SIAM J. IMAGING SCIENCES
Vol. 7, No. 2, pp. 1388–1419

iPiano: Inertial Proximal Algorithm for Nonconvex Optimization*

Peter Ochs[†], Yunjin Chen[‡], Thomas Brox[†], and Thomas Pock[‡]

Abstract. In this paper we study an algorithm for solving a minimization problem composed of a differentiable (possibly nonconvex) and a convex (possibly nondifferentiable) function. The algorithm iPiano combines forward-backward splitting with an inertial force. It can be seen as a nonsmooth split version of the Heavy-ball method from Polyak. A rigorous analysis of the algorithm for the proposed class of problems yields global convergence of the function values and the arguments. This makes the algorithm robust for usage on nonconvex problems. The convergence result is obtained based on the Kurdyka–Łojasiewicz inequality. This is a very weak restriction, which was used to prove convergence for several other gradient methods. First, an abstract convergence theorem for a generic algorithm is proved, and then iPiano is shown to satisfy the requirements of this theorem. Furthermore, a convergence rate is established for the general problem class. We demonstrate iPiano on computer vision problems—image denoising with learned priors and diffusion based image compression.

Key words. nonconvex optimization, Heavy-ball method, inertial forward-backward splitting, Kurdyka–Łojasiewicz inequality, proof of convergence

AMS subject classifications. 32B20, 47J06, 47J25, 47J30, 49M15, 62H35, 65K10, 90C53, 90C26, 90C06, 90C30, 94A08

DOI. 10.1137/130942954

1. Introduction.

The gradient method is certainly one of the most fundamental but also one of the most simple algorithms to solve smooth convex optimization problems. In the last several decades, the gradient method has been modified in many ways. One of those improvements is to consider so-called multistep schemes [38, 35]. It has been shown that such schemes significantly boost the performance of the plain gradient method. Triggered by practical problems in signal processing, image processing, and machine learning, there has been an increased interest in so-called composite objective functions, where the objective function is given by the sum of a smooth function and a nonsmooth function with an easy-to-compute proximal map. This initiated the development of the so-called proximal gradient or forward-backward method [28], which combines explicit (forward) gradient steps w.r.t. the smooth part with proximal (backward) steps w.r.t. the nonsmooth part.

In this paper, we combine the concepts of multistep schemes and the proximal gradient method to efficiently solve a certain class of *nonconvex*, nonsmooth optimization problems.

*Received by the editors October 28, 2013; accepted for publication (in revised form) April 2, 2014; published electronically June 17, 2014. The first and third authors acknowledge funding by the German Research Foundation (DFG grant BR 3815/5-1).

http://www.siam.org/journals/siims/7-2/94295.html

[†]Department of Computer Science and BIOSS Centre for Biological Signalling Studies, University of Freiburg, Georges-Köhler-Allee 052, 79110 Freiburg, Germany (ochs@cs.uni-freiburg.de, brox@cs.uni-freiburg.de).

[‡]Institute for Computer Graphics and Vision, Graz University of Technology, Inffeldgasse 16, A-8010 Graz, Austria (cheny@icg.tugraz.at, pock@icg.tugraz.at). The second and fourth authors were supported by the Austrian science fund (FWF) under the START project *BIVISION*, Y729.

Although the transfer of knowledge from convex optimization to nonconvex problems is very challenging, it aspires to find efficient algorithms for certain nonconvex problems. Therefore, we consider the subclass of nonconvex problems

$$\min_{x\in\mathbb{R}^N} \; f(x) + g(x)\,,$$

where g is a *convex (possibly nonsmooth)* and f is a *smooth (possibly nonconvex)* function. The sum $f+g$ comprises nonsmooth, nonconvex functions. Despite the nonconvexity, the structure of f being smooth and g being convex makes the forward-backward splitting algorithm well defined. Additionally, an inertial force is incorporated into the design of our algorithm, which we termed *iPiano*. Informally, the update scheme of the algorithm that will be analyzed is

$$x^{n+1} = (I+\alpha\partial g)^{-1}(x^n - \alpha\nabla f(x^n) + \beta(x^n - x^{n-1}))\,,$$

where α and β are the step size parameters. The term $x^n - \alpha\nabla f(x^n)$ is referred to as the *forward step*, $\beta(x^n - x^{n-1})$ as the *inertial term*, and $(I+\alpha\partial g)^{-1}$ as the backward or *proximal step*.

