SIAM J. Sci. Comput.
Vol. 39, No. 5, pp. S681–S701

A MULTILEVEL PROXIMAL GRADIENT ALGORITHM FOR A CLASS OF COMPOSITE OPTIMIZATION PROBLEMS[*]

PANOS PARPAS[†]

Abstract. Composite optimization models consist of the minimization of the sum of a smooth (not necessarily convex) function and a nonsmooth convex function. Such models arise in many applications where, in addition to the composite nature of the objective function, a hierarchy of models is readily available. It is common to take advantage of this hierarchy of models by first solving a low fidelity model and then using the solution as a starting point to a high fidelity model. We adopt an optimization point of view and show how to take advantage of the availability of a hierarchy of models in a consistent manner. We do not use the low fidelity model just for the computation of promising starting points but also for the computation of search directions. We establish the convergence and convergence rate of the proposed algorithm. Our numerical experiments on large scale image restoration problems and the transition path problem suggest that, for certain classes of problems, the proposed algorithm is significantly faster than the state of the art.

Key words. composite optimization, multigrid, nonsmooth optimization

AMS subject classifications. 90-08, 90C25, 90C26

DOI. 10.1137/16M1082299

1. Introduction.

It is often possible to exploit the structure of large scale optimization models in order to develop algorithms with lower computational complexity. We consider the case when the fidelity by which the optimization model captures the underlying application can be controlled. Typical examples include the discretization of partial differential equations in computer vision and optimal control [5], the number of features in machine learning applications [30], the number of states in a Markov decision processes [27], and nonlinear inverse problems [25]. Indeed anytime a finite dimensional optimization model arises from an infinite dimensional model it is straightforward to define such a hierarchy of optimization models. In many areas, it is common to take advantage of this structure by solving a low fidelity (coarse) model and then use the solution as a starting point in the high fidelity (fine) model. In this paper we adopt an optimization point of view and show how to take advantage of the availability of a hierarchy of models in a consistent manner. We do not use the coarse model just for the computation of promising starting points but also for the computation of search directions. We consider optimization models that consist of the sum of a smooth but not necessarily convex function and a nonsmooth convex function. These kind of problems are referred to as *composite optimization models.*

The algorithm we propose is similar to the *proximal gradient method* (PGM). There is a substantial amount of literature related to proximal algorithms, and we refer the reader to [26] for a review of recent developments. The main difference between PGM and the algorithm we propose is that we use both gradient information and a coarse model in order to compute a search direction. This modification of PGM for the computation of the search direction is akin to multigrid algorithms developed recently

[*] Received by the editors July 1, 2016; accepted for publication (in revised form) May 8, 2017; published electronically October 26, 2017.

http://www.siam.org/journals/sisc/39-5/M108229.html

Funding: The work of this author was supported by EPSRC grants EP/M028240 and EP/K040723 and a Marie Curie Career Integration Grant (PCIG11-GA-2012-321698 SOC-MP-ES).

[†] Department of Computing, Imperial College London, 180 Queens Gate, SW7 2AZ (p.parpas@imperial.ac.uk).

by a number of authors. There exists a considerable number of papers exploring the idea of using multigrid methods in optimization [5]. However, the large majority of these are concerned with solving the linear system of equations to compute a search direction using linear multigrid methods (both geometric and algebraic). A different approach, and the one we adopt in this paper, is the class of multigrid algorithms proposed in [21] and further developed in [19]. The framework proposed in [21] was used for the design of a first order unconstrained line search algorithm in [32] and a trust region algorithm in [12]. The trust region framework was extended to deal with box constraints in [11]. The general equality constrained case was discussed in [22], but no convergence proof was given. Numerical experiments with multigrid are encouraging, and a number of numerical studies have appeared so far; see, e.g., [10, 23]. The algorithm we develop combines elements from PGM (gradient proximal steps) and the multigrid framework (coarse correction steps) developed in [21] and [32]. We call the proposed algorithm *multilevel proximal gradient method* (MPGM).

The literature in multilevel optimization is largely concerned with models where the underlying dynamics are governed by differential equations, and convergence proofs exist only for the smooth case and with simple box or equality constraints. When the functions involved in the optimization problem are convex, the PGM method is identical to the so-called iterative shrinkage thresholding algorithm (ISTA). In this case, it is possible to accelerate the algorithm to achieve an improved rate of convergence. We refer the interested reader to [1] for the accelerated version of ISTA and to our recent work [15] for accelerated multilevel methods for composite convex optimization models. The main contribution of this work is the extension of the multilevel framework to composite optimization problems when the smooth function is not convex. In addition, our algorithm also allows some amount of nonsmoothness in the objective function. For example, our algorithm allows the use of indicator functions to model certain types of constraints. Theoretically, the algorithm is valid for any convex constraint, but the algorithm is computationally feasible when the proximal projection step can be performed in closed form or when it has a low computational cost. Fortunately, many problems in machine learning, computer vision, statistics, and computational chemistry do satisfy our assumptions.

The general case of models with a nonsmooth objective and constraints has not received as much attention as the smooth unconstrained case. In [11] the authors assume that the objective function is twice continuously differentiable and box constraints are handled with specialized techniques. The proximal framework we develop in this paper allows for a large class of nonsmooth optimization models. In addition, our convergence proof is different from the one given in [21] and [4] in that we do not assume that the algorithm used in the finest level performs one iteration after every coarse correction step. Our proof is based on analyzing the whole sequence generated by the algorithm and does not rely on asymptotic results as in previous works [12, 32]. Problems involving constraints appear in obstacle problems and more general variational inequality problems too. Specialized multigrid methods based on active set, Lagrangian, and penalization methods have all been proposed (see [8] for a review). A class of nonsmooth problems were addressed in [9] using a nonsmooth variant of the Newton method. The proposed method differs from the papers above in that we propose a general framework for first order algorithms that applies to a wide class of convex and nonconvex nonsmooth problems. Our method does assume that the only source of nonconvexity is present in the smooth part of the problems and that the application of the proximal operator is not computationally expensive. (Exact assumptions are given in the next section.)

Outline. The rest of the paper is structured as follows: in the next section we introduce our notation and assumptions. We also discuss the role of quadratic approximations in composite optimization models and describe our algorithm. The convergence of the algorithm is established in section 3, and we report numerical results in section 4.

2. Problem formulation and algorithm description.

The main difference between the proposed algorithm, MPGM, and existing algorithms such as the PGM is that we do not use a quadratic approximation for all iterations. Instead, we use a coarse model approximation for some iterations. In this section, we first describe the optimization problem under investigation and introduce our notation and assumptions. We then briefly review the use of quadratic approximations in the conventional PGM. We then specify how information can be transferred from the low dimensional model (coarse model) to the fine model and vice versa. Finally, we end this section by describing the proposed algorithm.

2.1. Optimization model and assumptions.

We will assume that the optimization model can be formulated using two levels of fidelity, a fine model and a coarse model. It is important to stress that the proposed algorithm aims to find the optimal solution of the (original) fine model. The coarse model is only used as an auxiliary model to construct a direction of search. The coarse model plays a similar role that the quadratic approximation model plays in the PGM.

We use h and H to indicate whether a particular quantity/property is related to the fine and coarse model, respectively. It is easy to generalize the algorithm to more levels, but with only two levels the notation is simpler. The fine model is assumed to have the following form:

$$\min_{x_h \in \mathbb{R}^h} \left\{ F_h(x) \triangleq f_h(x_h) + g_h(x_h) \right\}.$$

We make the following assumptions regarding the fine model above.

Assumption 1. The function $f_h : \mathbb{R}^h \to \mathbb{R}$ is a smooth (not necessarily convex) function with a Lipschitz continuous gradient. The Lipschitz constant will be denoted by L_h.

Assumption 2. The function $g_h : \mathbb{R}^h \to \mathbb{R}$ is a continuous convex function that is possibly nonsmooth but admits a smooth approximation (in the sense of [2, Definition 2.1]; see also Definition 2.1 below).

Assumption 3. The set $X_h^\star = \arg\min_{x_h \in \mathbb{R}^h} \ F_h(x_h)$ is not empty.

All the assumptions above are standard except the part of Assumption 2 stating that g_h admits a smooth approximation. This later statement requires some further clarification. First the definition of a smoothable convex function is given below. We refer the interested reader to [2] for further properties regarding smoothable convex functions.

