resolution images of 40×30 achieved around 85% recognition rate, while using images of 200×200 we went up to 100%). Hence, in the remaining of our experiments the dictionary used is $\begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix}$ of $40,000 \times 48,824$ and some subsets of this dictionary.

In order to show the improvements of our method with regards to FISTA as well as to MISTA, we have designed the following experimental settings

- Using 440 facial images, then $\begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix}$ is of $40,000 \times 40,440$
- Using 944 facial images, then $\begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix}$ is of $40,000 \times 40,944$
- Using 8,824 facial images, then $\begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix}$ is of $40,000 \times 48,824$

For each dictionary we randomly chose 20 different input images $\mathbf{b}$ that are not in the dictionary and ran the algorithms with 4 different random starting points. For all of those

[8] Some people had up to 5 facial images in the database some only one.

© xxxx Society for Industrial and Applied Mathematics

	440 images	944 images	8824 images
MISTA	7	2	2
MAGMA	7	7	13

Table 4.1: Number of coarse levels used for MISTA and MAGMA for each dictionary

experiments we set the parameters of the algorithms as follows: $\kappa = 0.9$, 0.8 and 0.7 respectively for experiment settings with 440, 944 and 8,824 images in the database, $K_d = 30$ for the two smaller settings and $K_d = 50$ for the largest one, the convergence tolerance was set to $\epsilon = 10^{-6}$ for the two smaller experiments and $\epsilon = 10^{-7}$ for the largest one, $\lambda = 10^{-6}$, $\tau = 0.95$ and $s^0 = 10$. For larger dictionaries we slightly adjust κ and K_d in order to adjust for the larger problem size, giving the algorithm more freedom to perform coarse iterations. In all experiments we solve the coarse models with lower tolerance than the original problem, namely with tolerance 10^{-3}.

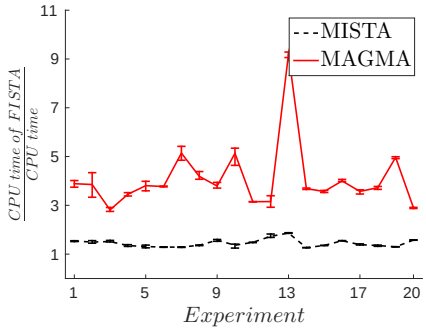
For all experiments we used as small coarse models as possible so that the corresponding algorithm could produce sparse solutions correctly identifying the sought faces in the dictionaries. Specifically, for experiments on the largest database with 8824 images we used 13 levels for MAGMA (so that in the coarse model $\mathbf{A}_H$ has only 2 columns) and only two levels for MISTA, since with more than two levels MISTA was unable to produce sparse solutions. The number of levels used in MAGMA and MISTA are tabulated in Table 4.1.

In all experiments we measure and compare the CPU running times of each tested algorithm, because comparing the number of iterations of a multi-level method against a standard first order algorithm would not be fair. This is justified as the multilevel algorithm may need fewer iterations on the fine level, but with a higher computational cost for obtaining each coarse direction. Comparing all iterations including the ones in coarser levels would not be fair either, as those are noticeably cheaper than the ones on the fine level. Even comparing the number of objective function values and gradient evaluations would not be a good solution, as on the coarse level those are also noticeably cheaper than on the fine level. Furthermore, the multilevel algorithm requires additional computations for constructing the coarse model, as well as for vector restrictions and prolongations. Therefore, we compare the algorithms with respect to CPU time.

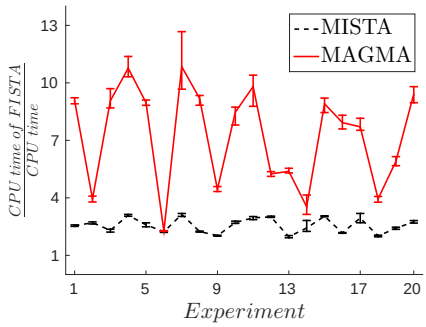
Figures 4.1a, 4.1b and 4.1c show the relative CPU running times until convergence of MAGMA and MISTA compared to FISTA[9]. The horizontal axis indicate different input images $\mathbf{b}$, whereas the vertical lines on the plots show the respective highest and lowest values obtained from different starting points. Furthermore, in Table 4.2 we present the average CPU times required by each algorithm until convergence for each problem setting, namely with databases of 440, 944 and 8824 images. As the experimental results show, MAGMA is $2-10$ times faster than FISTA. On the other hand, MISTA is more problem dependant. On some instances it can be even faster than MAGMA, but on most cases its performance is somewhere in between FISTA and MAGMA.