For $g \equiv 0$ the proximal step is the identity, and the update scheme is usually referred to as the *Heavy-ball method*. This reduced iterative scheme is an explicit finite differences discretization of the so-called *Heavy-ball with friction* dynamical system

$$\ddot{x}(t) + \gamma\dot{x}(t) + \nabla f(x(t)) = 0\,.$$

It arises when Newton's law is applied to a point subject to a constant friction $\gamma > 0$ (of the velocity $\dot{x}(t)$) and a gravity potential f. This explains the name "Heavy-ball method" and the interpretation of $\beta(x^n - x^{n-1})$ as inertial force.

Setting $\beta = 0$ results in the forward-backward splitting algorithm, which has the nice property that in each iteration the function value decreases. Our convergence analysis reveals that the additional inertial term prevents our algorithm from monotonically decreasing the function values. Although this may look like a limitation on first glance, demanding monotonically decreasing function values anyway is too strict as it does not allow for provably optimal schemes. We refer to a statement of Nesterov [35]: "In convex optimization the optimal methods never rely on relaxation. First, for some problem classes this property is too expensive. Second, the schemes and efficiency estimates of optimal methods are derived from some global topological properties of convex functions."[1] The negative side of better efficiency estimates of an algorithm is usually the convergence analysis. This is even true for convex functions. In case of nonconvex and nonsmooth functions, this problem becomes even more severe.

Contributions. Despite this problem, we can establish convergence of the sequence of function values for the general case, where the objective function is required only to be a composition of a convex and a differentiable function. Regarding the sequence of *arguments* generated by the algorithm, existence of a converging subsequence is shown. Furthermore, we show that each limit point is a critical point of the objective function.

[1] Relaxation is to be interpreted as the property of monotonically decreasing function values in this context. Topological properties should be associated with geometrical properties.

To establish convergence of the *whole* sequence in the nonconvex case is very hard. However, with slightly more assumptions to the objective, namely, that it satisfies the Kurdyka–Łojasiewicz inequality [30, 31, 26], several algorithms have been shown to converge [14, 5, 3, 4]. In [5] an abstract convergence theorem for descent methods with certain properties is proved. It applies to many algorithms. However, it cannot be used for our algorithm. Based on their analysis, we prove an abstract convergence theorem for a different class of descent methods, which applies to iPiano. By verifying the requirements of this abstract convergence theorem, we manage to also show such a strong convergence result. From a practical point of view of image processing, computer vision, or machine learning, the Kurdyka–Łojasiewicz inequality is almost always satisfied. For more details about properties of Kurdyka–Łojasiewicz functions and a taxonomy of functions that have this property, we refer the reader to [5, 10, 26].

The last part of the paper is devoted to experiments. We exemplarily present results on computer vision tasks, such as denoising and image compression, and show that entering the staggering world of nonconvex functions pays off in practice.

2. Related work.

Forward-backward splitting. In convex optimization, splitting algorithms usually originate from the proximal point algorithm [39]. It is a very general algorithm, and results on its convergence affect many other algorithms. Practically, however, computing one iteration of the algorithm can be as hard as the original problem. Among the strategies to tackle this problem are splitting approaches such as Douglas–Rachford [28, 18], several primal-dual algorithms [12, 37, 23], and forward-backward splitting [28, 16, 7, 35]; see [15] for a survey.

The forward-backward splitting schemes seem to be especially appealing to generalize to nonconvex problems. This is due to their simplicity and the existence of simpler formulations in some special cases such as, for example, the gradient projection method, where the backward step is the projection onto a set [27, 22]. In [19] the classical forward-backward algorithm, where the backward step is the solution of a proximal term involving a convex function, is studied for a nonconvex problem. In fact, the same class of objective functions as in the present paper is analyzed. The algorithm presented here comprises the algorithm from [19] as a special case. Also Nesterov [36] briefly discusses this algorithm in a general setting. Even the reverse setting is generalized in the nonconvex setting [5, 11], namely, where the backward step is performed on a nonsmooth, nonconvex function.