DEFINITION 2.1 (smoothable convex function [2]). *The function $g_h : \mathbb{R}^h \to \mathbb{R}$ is called (α, β, K)-smoothable 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 $g_h^\mu : \mathbb{R}^h \to \mathbb{R}$ such that the following hold:*

(i) $g_h(x_h) - \beta_1 \mu \leq g_h^\mu(x_h) \leq g_h(x_h) + \beta_2 \mu \ \forall x_h \in \mathbb{R}^h$.

(ii) *The function g_h^μ has a Lipschitz continuous gradient over $\mathbb{R}^h$ and a Lipschitz constant that is less than or equal to $K + \alpha/\mu$.*

There are many ways to construct a smooth approximation for a nonsmooth function. We chose the approach described in [2] because it maintains the convexity of the function. In addition, for most functions that have a closed form expression for the proximal operator, it is well known how to construct smoothable approximation that is computationally efficient. For these reasons the smooth approximation defined above is appropriate for the class of problems we consider in this paper. Any one of the techniques reviewed in [2] can be used to construct a smooth approximation. For future reference, we define the smooth approximation of F_h as follows:

$$F_h^\mu(x) \triangleq f_h(x_h) + g_h^\mu(x_h). \tag{1}$$

The incumbent solution at iteration k in resolution h is denoted by $x_{h,k}$. We use $f_{h,k}$ and $\nabla f_{h,k}$ to denote $f_h(x_{h,k})$ and $\nabla f_h(x_{h,k})$, respectively. Unless otherwise specified we use $\|.\|$ to denote $\|.\|_2$.

2.2. Quadratic approximations and the proximal gradient method. In order to update the current solution at iteration k, PGM employs a quadratic approximation of the smooth component of the objective function and then solves the *proximal subproblem*,

$$x_{h,k+1} = \arg\min_y f_{h,k} + \langle \nabla f_{h,k}, y - x_{h,k} \rangle + \frac{L_h}{2} \left\| x_{h,k} - y \right\|^2 + g(y). \tag{2}$$

Note that the above can be rewritten as follows:

$$x_{h,k+1} = \arg\min_y \frac{L_h}{2} \left\| y - \left(x_{h,k} - \frac{1}{L_h} \nabla f_{h,k} \right) \right\|^2 + g(y).$$

When the Lipschitz constant is known, the PGM keeps updating the solution vector by solving the optimization problem above. For later use, we define the generalized *proximal operator* as follows:

$$\operatorname{prox}_h(x) = \arg\min_y \frac{1}{2} \left\| y - x \right\|^2 + g(y).$$

The *gradient mapping* at the point $x_{h,k}$ is defined as follows:

$$D_h(L_k, x_{h,k}) \triangleq L_k \left[x_{h,k} - \operatorname{prox}_h \left(x_{h,k} - \frac{1}{L_k} \nabla f_{h,k} \right) \right].$$

Note that in the definition of the gradient mapping above we use L_k instead of the actual Lipschitz constant L_h. The reason for this is that in general L_h is unknown and in most practical implementations an estimate is used instead. Using our notation, the PGM updates the current solution as follows:

$$x_{h,k+1} = x_{h,k} - \frac{1}{L_k} D_h(L_k, x_{h,k}).$$

Note that if $g(y)$ is not present, then the PGM reduces to the classical gradient method (with a fixed time step $1/L_k$). If $g(y)$ is an indicator function, then the PGM reduces to the projected gradient method. The direction of search (D_h) is obtained by solving an optimization model either in closed form or using an iterative method. It is known that the PGM algorithm described in this section converges in the sense that

$$\min_{k=1,\ldots,N} \|D_h(L_k, x_{h,k})\| \leq A/N$$

for some constant A that depends only on the Lipschitz constant of f_h and the initial point $x_{h,0}$. See [26] for a review of recent results. We note that taking the minimum of the norm of the optimality conditions is by far the most common convergence criterion for first order projected gradients methods (see also [24, Theorem 3]).

Since the nonconvex function f_h is linearized and g_h is convex, it follows that the proximal subproblem is convex. This convex approximation is solved in every iteration in order to obtain the next iterate. The algorithm we propose in this paper constructs the search direction using a different optimization model that is not necessarily convex but has a reduced number of dimensions. Before we introduce the coarse model, we need first to address the issue of how to transfer information between different levels.

2.3. Information transfer between levels. Multilevel algorithms require information to be transferred between levels. In the proposed algorithm we need to transfer information concerning the incumbent solution, proximal projection, and gradient around the current point. At the fine level, the design vector x_h is a vector in $\mathbb{R}^h$. At the coarse level the design vector is a vector in $\mathbb{R}^H$ and $H < h$. At iteration k, the proposed algorithm projects the current solution $x_{h,k}$ from the fine level to the coarse level to obtain an initial point for the coarse model denoted by $x_{H,0}$. This is achieved using a suitably designed matrix (I_h^H) as follows:

$$x_{H,0} = I_h^H x_{h,k}.$$

The matrix $I_h^H \in \mathbb{R}^{H\times h}$ is called a *restriction operator* and its purpose is to transfer information from the fine to the coarse model. There are many ways to define this operator and we will discuss some possibilities in section 3. This is a standard technique in multigrid methods both for solutions of linear and nonlinear equations and for optimization algorithms [7, 21]. In addition to the restriction operator, we also need to transfer information from the coarse model to the fine model. This is done using the *prolongation operator* $I_H^h \in \mathbb{R}^{h\times H}$. The standard assumption in multigrid literature [7] is to assume that $I_h^H = c(I_H^h)^\top$, where c is some positive scalar.

2.4. Coarse model construction. The construction of the coarse models in multilevel algorithms is a subtle process. It is this process that sets apart rigorous multilevel algorithms with performance guarantees from other approaches (e.g., kriging methods) used in the engineering literature. A key property of the coarse model is that locally (i.e., at the initial point of the coarse model, $x_{H,0}$) the optimality conditions of the two models match in a certain sense. In the unconstrained case, this is achieved by adding a linear term in the objective function of the coarse model [12, 21, 32]. In the constrained case the linear term is used to match the gradient of the Lagrangian [21]. However, the theory for the constrained case of multilevel algorithms is less developed, and the nonsmooth case has received even less attention.

For nonsmooth optimization problems, we propose a coarse model that has the following form:

$$F_H(x_H) = f_H(x_H) + g_H(x_H) + \langle v_H, x_H\rangle. \tag{3}$$

We make similar assumptions for the coarse model as we did for the fine model.

Assumption 4. The functions $f_H : \mathbb{R}^H \to \mathbb{R}$ and $g_H : \mathbb{R}^H \to \mathbb{R}$ are smooth functions with a Lipschitz continuous gradient. In addition the function g_H is convex. The Lipschitz constant for $f_H + g_H$ will be denoted by L_H.

Assumption 5. The set $X_H^\star = \arg\min_{x_H \in \mathbb{R}^H} \ F_H(x)$ is not empty.

As is standard in the multigrid literature we assume that when f_h and g_h are given the construction of f_H and g_H is possible. Finally, the third term in (3) will be used to ensure that the gradient of the smooth model is consistent with the gradient of the smooth fine model (1). The v_H term changes every time the coarse model is used, and it is given by

$$v_H = I_h^H \nabla F_{h,k}^\mu - (\nabla f_{H,0} + \nabla g_{H,0}). \tag{4}$$

It is easy to see that with the definition of v_H given above, the following first order coherency condition holds:

$$I_h^H \nabla F_{h,k}^\mu = \nabla F_{H,0}.$$

The definition of the v_H term and the so-called first order coherency condition was proposed in [21] (for the differentiable case).

To establish convergence of the algorithm, the main assumptions on the coarse model, Assumptions 4 and 5 above, together with definition (4) are enough. In particular, this means that the functions f_H and g_H may not be just reduced order versions of f_h and g_h. As an example, one can choose f_H to be a reduced order quadratic approximation of f_h. In this case, the direction obtained from the coarse model will contain information from the Hessian of f_h without having to solve a large system of equations in $\mathbb{R}^h$. This approximation is called the Galerkin approximation in the multigrid literature and has been shown to be effective in optimization in [10]. Of course, for the coarse model to be useful in practice it is important to exploit the hierarchical structure of the underlying application. Many applications that include discretized partial differential equations, image processing applications, or any application that includes an approximation of an infinite dimensional object are good candidates for the application of the proposed algorithm. Obviously, the choice of the lower dimensional model will have a great bearing on the success of the algorithm. If the underlying model has no hierarchical structure, then a multilevel method may not be appropriate as at best it may provide a good starting point.