[9]We only report the performance of FISTA, as AGM and FISTA demonstrate very similar performances on all experiments.

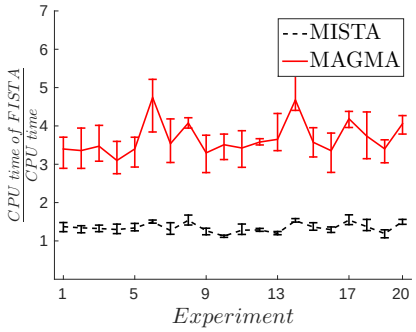
© xxxx Society for Industrial and Applied Mathematics



(a) CPU times with 440 images



(b) CPU times with 944 images



(c) CPU times with 8824 images

Figure 4.1: **Comparing MAGMA and MISTA with FISTA**. Relative CPU times of MAGMA and MISTA compared to FISTA on face recognition experiments with dictionaries of varying number of images (440, 944 and 8824). For each dictionary 20 random input images **b** were chosen, which indicate the numbers on horizontal axis. Additionally, the results with 4 random starting points for each experiment are reflected as horizontal error bars.

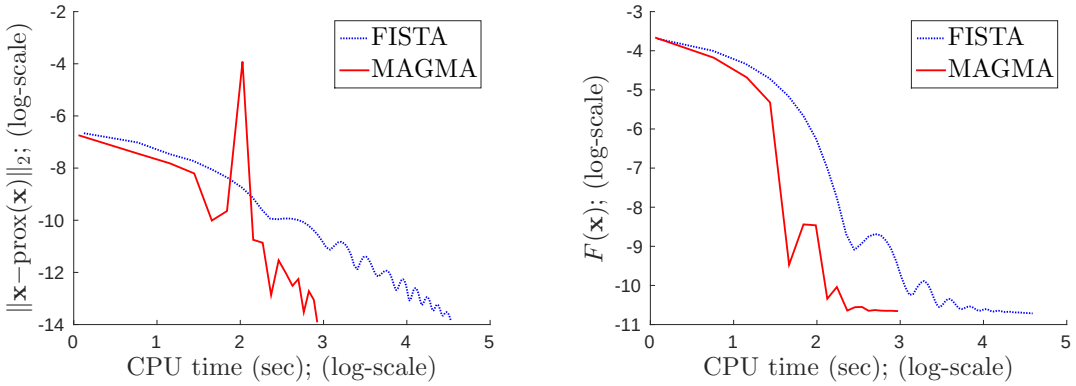
	440 images	944 images	8824 images
FISTA	98.69	175.77	1753
MISTA	68.77	70.4	1302
MAGMA	25.73	29.62	481

Table 4.2: Average CPU running times (seconds) of MISTA, FISTA and MAGMA

In order to better understand the experimental convergence speed of MAGMA, we measured the error of stopping criteria[10] and objective function values generated by MAGMA and FISTA over time from an experiment with 440 images in the dictionary. The stopping criteria are log-log-plotted in Figure 4.2a and the objective function values - in Figure 4.2b. In all

[10]We use the norm of the gradient mapping as a stopping criterion

© xxxx Society for Industrial and Applied Mathematics



(a) Gradient mapping (b) Function value

Figure 4.2: **MAGMA and FISTA convergences over time**. Comparisons of gradient mappings and function values of MAGMA, MISTA and FISTA over time.

figures the horizontal axis is the CPU time in seconds (in log scale). Note that the coarse correction steps of MAGMA take place at the steep dropping points giving large decrease, but then at gradient correction steps first large increase and then relatively constant behaviour is recorded for both plots. Overall, MAGMA reduces both objective value and the norm of the gradient mapping significantly faster.