As the amount of data to be processed is growing and algorithms are supposed to exploit all the data in each iteration, inexact methods become interesting, though we do not consider erroneous estimates in this paper. Forward-backward splitting schemes also seem to work for nonconvex problems with erroneous estimates [44, 43]. A mathematical analysis of inexact methods can be found, e.g., in [14, 5], but with the restriction that the method is explicitly required to decrease the function values in each iteration. The restriction comes with significantly improved results with regard to the convergence of the algorithm. The algorithm proposed in this paper provides strong convergence results, although it does not require the function values to decrease.

Optimization with inertial forces. In his seminal work [38], Polyak investigates multistep schemes to accelerate the gradient method. It turns out that a particularly interesting case is given by a two-step algorithm, which has been coined the *Heavy-ball* method. The name of the method is due to the fact that it can be interpreted as an explicit finite difference discretization of the so-called *Heavy-ball with friction* dynamical system. It differs from the usual gradient method by adding an inertial term that is computed by the difference of the two preceding iterations. Polyak showed that this method can speed up convergence in comparison to the standard gradient method, while the cost of each iteration stays basically unchanged.

The popular accelerated gradient method of Nesterov [35] obviously shares some similarities with the Heavy-ball method, but it differs from it in one regard: while the Heavy-ball method uses gradients based on the current iterate, Nesterov's accelerated gradient method evaluates the gradient at points that are extrapolated by the inertial force. On strongly convex functions, both methods are equally fast (up to constants), but Nesterov's accelerated gradient method converges much faster on weakly convex functions [17].

The Heavy-ball method requires knowledge of the function parameters (the Lipschitz constant of the gradient and the modulus of strong convexity) to achieve the optimal convergence rate, which can be seen as a disadvantage. Interestingly, the conjugate gradient method for minimizing strictly convex quadratic problems can be expressed as the Heavy-ball method. Hence, it can be seen as a special case of the Heavy-ball method for quadratic problems. In this special case, no additional knowledge of the function parameters is required, as the algorithm parameters are computed online.

The Heavy-ball method was originally proposed for minimizing differentiable convex functions, but it has been generalized in different ways. In [45], it has been generalized to the case of smooth nonconvex functions. It is shown that, by considering an appropriate Lyapunov objective function, the iterations are attracted by the connected components of stationary points. In section 4 it will become evident that the nonconvex Heavy-ball method is a special case of our algorithm, and also the convergence analysis of [45] shows some similarities to ours.

In [2, 1], the Heavy-ball method has been extended to maximal monotone operators, e.g., the subdifferential of a convex function. In a subsequent work [34], it has been applied to a forward-backward splitting algorithm, again in the general framework of maximal monotone operators.

3. An abstract convergence result.

3.1. Preliminaries.

We consider the Euclidean vector space $\mathbb{R}^N$ of dimension $N \geq 1$ and denote the standard inner product by $\langle \cdot, \cdot \rangle$ and the induced norm by $\|\cdot\|_2^2 := \sqrt{\langle \cdot, \cdot \rangle}$. Let $F\colon \mathbb{R}^N \to \mathbb{R} \cup \{+\infty\}$ be a proper lower semicontinuous function.

Definition 3.1 (effective domain, proper). *The* (effective) domain *of F is defined by* $\operatorname{dom} F := \{x \in \mathbb{R}^N : F(x) < +\infty\}$. *The function is called* proper *if* $\operatorname{dom} F$ *is nonempty.*

In order to give a sound description of the first order optimality condition for a nonconvex, nonsmooth optimization problem, we have to introduce the generalization of the subdifferential for convex functions.

Definition 3.2 (limiting-subdifferential). *The* limiting-subdifferential *(or simply* subdifferential*) is defined by (see* [40, Def. 8.3]*)*

$$\partial F(x) = \{\xi \in \mathbb{R}^N \,|\, \exists y_k \to x,\ F(y_k) \to F(x),\ \xi_k \to \xi,\ \xi_k \in \widehat{\partial} F(y_k)\}\,, \tag{1}$$

which makes use of the Fréchet subdifferential *defined by*

$$\widehat{\partial} F(x) = \{\xi \in \mathbb{R}^N |\ \liminf_{\substack{y \to x \\ y \neq x}} \tfrac{1}{\|x-y\|_2} \left(F(y) - F(x) - \langle y - x, \xi \rangle\right) \geq 0\},$$

when $x \in \operatorname{dom} F$ *and by* $\widehat{\partial} F(x) = \varnothing$ *else.*

The *domain* of the subdifferential is $\operatorname{dom} \partial F := \{x \in \mathbb{R}^N |\ \partial F(x) \neq \varnothing\}$.