2.5. Multilevel proximal gradient method (MPGM). Rather than computing a search direction using a quadratic approximation, we propose to construct an approximation with favorable computational characteristics for at least some iterations. In the context of optimization, favorable computational characteristics means reducing the dimensions of the problem and increasing its smoothness. This approach facilitates the use of nonlinear approximations around the current point. The motivation behind this class of approximations is that the global nature of the approximation would reflect global properties of the model that would (hopefully) yield better search directions.

At iteration k the algorithm can compute a search direction using one of two approximation techniques. The first possibility is to construct a quadratic approximation for f_h around $x_{h,k}$ in order to obtain $x_{h,k+1}$ from the solution of the proximal subproblem in (2). We then say that the algorithm performed a *gradient correction iteration.* The classical PGM only performs gradient correction iterations. The second possibility for the proposed algorithm is to use the coarse model in (3) in lieu of a quadratic approximation. When this type of step is performed we say that the algorithm performs a *coarse correction iteration.* Note that gradient correction iterations do not use the smooth approximation. The smooth approximation is only used in order to construct a smooth coarse model and for the computation of the coarse correction. The conditions that need to be satisfied for the algorithm to perform a coarse iteration are given below.

CONDITION 1. *A coarse correction iteration is performed when both conditions below are satisfied,*

$$\begin{aligned} \|I_h^H D_{h,k}\| &> \kappa \|D_{h,k}\|, \\ \|x_{h,k} - \tilde{x}_h\| &> \eta \|\tilde{x}_h\|, \end{aligned} \tag{5}$$

where $\tilde{x}_h$ is the last point to trigger a coarse correction iteration, and $\kappa, \eta \in (0,1)$.

By default, the algorithm performs gradient correction iterations as long as Condition 1 is not satisfied. Once it is satisfied, the algorithm will do a coarse correction iteration. The conditions were based on our own numerical experiments and the results in [12, 21, 32]. The first condition in (5) prevents the algorithm from performing coarse iterations when the first order optimality conditions are almost satisfied. If the current fine level iterate is close to being optimal the coarse model constructs a correction term that is nearly zero. Typically, κ is the tolerance on the norm of the first order optimality condition of (the fine) level h or alternatively $\kappa \in (0, \min(1, \min \|I_h^H\|))$. The second condition in (5) prevents a coarse correction iteration when the current point is very close to $\tilde{x}_h$. The motivation is that performing a coarse correction at a point $x_{h,k}$ that satisfies both the above conditions will yield a new point close to the current $x_{h,k}$. Note that, based on our numerical experiments, it is unlikely that the first condition will be satisfied and the second will not be.

Suppose for now that the algorithm decides to perform a coarse correction iteration at the current point $x_{h,k}$. Then the coarse model in (3) is constructed using the initial point $x_{H,0} = I_h^H x_{h,k}$. Note that the linear term in (4) changes every time the algorithm deploys the coarse model approximation. The algorithm then performs $m > 0$ iterations of the PGM algorithm on the coarse model (3). For the convergence of the algorithm it is not necessary to use PGM on the coarse model. A faster algorithm can be used, but for the sake of consistency we perform the analysis with PGM. After m iterations we obtain the *coarse correction term*,

$$e_{H,m} = x_{H,0} - x_{H,m}.$$

After the coarse correction term is computed, it is projected to the fine level using the prolongation operator and it is used as a search direction. We denote the direction of search obtained from the coarse model as $d_{h,k}$ and is defined as follows:

$$d_{h,k} \triangleq I_H^h e_{H,m}.$$

The current solution at the fine level is updated as follows:

$$x_{h,k+1} = \operatorname{prox}\left(x_{h,k}^+ - \beta_k \nabla f_h(x_{h,k}^+)\right),$$

where

$$x_{h,k}^+ = x_{h,k} - \alpha_k d_{h,k}.$$

The two step sizes α_k and β_k are selected according to an Armijo step size strategy and a backtracking strategy, respectively. Both of these step size strategies are standard and are specified in Algorithm 1 below.

Algorithm 1: Multilevel Proximal Gradient Method (MPGM)

```
1  Input:
2  Initial point: $x_{h,0}$
3  Error tolerance parameter: $\epsilon_h > 0$
4  Line search parameters: $0 < \kappa_1 < \frac{1}{2}$, $1 - \kappa_1 \leq \kappa_2 \leq 1$
5  Coarse model parameters: $m > 0$, $\mu > 0$, and $0 < \gamma < 1$
6  Initialization: Set iteration counters $k = 0$, $r = 0$
7  if Condition 1 is satisfied at $x_{h,k}$ then
8      Set $x_{H,0} = I_h^H x_{h,k}$, and use the coarse model (3) with the following linear
        term:
        (6)    $v_H = I_h^H \nabla F_{h,k}^{\mu_k} - (\nabla f_{H,0} + \nabla g_{H,0}),$
        where $\mu_k = \mu\gamma^r$.
9      For $l = 0, \ldots, m$
               $x_{H,l+1} = x_{H,l} - \beta_{H,l} \nabla F_H(x_{H,l}),$
        where $\beta_{H,l}$ is chosen so that
        (7)
        $F_{H,0} + \kappa_2 \langle \nabla F_H(x_{H,0}), x_{H,l+1} - x_{H,0} \rangle < F_H(x_{H,l+1}) \leq F_{H,l} - \kappa_1 \beta_{H,l} \|\nabla F_{H,l}\|^2.$
10     Set $d_{h,k} = I_H^h (x_{H,0} - x_{H,m})$, $r \leftarrow r + 1$ and compute
               $x_{h,k}^+ = x_{h,k} - \alpha_{h,k} d_{h,k},$
        where $\alpha_{h,k}$ is chosen such that
        (8)    $F_h(x_{h,k}^+) \leq F_{h,k} - \alpha_{h,k} \langle \nabla F_{h,k}^{\mu_k}, d_{h,k} \rangle.$
        Update the current solution,
        (9)    $x_{h,k+1} = x_{h,k}^+ - \frac{1}{L_k} D_h(L_k, x_{h,k}^+),$
        where $L_k$ is chosen such that
        (10)
        $f(x_{h,k+1}) \geq f(x_{h,k}) + \langle \nabla f(x_{h,k}), x_{h,k+1} - x_{h,k} \rangle + \frac{L_k}{2} \|x_{h,k+1} - x_{h,k}\|^2.$
        Set $k \leftarrow k + 1$ and go to step 7.
11 end
12 else
13
               $x_{h,k+1} = x_{h,k} - \frac{1}{L_k} D_h(L_k, x_{h,k}),$
        where $L_k$ is chosen to satisfy the condition in (10).
14     If
               $\min_{0 \leq j \leq k+1} \|D(L_j, x_{h,j})\|^2 < \epsilon_h,$
        then terminate.
15     Otherwise set $k \leftarrow k + 1$ and go to step 7.
16 end
```

The algorithm is fully specified in the caption above (see Algorithm 1), but a few remarks are in order. First, if Condition 1 is never satisfied, then the algorithm reduces to the proximal gradient method. Its convergence with the specific step size strategy used above will be established in the next section. Of course, Condition 1 will be satisfied, at least in some iterations, for all cases of practical interest. In this case, the main difference between the proposed algorithm and the proximal gradient algorithm is the computation of the search direction $d_{h,k}$ which is used to compute an intermediate point $x^{+}_{h,k}$. To compute the coarse correction term $d_{h,k}$, the proposed algorithm performs m iterations of a first order algorithm. In fact, the only difference between the steps performed at the coarse level and the steepest descent algorithm applied to (3) is the step size strategy in (7). The right-hand side of (7) is nothing but the standard Armijo condition. The left-hand side of (7) is the line search condition used in [32]. Finally, the termination check can easily be implemented efficiently without the need to hold all the previous incumbent solutions in memory. Indeed the only solution required to be kept in memory is the best solution encountered so far.