4.4. Stochastic Algorithms for Robust Face Recognition. Since (randomized) coordinate descent (CD) and stochastic gradient (SG) algorithms are considered as state of the art for general ℓ_1 regularized least squared problems, we finish this section with a discussion on the application of these methods to the robust face recognition problem. We implemented two recent algorithms: accelerated randomised proximal coordinate gradient method (APCG) [11] [31] and proximal stochastic variance reduced gradient (SVRG) [50]. We also tried block coordinate methods with cyclic updates [7], however the randomised version of block coordinate method performs significantly better, hence we show only the performances of APCG and SVRG so as not to clutter the results.

In our numerical experiments we found that CD and SG algorithms are not suitable for robust face recognition problems. There are two reasons for this. Firstly, the data matrix contains highly correlated data. The correlation is due to the fact that we are dealing with a fine-grained object recognition problem. That is, the samples of the dictionary are all facial images of different people and the problem is to identify the particular individual. The second reason is the need to extend the standard ℓ_1 regularized least squared model so that it can handle gross errors, such as occlusions. It can be achieved by using the dense error correction formulation [47] in (4.1) (also referred to as the "bucket" model in [47]).

To demonstrate our argument we implemented APCG and SVRG and compared them with FISTA [5] and our method - MAGMA. We run all four algorithms for a fixed time and

[11] We implemented the efficient version of the algorithm that avoids full vector operations.

© xxxx Society for Industrial and Applied Mathematics

compare the achieved function values. For APCG we tried three different block size strategies, namely 1, 10 and 50. While 10 and 50 usually performed similarly, 1 was exceptionally slow, so we report the best achieved results for all experiments. For SVRG one has to choose a fixed step size as well as a number of inner iterations. We tuned both parameters to achieve the best possible results for each particular problem.

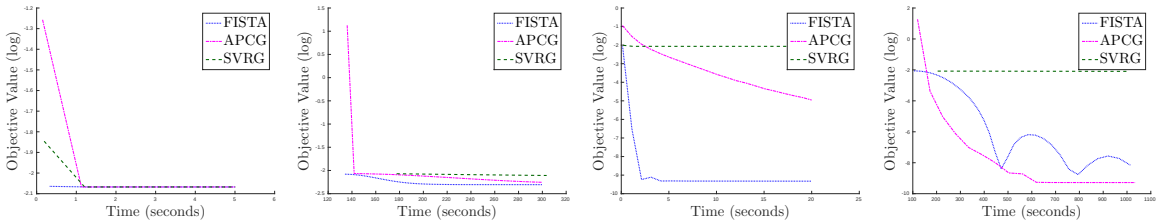
We performed the experiments on 5 databases: the 3 reported in the previous subsection (with 440, 944 and 8824 images of 200×200 dimension) and two "databases" with data generated uniformly at random with dimensions $40,000 \times 100$ and $40,000 \times 8,000$. The later experiments are an extreme case where full gradient methods (FISTA) are outperformed by both APCG and SVRG when on standard ℓ_1-regularized least squared problems, while MAGMA is not applicable. The results are given in Figure 4.3. As we can see from Figures 4.3a and 4.3b all three methods can achieve low objective values very quickly for standard ℓ_1 regularized least squared problems. However, after adding an identity matrix to the right of database (dense error correction model) the performance of all algorithms changes: they all become significantly slower due to larger problem dimensions (Figures 4.3c and 4.3d). Most noticeably SVRG stagnates after first one-two iterations. For the smaller problem (Figure 4.3c) FISTA is the fastest to converge to a minimizer, but for the larger problem (Figure 4.3d) APCG performs best. The picture is quite similar when looking at optimality conditions instead of objective function values in Figures 4.3e to 4.3h: all three algorithms, especially SVRG are slower on the dense error correction model and for the largest problem APCG performs best.

This demonstrates that for ℓ_1-regularized least squares problems partial gradient methods with a random dictionary can often be faster than accelerated full gradient methods. However, for problems with highly correlated dictionaries and noisy data, such as the robust face recognition problem, the picture is quite different.