A point that needs to be discussed is the number of parameters that need to be specified for MPGM. Once the coarse model is specified, the only other requirement is to specify the smooth line search parameters (see line 4). These parameters are the ones that appear in standard Armijo line search procedures and are in general not too difficult to specify. In order to specify the coarse model we need to specify the γ and μ parameters. In practice we found that the algorithm is not very sensitive to these two parameters, as long as the μ_k is not allowed to become too small. A possible strategy to achieve this along with some default values that we found work very well are discussed in section 4.

The specification of the algorithm above assumes that only two levels are used. In practical implementations of the algorithm, it is beneficial to use more than two levels. However, the algorithm is easy to extend to more than two levels. In order to extend the algorithm to more than two levels, we use Algorithm 1 recursively. To be concrete, in the three-level case we take (3) as the "fine" model, construct a coarse model, and use Algorithm 1 to (approximately) solve it. Note that the second level is already smooth, so there is no need to use the smooth approximation. The same procedure is used for more than three levels.

3. Global convergence rate analysis.

In this section we establish the convergence of the algorithm under different assumptions. The first result (Theorem 3.1) covers the general case when f_h has a Lipschitz continuous gradient (but is not necessarily convex), and g_h is convex but is not necessarily smooth. In Theorem 3.1 the coarse model is not assumed to be convex. As in the single level case (see, e.g., [24, Theorem 3]) convergence is in terms of the following optimality criterion:

$$R_N = \min_{0 \le k \le N} \|D(L_{h,k}, x_{h,k})\|^2.$$

We remind the reader that the parameter μ_k appearing in the theorem below is the smoothing parameter used to specify the smooth fine model in (1).

THEOREM 3.1. *Suppose that Assumptions* 1–5 *hold and that* $L_{h,k} \ge L_h/2$. *Then the iterates of Algorithm* 1 *satisfy*

$$R_N \le \frac{2L_h}{N}\left(F_{h,0} - F_h^{\star} + \beta \sum_{k=0}^{N} \mu_k\right), \tag{11}$$

where

$$\mu_k = \begin{cases} \hat{\mu} & \textit{if Condition } 1 \textit{ was satisfied at iteration } k, \\ 0 & \textit{otherwise,} \end{cases}$$

where $\hat{\mu}$ is some positive scalar that may depend on k.

Proof. Note that if a coarse correction is performed, then according to (9), $x_{h,k+1}$ is the solution of the following optimization problem:

$$x_{h,k+1} = \arg\min_y \left\langle \nabla f_h(x_{h,k}^+), y - x_{h,k}^+ \right\rangle + \frac{L_k}{2}\|y - x_{h,k}^+\|^2 + g_h(y).$$

It follows from the optimality condition of the problem above:

$$\left\langle \nabla f_h(x_{h,k}^+) + \widetilde{g}(x_{h,k+1}) + L_k(x_{h,k+1} - x_{h,k}^+), y - x_{h,k+1} \right\rangle \geq 0,$$

where $\widetilde{g}(x_{h,k+1}) \in \partial g(x_{h,k+1})$. Rearranging the inequality above and specializing it to $y = x_{h,k}^+$, we obtain

$$\begin{aligned} \left\langle \nabla f_h(x_{h,k}^+), x_{h,k+1} - x_{h,k}^+ \right\rangle &\leq \left\langle \widetilde{g}(x_{h,k+1}), x_{h,k}^+ - x_{h,k+1} \right\rangle - L_k\|x_{h,k}^+ - x_{h,k+1}\|^2 \\ &\leq g(x_{h,k}^+) - g(x_{h,k+1}) - L_k\|x_{h,k}^+ - x_{h,k+1}\|^2, \end{aligned} \tag{12}$$

where the second inequality above follows from the subgradient inequality on g,

$$g(x_{h,k}^+) \geq g(x_{h,k+1}) + \left\langle \widetilde{g}(x_{h,k+1}), x_{h,k}^+ - x_{h,k+1} \right\rangle.$$

Since the gradient of f_h is Lipschitz continuous the descent lemma [3] yields the following inequality:

$$f_h(x_{h,k+1}) \leq f_h(x_{h,k}^+) + \left\langle \nabla f_h(x_{h,k}^+), x_{h,k+1} - x_{h,k}^+ \right\rangle + \frac{L_h}{2}\|x_{h,k}^+ - x_{h,k+1}\|^2.$$

Using the inequality above in (12), we obtain

$$F_h(x_{h,k+1}) \leq F_h(x_{h,k}^+) + \left(\frac{L_h}{2} - L_k\right)\|x_{h,k}^+ - x_{h,k+1}\|^2. \tag{13}$$

It follows from [32, Lemma 2.7] that there exists a $\beta_{H,l}$, $l = 0, \dots, m$ such that the two conditions in (7) are simultaneously satisfied, i.e., after m coarse iterations in the coarse model we have

$$\begin{aligned} F_H(x_{H,m}) &> F_{H,0} + \kappa_2\langle \nabla F_{H,0}, x_{H,m} - x_{H,0}\rangle \\ &= F_{H,0} - \kappa_2\langle \nabla F_{h,k}^{\mu_k}, d_{h,k}\rangle. \end{aligned}$$

The right-hand side of (7) implies that $F_{H,m} - F_H, 0 < 0$, and therefore $-d_{h,k}$ is a descent direction for F_h^μ at $x_{h,k}$. It follows from standard arguments (see [3]) that there exists a step size $\alpha_{h,k}$ that satisfies (8) such that

$$F(x_{h,k}^+) \leq F^\mu(x_{h,k}^+) + \beta_1\mu \leq F^\mu(x_{h,k}) + \beta_1\mu \leq F(x_{h,k}) + (\beta_1 + \beta_2)\mu_k,$$

where to obtain the last inequality we used Assumption (2). Combining the latter inequality with (13) we obtain

$$F_h(x_{h,k+1}) \le F_h(x_{h,k}) + \left(\frac{L_h}{2} - L_k\right)\|x^+_{h,k} - x_{h,k+1}\|^2 + \beta\mu_k. \tag{14}$$

Summing up all the inequalities in (14) we obtain

$$\begin{aligned} F_h(x_h^\star) \le F_h(x_{h,N+1}) \le F_h(x_{h,0}) &+ \sum_{k=0}^{N}\left(\frac{L_h}{2} - L_k\right)\|x^+_{h,k} - x_{h,k+1}\|^2 + \beta\sum_{k=0}^{N}\mu_k \\ &= F_h(x_{h,0}) + \sum_{k=0}^{N}\left(\frac{L_h - 2L_k}{L_k^2}\right)\|D_h(L_k, x_{h,k})\|^2 + \beta\sum_{k=0}^{N}\mu,k, \end{aligned}$$

where we used (9) to obtain the last equality. Using our assumption that $L_{h,k} \ge L_h/2$ and the definition of R_N we obtain

$$R_N\frac{N}{2L_h} \le F_h(x_{h,0}) - F_h(x_h^\star) + \beta\sum_{k=0}^{N}\mu_k,$$

as required. □

For the particular choice of the smoothing parameter we made in Algorithm 1 we have the following result.

COROLLARY 3.2. *Suppose that* $\mu_k \le \mu\gamma^k$ *for some* $\mu > 0$ *and* $0 < \gamma < 1$*; then*

$$R_N \le \frac{2L_h}{N}\left(F_{h,0} - F_h^\star + \beta\mu\frac{1-\gamma^{N+1}}{1-\gamma}\right).$$

Proof. This follows from the fact that the last term in (11) is a geometric series. □

Note that if the algorithm does not perform any coarse correction iterations, then the rate becomes

$$R_N \le \frac{2L_h}{N}(F_h(x_{h,0}) - F_h(x_h^\star)).$$

The rate above is the same rate obtained in [24, Theorem 3] for the nonconvex case of the gradient algorithm with a constant step size strategy. In the multilevel setting, we get the same rate but with a slightly worse constant. The augmented constant is due to the smoothing approximation. As shown in Corollary 3.2 when $\mu_k = \mu\gamma^k$ this constant is small and known in advance. In addition this constant is controllable by the user, and in most applications of interest it is negligible.

In order to simplify the derivations above we assumed that $L_{h,k} \ge L_h/2$. Of course in practice we may not know the Lipschitz constant, and this assumption may seem too restrictive. However, this can be easily addressed by using a backtracking strategy. To be precise, we start with L_0 and set $L_j = \eta^j L_{j-1}$ until (10) is satisfied. It follows from the descent lemma that such a constant exists. The convergence argument remains the same with ηL_h replacing L_h.