Having established that APCG and SVRG are superior than FISTA for random data with no gross errors, we turn our attention to the problem at hand, i.e. the robust face recognition problem. First we run FISTA, APCG, SVRG and MAGMA on problems with no noise and thus the bucket model (4.1) is not used. We run all algorithms for 100 seconds on a problem with $n = 440$ images of $m = 200 \times 200 = 40,000$ dimension in the database. As the results in Figure 4.4 show all algorithms achieve very similar solutions of good quality with APCG resulting in obviously less sparse solution (Figure 4.4g).

Then we run all the tested algorithms on the same problem, but this time we simulated occlusion by adding a black square on the incoming image. Thus, we need to solve the dense error correction optimisation problem (4.1). The results are shown in Figure 4.5. The top row contains the original occluded image and reconstructed images by each algorithm and the middle row contains reconstructed noises as determined by each corresponding algorithm. FISTA (Figure 4.5b) and MAGMA (Figure 4.5e) both correctly reconstruct the underlying image while putting noise (illumination changes and gross corruptions) into the error part (Figures 4.5f and 4.5i). APCG (Figure 4.5c), on the other hand, reconstructs the true image together with the noise into the error part (Figure 4.5g), as if the sought person does not have images in the database. SVRG seems to find the correct image in the database, however it fails to separate noise from the true image (Figures 4.5d and 4.5h). We also report the reconstruction vectors $\mathbf{x}^\star$ as return by each algorithm in the bottom row. FISTA (Figure

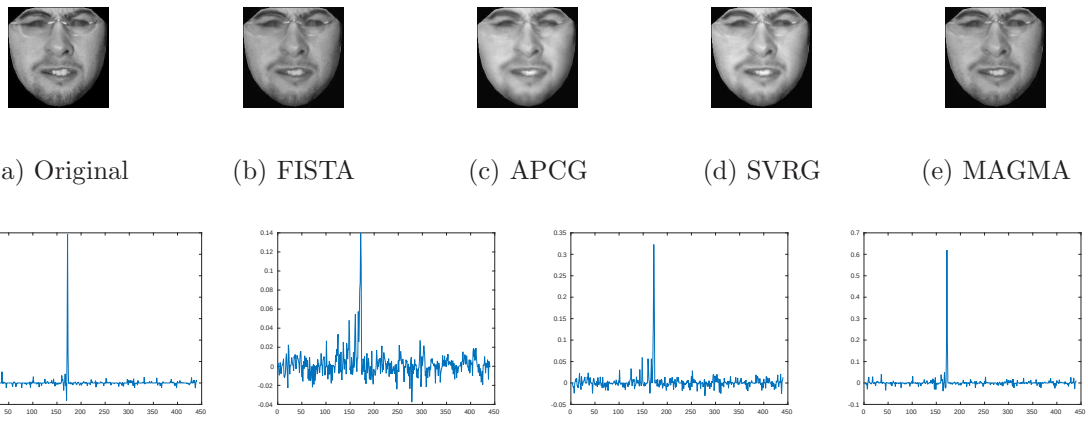
© xxxx Society for Industrial and Applied Mathematics



(a) $40,000 \times 100$ data dimension (b) $40,000 \times 8,000$ data dimension (c) $40,000 \times 100$ data dimension; bucket model (d) $40,000 \times 8,000$ data dimension; bucket model

(e) $40,000 \times 100$ data dimension (f) $40,000 \times 8,000$ data dimension (g) $40,000 \times 100$ data dimension; bucket model (h) $40,000 \times 8,000$ data dimension; bucket model

Figure 4.3: Comparing FISTA, APCG and SVRG after running them for a fixed amount of time with ((c), (d), (g) and (h)) and without ((a), (b), (e) and (f)) the bucket model on randomly generated data. The first row contains sequences of the objective values and the bottom row contains optimality criteria, i.e gradient mappings.