The analysis in this section can be extended to more than two levels. When more than two levels are used the coarse model is smooth, and the analysis is performed by essentially assuming that there is no nonsmooth component. The analysis is more complicated in the multilevel case, but the proof follows the same steps. In particular, for Theorem 3.1 to hold true for the multilevel case, we need to ensure that $F_{H,m} - F_{h,0} < 0$ remains true even if the $x_{H,m}$ is computed from another coarse model. This is easily established by using the same argument as in Theorem 3.1.

We end this section by establishing the convergence rate for the strongly convex case. In particular we change Assumptions 1 and 4 as follows.

Assumption 6. The function $f_h : \mathbb{R}^h \to \mathbb{R}$ is a smooth strongly convex function with a Lipschitz continuous gradient. The Lipschitz constant will be denoted by L_h and the strong convexity constant by γ_f.

Assumption 7. The functions $f_H : \mathbb{R}^H \to \mathbb{R}$ and $g_H : \mathbb{R}^H \to \mathbb{R}$ are smooth functions with a Lipschitz continuous gradient. In addition the function g_H is convex and f_H is strongly convex. The Lipschitz constant for $f_H + g_H$ will be denoted by L_H, and the strong convexity constant is given by γ_F.

THEOREM 3.3. *Suppose that the same assumptions as in Theorem* 3.1 *hold, except that Assumption* 1 *is replaced by Assumption* 6 *and Assumption* 4 *is replaced by Assumption* 7; *then the following hold:*

(a) *If* $\gamma_f/2L_h \leq 1$ *and* $\mu_i \leq \mu\gamma^i$ *is chosen such with* $0 < \gamma \leq 1/4$, *then*

$$F_h(x_{h,k+1}) - F_h(x^\star) \leq \left(\frac{1}{2}\right)^k \left(F_h(x_{h,0}) - F_h(x^\star) + 2\beta\mu\left(1 - \frac{1}{2^k}\right)\right).$$

(b) *If* $\gamma_f/2L_h \geq 1$ *and* $\mu_i \leq \mu\gamma^i$ *is chosen such with* $0 < \gamma < \delta$, *then*

$$F_h(x_{h,k+1}) - F_h(x^\star) \leq \left(1 - \frac{\gamma_f}{4L_h}\right)^k \left(F_h(x_{h,0}) - F_h(x^\star) + \beta\mu\frac{1-\delta^{k+1}}{1-\delta}\right),$$

where $\delta \triangleq (4\gamma_f L_h)/(4L_h - \gamma_f)$.

Proof. (a) Suppose that at iteration k a coarse correction direction is computed using steps 8 and 9. Using the descent lemma and the fact that $x_{h,k+1}$ solves the quadratic subproblem around $x_{h,k}^+$, we obtain the following:

$$\begin{aligned} F_h(x_{h,k+1}) &\leq f(x_k^+) + \left\langle \nabla f(x_k^+), x_{h,k+1} - x_{h,k}^+ \right\rangle + \frac{L_h}{2}\|x_{h,k+1} - x_{h,k}^+\|^2 + g_h(x_{h,k+1}) \\ &= \min_z f(x_k^+) + \left\langle \nabla f(x_k^+), z - x_{h,k}^+ \right\rangle + \frac{L_h}{2}\|z - x_{h,k}^+\|^2 + g_h(z) \\ &\leq \min_z f_h(z) + g_h(z) + \frac{L_h}{2}\|z - x_{h,k}^+\|^2. \end{aligned}$$

Let $z = \lambda x^\star + (1-\lambda)x_{h,k}^+$ for $0 \leq \lambda \leq 1$, and use the fact that $F_h(x) = f_h(x) + g_h(x)$ is strongly convex to obtain the following estimate:

$$\begin{aligned} F_h(x_{h,k+1}) &\leq \min_{0\leq\lambda\leq 1} \lambda F_h(x^\star) + (1-\lambda)F_h(x_k^+) + \frac{L_h}{2}\|\lambda(x^\star - x_{h,k}^+)\|^2 \\ &\leq \min_{0\leq\lambda\leq 1} F(x_k^+) - \lambda\left(1 - \frac{\lambda L_h}{\gamma_f}\right)\left(F_h(x_k^+) - F_h(x^\star)\right), \end{aligned} \tag{15}$$

where to obtain the last inequality we used the fact that for a strongly convex function, the following holds:

$$F_h(x_k^+) - F(x^\star) \geq \frac{\gamma_f}{2}\|x_k^+ - x^\star\|^2.$$

The minimum in (15) is obtained for either $\lambda = 1$ or $\lambda = \gamma_f/2L_h$. If $\lambda = 1$ (i.e., $\gamma_f/2L_h \geq 1$), then

$$\begin{aligned} F_h(x_{h,k+1}) - F_h(x^\star) &\leq \frac{L_f}{\gamma_f}\left(F_h(x^+_{h,k}) - F_h(x^\star)\right) \\ &\leq \frac{1}{2}\left(F_h(x^+_{h,k}) - F_h(x^\star)\right). \end{aligned} \tag{16}$$

Using the same arguments used in Theorem 3.1, we obtain

$$F(x^+_{h,k}) \leq F(x_{h,k}) + \beta\mu_k. \tag{17}$$

Using (17) in (16) we obtain

$$F_h(x_{h,k+1}) - F_h(x^\star) \leq \frac{1}{2}\left(F_h(x_{h,k}) - F_h(x^\star)\right) + \frac{1}{2}\beta\mu_k.$$

If a coarse correction term was not done, then we use the convention that μ_k is zero and that $x_{h,k} = x^+_{h,k}$. Therefore after k iterations and with r coarse correction steps, we obtain

$$\begin{aligned} F_h(x_{h,k+1}) - F_h(x^\star) &\leq \left(\frac{1}{2}\right)^k \left(F_h(x^{h,0}) - F_h(x^\star)\right) + \beta\mu\sum_{i=0}^{k}\frac{1}{2^{k-i}}\mu_i \\ &\leq \left(\frac{1}{2}\right)^k \left(F_h(x_{h,0}) - F_h(x^\star) + 2\beta\mu\left(1 - \frac{1}{2^k}\right)\right), \end{aligned}$$

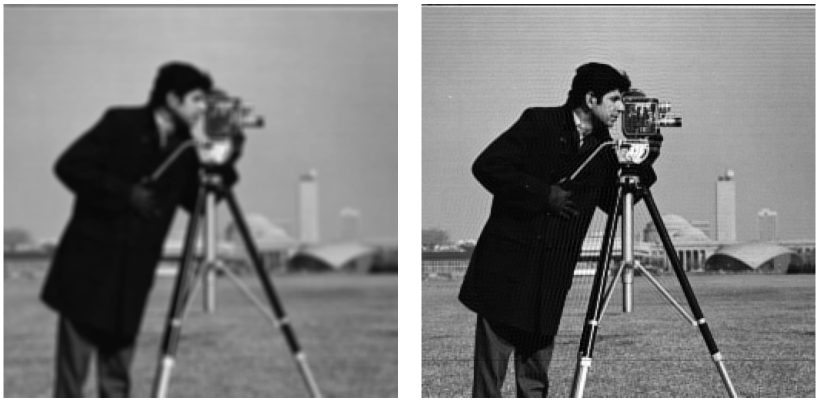
where to obtain the last inequality we used our assumption that $\mu_i \leq \mu\gamma^i$ and $\gamma \leq \frac{1}{4}$. If $\gamma_f/2L_h \leq 1$, then $\lambda = \gamma_f/2L_h$, and using the same argument as above we obtain

$$\begin{aligned} F_h(x_{h,k+1}) - F_h(x^\star) &\leq \left(1 - \frac{\gamma_f}{4L_h}\right)^k \left(F_h(x^{h,0}) - F_h(x^\star)\right) + \beta\sum_{i=0}^{k}\left(1 - \frac{\gamma_f}{4L_h}\right)^{k-i}\mu_i \\ &\leq \left(1 - \frac{\gamma_f}{4L_h}\right)^k \left(F_h(x_{h,0}) - F_h(x^\star) + \beta\mu\sum_{i=0}^{k}\left(\frac{4L_h\gamma}{4L_h - \gamma_f}\right)^i\right) \\ &= \left(1 - \frac{\gamma_f}{4L_h}\right)^k \left(F_h(x_{h,0}) - F_h(x^\star) + \beta\mu\frac{1 - \delta^{k+1}}{1 - \delta}\right), \end{aligned}$$

as required. ☐

4. Numerical experiments.

In this section we illustrate the numerical performance of the algorithm on both convex and nonconvex problems. For the convex problem we chose the image restoration problem, and for the nonconvex case we chose the problem of finding transition paths in energy landscapes. Both problems have been used as benchmark problems and are important applications in image processing and computational chemistry, respectively. The two problem classes are quite different. In particular, the image restoration problem is convex and unconstrained but has more than a million variables, whereas the saddle point problem is nonconvex, constrained but is lower dimensional. We compare the performance of the proposed MPGM against the state of the art for image restoration problems (FISTA [1]) and against the Newton algorithm for the transition path problem. In our implementation of MPGM we only considered V-cycles. Other possible strategies may yield improved results, and this is an interesting topic for future work.