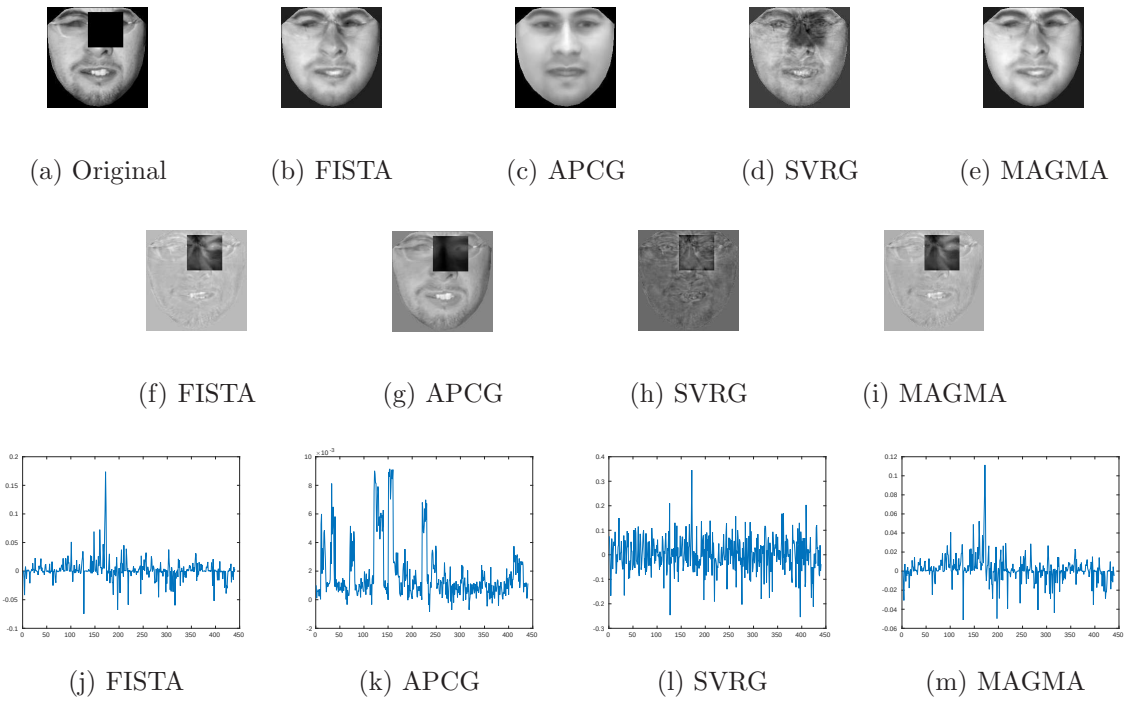


(a) Original (b) FISTA (c) APCG (d) SVRG (e) MAGMA

(f) FISTA (g) APCG (h) SVRG (i) MAGMA

Figure 4.4: Result of FISTA, APCG, SVRG and MAGMA in an image decomposition example without occlusion. Top row ((a) to (e)) contains the original input image and reconstructed images from each algorithm. The bottom row ((f) to (i)) contains the solutions $\mathbf{x}^\star$ from each corresponding algorithm.

© xxxx Society for Industrial and Applied Mathematics



(a) Original (b) FISTA (c) APCG (d) SVRG (e) MAGMA

(f) FISTA (g) APCG (h) SVRG (i) MAGMA

(j) FISTA (k) APCG (l) SVRG (m) MAGMA

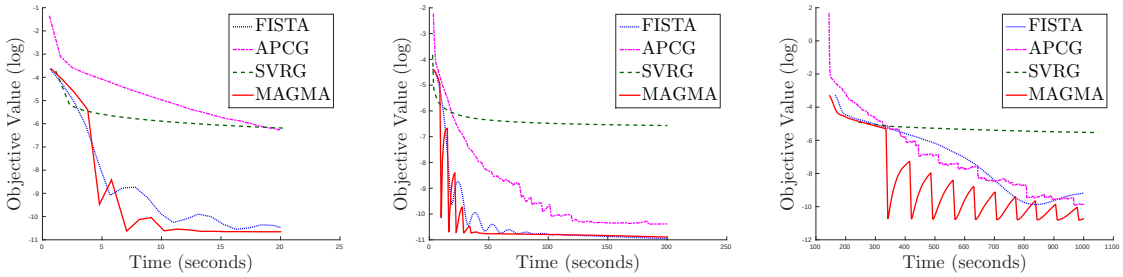
Figure 4.5: Result of FISTA, APCG, SVRG and MAGMA in an image decomposition example with corruption. Top row ((a)-(e)) contains the original occluded image and reconstructed images from each algorithm. The middle row ((f)-(i)) contains the reconstructed noise ($\mathbf{e}^\star$ reshaped as an image) as determined by corresponding algorithms. The bottom row ((j)-(m)) contains the solutions $\mathbf{x}^\star$ from each corresponding algorithm.