(a) Corrupted image with 0.5% noise

(b) Restored image

FIG. 1. (a) *Corrupted cameraman image used as the input vector b.* (b) *Restored image.*

4.1. The image restoration problem. The image restoration problem consists of the following composite convex optimization model:

$$\min_{x_h \in \mathbb{R}^h} \|A_h x_h - b_h\|_2^2 + \lambda_h \|W(x_h)\|_1,$$

where b_h is the vectorized version of the input image, A_h is the blurring operator based on the point spread function and reflexive boundary conditions, and $W(x_h)$ is the wavelet transform of the image. In our numerical experiments, the image in the fine model has resolution 1024×1024. The two dimensional version of the input image and the restored image are denoted by X_h and B_h, respectively. The first term in the objective function aims to find an image that is as close to the original image as possible, and the second term enforces a relationship between the pixels and ensures that the recovered image is neither blurred nor noisy. The regularization parameter λ_h is used to balance the two objectives. In our implementation of the fine model we used $\lambda_h = 10e-4$. Note that the first term is convex and differentiable, and the second term is also convex but nonsmooth. The blurring operator A_h is computed by utilizing an efficient implementation provided in the HNO package [13]. In particular, we rewrite the expensive matrix computation $A_h x_h - b_h$ in the reduced form,

$$A_h^c X_h (A_h^r)^\top - B_h,$$

where A_h^c, A_h^r are the row/column blurring operators and $A_h = A_h^r \otimes A_h^c$. We illustrate the problem of image restoration using the widely used cameraman image. Figure 1(a) is the corrupted image, and the restored image is shown in Figure 1(b). The restored image was computed with MPGM. The image restoration problem fits exactly the framework of convex composite optimization. In addition, it is easy to define a hierarchy of models by varying the resolution of the image. We discuss the issue of coarse model construction next.

4.1.1. The coarse model in the image restoration problem. We described MGPM as a two-level algorithm, but it is easy to generalize it to many levels. In our computations we used the fine model described above and two coarse models, one with resolution 512×512 and its coarse version, i.e., a model with 256×256. Each model on the hierarchy has a quarter of the variables of the model above it. We used the same algorithm parameters for all levels. We used the smoothing approach to construct the coarse model (see section 2.4). Following the smoothing approach we used the following objective function:

$$\min_{x_H \in \Omega_H} \|A_H x_H - b_H\|_2^2 + \langle v_H, x_H \rangle + \lambda_H \sum_{i \in \Omega_H} \sqrt{W(x_H)_i^2 + \mu^2} - \mu,$$

where $\mu = 0.2$ is the smoothing parameter, v_H was defined in (4), and λ_H is the regularizing parameter for the coarse model. The coarse model has fewer variables and is smoother; therefore the regularizing parameter should be reduced. In our experiments we used $\lambda_H = \lambda_h/2$. The parameter μ_k used in the definition of the v_H term in (6) was chosen as $\mu_k = \max\{0.001, \mu\gamma^r\}$, where $\mu = 0.2$ and $\gamma = 0.8$.

The information transfer between levels is done via a simple linear interpolation technique to group four fine pixels into one coarse pixel. The prolongation operator is given by $I_h^h = R_1 \otimes R_1^\top$. The matrix R_1 is specified using pencil notation as follows:

$$R_1 = \Big]\ \tfrac{1}{2} \quad 1 \quad \tfrac{1}{2}\ \Big[.$$

As usual we have $I_h^H = c(I_H^h)^\top$ with $c = 1/4$. This is a standard way to construct the restriction and prolongation operators, and we refer the reader to [7] for the details. The input image and the current iterate are restricted to the coarse scale as follows:

$$x_{H,0} = I_h^H x_{h,k}, \qquad b_H = I_h^H b_h.$$

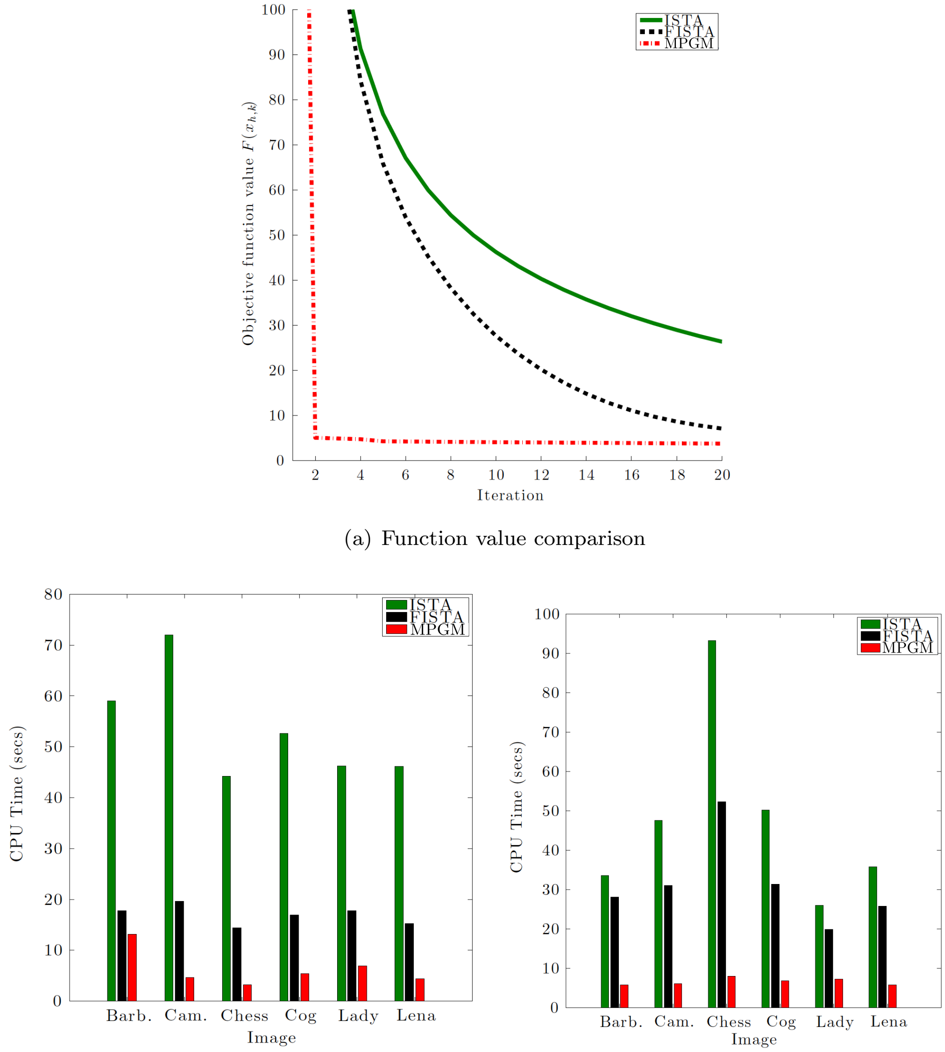
The standard matrix restriction $A_H = I_h^H A_h (I_h^H)^\top$ is not performed explicitly as we never need to store the large matrix A_h. Instead, only column and row operators A_h^c, A_h^r are stored in memory. The coarse blurring matrix is given by

$$A_H = A_H^r \otimes A_H^c,$$

where $A_H^c = R_1 A_h^c R_1^\top$ and $A_H^r = R_1 A_h^r R_1^\top$.