4.5j) and MAGMA (4.5m) both are fairly sparse with a clear spark indicating to the correct image in the database. SVRG (Figure 4.5l) also has a minor spark indicating to the correct person in the database, however it is not sparse, since it could not separate the noise. The result from APCG (Figure 4.5k) is the worst, since it is not sparse and indicates to multiple different images in the database, thus failing in the face recognition task.

Note that this is not a specifically designed case, in fact, SVRG, APCG and other algorithms that use partial information per iteration might suffer from this problem. Indeed, APCG uses one or a few columns of $\mathbf{A}$ at a time, thus effectively transforming the original large database into many tiny ones, which cannot result in good face recognition. SVRG, on the other hand, uses one row of $\mathbf{A}$ and all of variable $\mathbf{x}$ at a time for all iterations, which allows it to solve the recognition task correctly. However, when the dense error correction optimisation problem in (4.1) is used the minibatch approach results in using only one row of the identity matrix with each corresponding row of $\mathbf{A}$ and one coordinate from variable $\mathbf{e}$ together with $\mathbf{x}$, therefore resulting in poor performance when there is noise present.

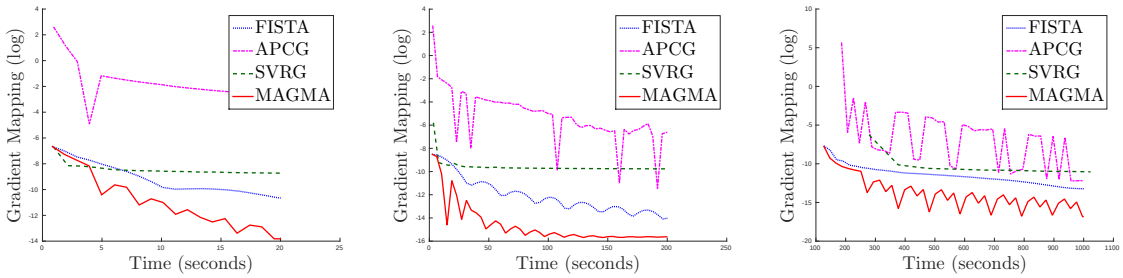
To further investigate the convergence properties of the discussed algorithms, we plot the

© xxxx Society for Industrial and Applied Mathematics



(a) 440 images (b) 944 images (c) 8824 images

Figure 4.6: Comparing the objective values of FISTA, APCG, SVRG and MAGMA after running them for a fixed amount of time on dictionaries of various sizes.



(a) 440 images (b) 944 images (c) 8824 images

Figure 4.7: Comparing the first order optimality criteria of FISTA, SVRG, APCG and MAGMA after running them for a fixed amount of time on dictionaries of various sizes.

objective function values achieved by APCG, SVRG, FISTA and MAGMA on problems with 440, 944 and 8824 images in the database in Figure 4.6 [12]. As we can see from Figures 4.6a and 4.6b APCG is slower than FISTA for smaller problems and slightly outperforms it only for the largest dictionary (Figure 4.6c). SVRG on the other hand quickly drops in function after one-two iterations, but then stagnates. MAGMA is significantly faster than all three algorithms in all experiments.

However, looking at the objective function value alone can be deceiving. In order to obtain a full picture of the performance of the algorithms for the face recognition task, one has to look at the optimality conditions and the sparsity of the obtained solution. In order to test it, we run FISTA, APCG, SVRG and MAGMA until the first order condition is satisfied with $\epsilon = 10^{-6}$ error. As the plots in Figure 4.7 indicate APCG and SVRG are slower to achieve low convergence criteria. Furthermore, as Table 4.3 shows they also produce denser solutions with higher ℓ_1-norm.

[12] For all experiments we use the same standard parameters.