The condition to use the coarse model in MGPM is specified in (5), and we used the parameters $\kappa = 0.5$ and $\eta = 1$ in our implementation.

4.1.2. Performance comparison. We compare the performance of our methods with FISTA and ISTA using a representative set of corrupted images (blurred with 0.5% additive noise). FISTA is considered to be a state of the art method for the image restoration problem [1] and has a theoretically better convergence rate than the proposed algorithm. ISTA has the same convergence rate as the proposed algorithm. We compare the CPU time required to achieve convergence of MPGM against ISTA and FISTA. The solution tolerance is $e_h = 2\%$ for all algorithms. We chose to report CPU times since the computational complexity of MPGM per iteration can be larger than ISTA or FISTA. We tested the algorithm on several images, and below we report results on a representative set of six images. All our test images have the same size, 1024 × 1024. At this resolution, the optimization model at the fine scale has more than 10^6 variables (1,048,576, to be precise). We implemented the ISTA and FISTA algorithms with the same parameter settings as [1]. For the fine model we used the standard backtracking line strategy for ISTA as in [1]. All algorithms were implemented in MATLAB and run on a standard desktop PC. Due to space limitations, we only report detailed convergence results from the widely used cameraman image. The images we used, the source code for MGPM, and further numerical experiments can be obtained from the web page of the author https://www.doc.ic.ac.uk/~pp500.



(a) Function value comparison

(b) Images blurred with 0.48% noise

(c) Images blurred with 1% noise

FIG. 2. (a) *Comparison of the three algorithms in terms of function value. MPGM clearly outperforms the other algorithms and converges in essentially* 5 *iterations, while others have not converged even after* 100 *iterations. CPU time required to find a solution within* 2% *of the optimum for the three algorithms.* (b) *Results for blurred images with* 0.5% *noise.* (c) *Results for blurred images with* 1% *noise. Higher levels of noise lead to more ill conditioned problems. The figures in* (b) *and* (c) *compare CPU times and suggest that MPGM is on average ten times faster than ISTA and three/four times than FISTA.*

In Figure 2(a) we compare the three algorithms in terms of the progress they make in function value reduction. In this case we see that MPGM clearly outperforms ISTA. This result is not surprising since MPGM is a more specialized algorithm with the same convergence properties. However, what is surprising is that MPGM still outperforms the theoretically superior FISTA. Clearly, MPGM outperforms FISTA in early iterations and is comparable in latter iterations.

Figure 2 gives some idea of the performance of the algorithm, but of course what matters most is the CPU time required to compute a solution. This is because an iteration of MPGM requires many iterations in a coarse model, and therefore comparing the algorithms in terms of the number of iterations is not fair. In order to level the playing field, we compare the performance of the algorithms in terms of CPU time required to find a solution that satisfies the optimality conditions within 2%. Two experiments were performed on a set of six images. The first experiment takes as input a blurred image with 0.5% additive Gaussian noise, and the second experiment uses 1% additive noise. We expect the problems with the 1% additive noise to be more difficult to solve than the one with 0.5% noise. This is because the corrupted image is more ill conditioned. Figure 2(b) shows the performance of the three algorithms on blurred images with 0.5% noise. We can see that MPGM outperforms both ISTA and FISTA by some margin. On average MPGM is four times faster than FISTA and ten times faster than ISTA. In Figure 2(c), we see an even greater improvement of MPGM over ISTA/FISTA. This is expected since the problem is more ill conditioned (with 1% noise as opposed to 0.5% noise in Figure 2(c)), and so the fine model requires more iterations to converge. Since ISTA/FISTA perform all their computation with the ill conditioned model, CPU time increases as the amount of noise in the image increases. On the other hand, the convergence of MPGM depends less on how ill conditioned the model is since one of the effects of averaging is to decrease ill conditioning.

4.2. The transition path problem. The computational of transition paths is a fundamental problem in computational chemistry, material science, and biology. Most methods for computing fall into two broad classes: chain of state methods and surface walking methods. For a recent review and a description of several methods in the latter category we refer the interested reader to [33]. We approach the problem using a chain of state method. In these methods we are given two initial points x_a and x_b that are stable points on a potential energy surface. Usually these two points are local minima. The transition path problem in this case is to find how the geometry and the energy changes from one stable state to the next.

The mathematical formulation of the transition path problem using an optimization problem was established in [20] through the mountain pass theorem. We will adopt the formulation from [20]. We will review the formulation of the optimization problem below, but for the precise properties of the model we refer the reader to [20]. The method (also known as the "elastic string algorithm") is derived from the solution of the following infinite dimensional saddle point problem:

$$\inf_{p\in\Gamma} \max_{t\in[0,1]} \; f(p(t)), \tag{18}$$

where f is the nonconvex potential energy function, and Γ is the set of all paths connecting x_a and x_b. We also impose the initial and final conditions $p(0) = x_a$ and $p(1) = x_b$. By only considering piecewise linear paths with m breakpoints, the elastic string algorithm (described in detail in [20]) reduces the infinite dimensional problem in (18) to the following finite dimensional problem:

$$\min\{v(x) | \|x_{k+1} - x_k\|_1 \le h_k,\ 0 \le k \le m,\ x_0 = x_a,\ x_m = x_b\}, \tag{19}$$

where $v(x) = \max(f(x_0), \ldots, f(x_m))$. In order to place (19) in the same format as the optimization models we consider in this paper we use entropic smoothing for the max function [28] and replace the constraints with an indicator function. The entropic smoothing function is given by

$$\phi_\lambda(x) = \frac{1}{\lambda}\log\left(\sum_{k=0}^{m} \exp(\lambda f_k(x))\right).$$

This is a frequently used function to smooth the max function appearing in minimax problems; for its properties see [28, 31]. We note that the proximal operator associated with the constraints of the problem in (19) can be computed in closed form. However to define the coarse model we need to smooth the indicator function. To smooth the indicator function we used the Moreau–Yosida regularization approach [14]. To be precise, letting $g(x)$ denote the indicator function of the problem in (19), the Moreau–Yosida regularization of g (also referred to as the Moreau envelope) is given by

$$g_\mu(x) = \inf_w \left\{ g(w) + \frac{1}{2\mu}\|w - x\|^2 \right\}.$$

Let $\widehat{x}$ denote the unique minimizer of the problem above, and then the gradient of $g_\mu(x)$ is Lipschitz continuous and is given by

$$\nabla g_\mu(x) = \frac{1}{\mu}[x - \widehat{x}].$$

We used the following smoothing parameters $\lambda = \mu = 0.01$. For the fine model we used 2^h breakpoints with $h = 10$. This means that for a potential energy function in n dimensions the optimization problem has $n \times 2^{10}$ variables and 2^{10} constraints. We solved the fine model using the same method as in [20], i.e., the Newton's method with an Armijo line search. Before we compare the Newton method with the proposed algorithm we describe the construction of the coarse model.

4.2.1. The coarse model in the transition path problem. As for the image restoration problem, the coarse model for the transition path problem is very easy to construct. We used eight levels, with $2^2, 2^3, \ldots, 2^9$ breakpoints. For the prolongation and restriction operators we used the same simple linear interpolation technique as in the image restoration problem. Finally, we used $\kappa = 0.5$ and $\eta = 1$ for the coarse correction criteria. (These are the same parameters we used in the image restoration problem.) We used the same strategy to select μ_k as the one described in section 4.1.1.

4.2.2. Performance comparison. We tested the performance of MPGM against the Newton algorithm on seven widely used benchmark problems. We used the three test problems from [20]. For these three potentials we used the same parameters and functional form described in [20], and for the interest of space we omit the details here. We also used four test problems from [33]. For later reference we tabulate the name of the problem and the appropriate reference in Table 1. As in the previous case study, comparing the performance of the proposed algorithm in terms of CPU time is the fairest metric. In Figure 3 we plot the results. It can be seen from this figure that on all but two problems the proposed algorithm is far superior than the Newton algorithm.

5. Conclusions. We developed an MPGM for composite optimization models. The key idea behind MPGM is, for some iterations, to replace the quadratic approximation with a coarse approximation. The coarse model is used to compute search directions that are often superior to the search directions obtained using just gradient information. We showed how to construct coarse models in the case where the objective function is nondifferentiable. For the convex case, our initial numerical experiments show that the proposed MISTA algorithm is on average ten times faster than ISTA and three-four times faster (on average) than the theoretically superior FISTA algorithm. For the nonconvex case, we tested the performance of the

TABLE 1
Test problems in numerical experiments.