© xxxx Society for Industrial and Applied Mathematics

Method	CPU Time (seconds)	Number of iterations	$\|\mathbf{x}^\star\|_1$
FISTA	1690	554	3.98
APCG	680	68,363 (block size 10)	6.67
SVRG	$\geq 10,000$	≥ 93 (epoch length 17,648)	5.19
MAGMA	526	144 fine and 120 coarse	3.09

Table 4.3: Running FISTA, APCG, SVRG and MAGMA on the largest dictionary (with 8824 images) until convergence with $\epsilon = 10^{-6}$ accuracy or maximum time of 10,000 CPU seconds returning $\mathbf{x}^*$ as a solution.

5. Conclusion and Discussion. In this work we presented a novel accelerated multi-level algorithm - MAGMA, for solving convex composite problems. We showed that in theory our algorithm has an optimal convergence rate of $\mathcal{O}(1/\sqrt{\epsilon})$, where ϵ is the accuracy. To the best of our knowledge this is the first multi-level algorithm with optimal convergence rate. Furthermore, we demonstrated on several large-scale face recognition problems that in practice MAGMA can be up to 10 times faster than the state of the art methods.

The promising results shown here are encouraging, and justify the use of MAGMA in other applications. MAGMA can be used to solve composite problems with two non-smooth parts. Another approach for applying FISTA on a problem with two non-smooth parts was given in [40]. This problem setting is particularly attractive, because of its numerous applications, including Robust PCA [15]. MAGMA's extension for solving computer vision applications of Robust PCA (such as image alignment [42]) is an interesting direction for future research.

In terms of theory, we note that even though we prove that MAGMA has the same worst case asymptotic convergence rate as Nesterov's accelerated methods, it would be interesting to see, what conditions (we expect it to be high correlation of the columns of $\mathbf{A}$) imposed on the problem ensure that MAGMA has a better convergence rate or at least a strictly better constant factor. This could also suggest an automatic way for choosing parameters κ and K_d optimally and a closed form solution for the step size s_k.

Acknowledgements. The authors would like to express sincere appreciation to the two anonymous referees for their useful suggestions and for having drawn the authors' attention to additional relevant literature. The first author's work was partially supported by Luys foundation. The work of the second author was partially supported by EPSRC grants EP/M028240, EP/K040723 and an FP7 Marie Curie Career Integration Grant (PCIG11-GA-2012-321698 SOC-MP-ES). The work of S. Zafeiriou was partially funded by EPSRC project FACER2VM (EP/N007743/1).

REFERENCES

[1] Zeyuan Allen-Zhu and Lorenzo Orecchia. A novel, simple interpretation of nesterov's accelerated method as a combination of gradient and mirror descent. *arXiv preprint arXiv:1407.1537*, 2014.

[2] Edoardo Amaldi and Viggo Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1):237–260, 1998.

[3] Gilles Aubert and Pierre Kornprobst. *Mathematical problems in image processing: partial differential equations and the calculus of variations*, volume 147. Springer Science & Business Media, 2006.
[4] Amir Beck and Marc Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *Image Processing, IEEE Transactions on*, 18(11):2419–2434, 2009.
[5] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.
[6] Amir Beck and Marc Teboulle. Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization*, 22(2):557–580, 2012.
[7] Amir Beck and Luba Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013.
[8] Peter N Belhumeur, David W Jacobs, David Kriegman, and Neeraj Kumar. Localizing parts of faces using a consensus of exemplars. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 545–552. IEEE, 2011.
[9] Aharon Ben-Tal and Arkadi Nemirovski. *Lectures on Modern Convex Optimization*. Georgia Tech, 2013.
[10] Dimitri P Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.
[11] Alfio Borzì and Volker Schulz. Multigrid methods for PDE optimization. *SIAM review*, 51(2):361–395, 2009.
[12] Dietrich Braess, Maksimillian Dryja, and W Hackbush. A multigrid method for nonconforming FE-discretisations with application to non-matching grids. *Computing*, 63(1):1–25, 1999.
[13] William L Briggs, Steve F McCormick, et al. *A multigrid tutorial*. Siam, 2000.
[14] Emmanuel J Candès. Compressive sampling. In *Proceedings oh the International Congress of Mathematicians: Madrid, August 22-30, 2006: invited lectures*, pages 1433–1452, 2006.
[15] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.
[16] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.
[17] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions on*, 52(2):489–509, 2006.
[18] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE SPM*, 25(2):21–30, 2008.
[19] Antonin Chambolle, Ronald A De Vore, Nam-Yong Lee, and Bradley J Lucier. Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage. *Image Processing, IEEE Transactions on*, 7(3):319–335, 1998.
[20] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004.
[21] David L Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, 2006.
[22] Michael Elad. *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer Science & Business Media, 2010.
[23] Heinz Werner Engl, Martin Hanke, and Andreas Neubauer. *Regularization of inverse problems*, volume 375. Springer Science & Business Media, 1996.
[24] Olivier Fercoq and Peter Richtárik. Accelerated, parallel, and proximal coordinate descent. *SIAM Journal on Optimization*, 25(4):1997–2023, 2015.
[25] Mário AT Figueiredo and Robert D Nowak. An EM algorithm for wavelet-based image restoration. *Image Processing, IEEE Transactions on*, 12(8):906–916, 2003.
[26] Serge Gratton, Annick Sartenaer, and Philippe L Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, 19(1):414–444, 2008.
[27] Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.
[28] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical

© xxxx Society for Industrial and Applied Mathematics

Report 07-49, University of Massachusetts, Amherst, 2007.

[29] Seung-Jean Kim, Kwangmoo Koh, Michael Lustig, Stephen Boyd, and Dimitry Gorinevsky. An interior-point method for large-scale l 1-regularized least squares. *Selected Topics in Signal Processing, IEEE Journal of*, 1(4):606–617, 2007.

[30] Vuong Le, Jonathan Brandt, Zhe Lin, Lubomir Bourdev, and Thomas S Huang. Interactive facial feature localization. In *Computer Vision–ECCV 2012*, pages 679–692. Springer, 2012.

[31] Qihang Lin, Zhaosong Lu, and Lin Xiao. An accelerated randomized proximal coordinate gradient method and its application to regularized empirical risk minimization. *SIAM Journal on Optimization*, 25(4):2244–2273, 2015. arXiv preprint arXiv:1407.1296.

[32] Juergen Luettin and Gilbert Maître. Evaluation protocol for the extended M2VTS database (XM2VTSDB). Technical report, IDIAP, 1998.

[33] Stephen Nash. A multigrid approach to discretized optimization problems. *Optimization Methods and Software*, 14(1-2):99–116, 2000.

[34] Arkadi Nemirovski, D-B Yudin, and E-R Dawson. Problem complexity and method efficiency in optimization. 1982.

[35] Yu Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.

[36] Yu Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[37] Yurii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983.

[38] Yurii Nesterov. *Introductory lectures on convex optimization*, volume 87. Springer Science & Business Media, 2004.

[39] Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009.

[40] Francesco Orabona, Andreas Argyriou, and Nathan Srebro. PRISMA: Proximal iterative smoothing algorithm. *arXiv preprint arXiv:1206.2372*, 2012.

[41] Panos Parpas, Duy VN Luong, Daniel Rueckert, and Berc Rustem. A multilevel proximal algorithm for large scale composite convex optimization. 2014.

[42] Yigang Peng, Arvind Ganesh, John Wright, Wenli Xu, and Yi Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2233–2246, 2012.

[43] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[44] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B*, pages 267–288, 1996.

[45] Paul Tseng. On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM Journal on Optimization*, 2008.

[46] Zaiwen Wen and Donald Goldfarb. A line search multigrid method for large-scale nonlinear optimization. *SIAM Journal on Optimization*, 20(3):1478–1503, 2009.

[47] John Wright and Yi Ma. Dense error correction via-minimization. *Information Theory, IEEE Transactions on*, 56(7):3540–3560, 2010.

[48] John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, 2010.

[49] John Wright, Allen Y Yang, Arvind Ganesh, Shankar S Sastry, and Yi Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227, 2009.

[50] Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

[51] Allen Y Yang, Zihan Zhou, Arvind Ganesh Balasubramanian, S Shankar Sastry, and Yi Ma. Fast-minimization algorithms for robust face recognition. *Image Processing, IEEE Transactions on*, 22(8):3234–3246, 2013.

[52] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse represen-

© xxxx Society for Industrial and Applied Mathematics

tation. *Image Processing, IEEE Transactions on*, 19(11):2861–2873, 2010.