Problem #	Name	Reference
1	Camel Function	[20]
2	LEPS potential	[20]
3	LEPS/Gaussian potential	[20]
4	2-d energy function	[33]
5	Minyaev–Quapp surface	[33]
6	Stingray function	[33]
7	Eckhardt surface	[33]

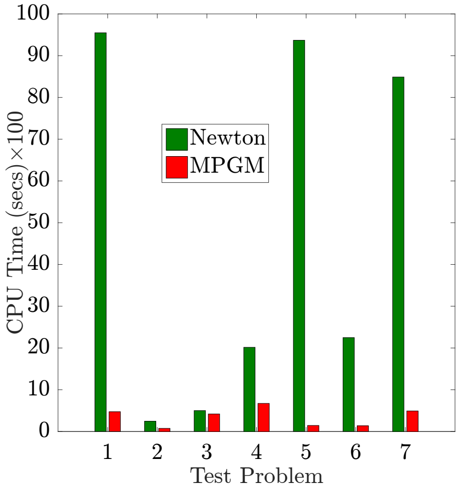


FIG. 3. *Comparison of MPGM with the Newton algorithm in terms of CPU time.*

algorithm on a problem arising in computational chemistry, and we showed that the proposed method is significantly faster than the Newton algorithm.

The initial numerical results are promising, but still the algorithm can be improved in a number of ways. For example, we only considered the most basic prolongation and restriction operators in approximating the coarse model. The literature on the construction of these operators is quite large, and there exist more advanced operators that adapt to the problem data and current solution (e.g., bootstrap AMG [6]). We expect that the numerical performance of the algorithm can be improved if these advanced techniques are used instead of the naive approach proposed here. In the last few years several algorithmic frameworks for large scale composite convex optimization have been proposed. Examples include active set methods [18], stochastic methods [16], Newton type methods [17], and block coordinate descent methods [29]. In principle, all these algorithmic frameworks could be combined with the multilevel framework developed in this paper. Based on the theoretical and numerical results obtained from the multilevel version of the proximal gradient method, we are hopeful that the multilevel framework can improve the numerical performance of many of the recent algorithmic developments in large scale composite optimization.

Acknowledgments. The author acknowledges the help of Ryan Duy Luong with the numerical experiments in section 4. We also thank the editors and anonymous referees for their helpful comments on earlier versions of the paper. The author was partially supported by EPSRC grants EP/M028240.

REFERENCES

[1] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sci., 2 (2009), pp. 183–202.

[2] A. Beck and M. Teboulle, *Smoothing and first order methods: A unified framework*, SIAM J. Optim., 22 (2012), pp. 557–580.

[3] D.P. Bertsekas, *Nonlinear Programming*, Optim. Comput. Ser., Athena Scientific, Belmont, MA, 1999.

[4] A. Borzì, *On the convergence of the MG/OPT method*, Proc. Appl. Math. Mech., 5 (2005), pp. 735–736.

[5] A. Borzì and V. Schulz, *Multigrid methods for PDE optimization*, SIAM Rev., 51 (2009), pp. 361–395.

[6] A. Brandt, J. Brannick, K. Kahl, and I. Livshits, *Bootstrap AMG*, SIAM J. Sci. Comput., 33 (2011), pp. 612–632.

[7] W.L. Briggs, V.E. Henson, and S.F. McCormick, *A Multigrid Tutorial*, 2nd ed., SIAM, Philadelphia, 2000.

[8] C. Gräser and R. Kornhuber, *Multigrid methods for obstacle problems*, J. Comput. Math., 27 (2009), pp. 1–44.

[9] C. Gräser, U. Sack, and O. Sander, *Truncated nonsmooth Newton multigrid methods for convex minimization problems*, in Domain Decomposition Methods in Science and Engineering XVIII, Springer, New York, 2009, pp. 129–136.

[10] S. Gratton, M. Mouffe, A. Sartenaer, P.L. Toint, and D. Tomanos, *Numerical experience with a recursive trust-region method for multilevel nonlinear bound-constrained optimization*, Optim. Methods Softw., 25 (2010), pp. 359–386.

[11] S. Gratton, M. Mouffe, P.L. Toint, and M. Weber-Mendonça, *A recursive-trust-region method for bound-constrained nonlinear optimization*, IMA J. Numer. Anal., 28 (2008), pp. 827–861.

[12] S. Gratton, A. Sartenae, and P.L. Toint, *Recursive trust-region methods for multiscale nonlinear optimization*, SIAM J. Optim., 19 (2008), pp. 414–444.

[13] P.C. Hansen, J.G. Nagy, and D.P. O'Leary, *Deblurring Images: Matrices, Spectra, and Filtering*, Vol. 3, SIAM, Philadelphia, 2006.

[14] J.B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms* I: *Fundamentals*, Vol. 305, Springer, New York, 2013.

[15] V. Hovhannisyan, P. Parpas, and S. Zafeiriou, *MAGMA: Multilevel accelerated gradient mirror descent algorithm for large-scale convex composite minimization*, SIAM J. Imaging Sci., 9 (2016), pp. 1829–1857.

[16] G. Lan, *An optimal method for stochastic composite optimization*, Math. Program., 133 (2012), pp. 365–397.

[17] J.D. Lee, Y. Sun, and M.A. Saunders, *Proximal Newton-type Methods for Minimizing Composite Functions*, SIAM J. Optim., 24 (2014), pp. 1420–1443.

[18] A.S. Lewis and S.J. Wright, *A Proximal Method for Composite Minimization*, Math. Program., 158 (2016), pp. 501–546.

[19] R.M. Lewis and S.G. Nash, *Model problems for the multigrid optimization of systems governed by differential equations*, SIAM J. Sci. Comput., 26 (2005), pp. 1811–1837.

[20] J.J. Moré and T.S. Munson, *Computing mountain passes and transition states*, Math. Program., 100 (2004), pp. 151–182.

[21] S.G. Nash, *A multigrid approach to discretized optimization problems*, Optim. Methods Softw., 14 (2000), pp. 99–116.

[22] S.G. Nash, *Properties of a class of multilevel optimization algorithms for equality-constrained problems*, Optim. Methods Softw., 29 (2014), pp. 137–159.

[23] S.G. Nash and R.M. Lewis, *Assessing the performance of an optimization-based multilevel method*, Optim. Methods Softw., 26 (2011), pp. 693–717.

[24] Y. Nesterov, *Gradient methods for minimizing composite objective function*, Math. Program., 140 (2013), pp. 125–161.

[25] S. Oh, A.B. Milstein, C.A. Bouman, and K.J. Webb, *A general framework for nonlinear multigrid inversion*, IEEE Trans. Image Process., 14 (2005), pp. 125–140.

[26] N. Parikh and S. Boyd, *Proximal algorithms*, Found. Trends Optim., 1 (2014), pp. 127–239.

[27] P. Parpas and M. Webster, *A stochastic multiscale model for electricity generation capacity expansion*, European J. Oper. Res., 232 (2014), pp. 359–374.

[28] E. Polak, R.S. Womersley, and H.X. Yin, *An algorithm based on active sets and smoothing for discretized semi-infinite minimax problems*, J. Optim. Theory Appl., 138 (2008), pp. 311–328.

[29] P. Richtárik and M. Takáč, *Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function*, Math. Program., 144 (2014), pp. 1–38.

[30] J.J. Thiagarajan, K.N. Ramamurthy, and A. Spanias, *Learning stable multilevel dictionaries for sparse representation of images*, IEEE Trans. Neural Netw. Learn. Syst., under review, 2013.

[31] A. Tsoukalas, P. Parpas, and B. Rustem, *A smoothing algorithm for finite min–max–min problems*, Optim. Lett., 3 (2009), pp. 49–62.

[32] Z. Wen and D. Goldfarb, *A line search multigrid method for large-scale nonlinear optimization*, SIAM J. Optim., 20 (2009), pp. 1478–1503.

[33] Z. Lei, Q. Du, and Z. Zheng, *Optimization-based shrinking dimer method for finding transition states*, SIAM J. Sci. Comput., 38 (2016), pp. A528–A544.