In what follows, we will consider the problem of finding a critical point $x^* \in \operatorname{dom} F$ of F, which is characterized by the necessary first-order optimality condition $0 \in \partial F(x^*)$.

We state the definition of the Kurdyka–Łojasiewicz property from [4].

Definition 3.3 (Kurdyka–Łojasiewicz property).

1. *The function* $F\colon \mathbb{R}^N \to \mathbb{R} \cup \{\infty\}$ *has the Kurdyka–Łojasiewicz property at* $x^* \in \operatorname{dom} \partial F$ *if there exist* $\eta \in (0, \infty]$, *a neighborhood* U *of* x^*, *and a continuous concave function* $\varphi\colon [0, \eta) \to \mathbb{R}_+$ *such that* $\varphi(0) = 0$, $\varphi \in C^1((0, \eta))$; *for all* $s \in (0, \eta)$ *it is* $\varphi'(s) > 0$, *and for all* $x \in U \cap [F(x^*) < F < F(x^*) + \eta]$ *the Kurdyka–Łojasiewicz inequality holds, i.e.,*
$$\varphi'(F(x) - F(x^*)) \operatorname{dist}(0, \partial F(x)) \geq 1.$$
2. *If the function* F *satisfies the Kurdyka–Łojasiewicz inequality at each point of* $\operatorname{dom} \partial F$, *it is called a KL function.*

Roughly speaking, this condition says that we can bound the subgradient of a function from below by a reparametrization of its function values. In the smooth case, we can also say that up to a reparametrization the function h is sharp, meaning that any nonzero gradient can be bounded away from 0. This is sometimes called a desingularization. It has been shown in [4] that a proper lower semicontinuous extended valued function h always satisfies this inequality at each nonstationary point. For more details and other interpretations of this property, and also for different formulations, we refer the reader to [10].

A big class of functions that have the KL property is given by real semialgebraic functions [4]. Real semialgebraic functions are defined as functions whose graph is a real semialgebraic set.

Definition 3.4 (real semialgebraic set). *A subset* S *of* $\mathbb{R}^N$ *is* semialgebraic *if there exists a finite number of real polynomials* $P_{i,j}, Q_{i,j}\colon \mathbb{R}^N \to \mathbb{R}$ *such that*

$$S = \bigcup_{j=1}^{p} \bigcap_{i=1}^{q} \{x \in \mathbb{R}^N : P_{i,j}(x) = 0 \text{ and } Q_{i,j} < 0\}.$$

3.2. Inexact descent convergence result for KL functions. In the following, we prove an abstract convergence result for a sequence $(z^n)_{n \in \mathbb{N}} := (x^n, x^{n-1})_{n \in \mathbb{N}}$ in $\mathbb{R}^{2N}$, $x^n \in \mathbb{R}^N$, $x^{-1} \in \mathbb{R}^N$, satisfying certain basic conditions, $\mathbb{N} := \{0, 1, 2, \ldots\}$. For convenience we use the abbreviation $\Delta_n := \|x^n - x^{n-1}\|_2$ for $n \in \mathbb{N}$. We fix two positive constants $a > 0$ and $b > 0$ and consider a proper lower semicontinuous function $F\colon \mathbb{R}^{2N} \to \mathbb{R} \cup \{\infty\}$. Then, the conditions we require for $(z^n)_{n \in \mathbb{N}}$ are as follows:

(H1) For each $n \in \mathbb{N}$, it holds that
$$F(z^{n+1}) + a\Delta_n^2 \leq F(z^n).$$

(H2) For each $n \in \mathbb{N}$, there exists $w^{n+1} \in \partial F(z^{n+1})$ such that

$$\|w^{n+1}\|_2 \le \frac{b}{2}(\Delta_n + \Delta_{n+1}) .$$

(H3) There exists a subsequence $(z^{n_j})_{j\in\mathbb{N}}$ such that

$$z^{n_j} \to \tilde{z} \quad \text{and} \quad F(z^{n_j}) \to F(\tilde{z}) \qquad \text{as } j \to \infty .$$

Based on these conditions, we derive the same convergence result as in [5]. The statements and proofs of the subsequent results follow the same ideas as those in [5]. We modified the involved calculations according to our conditions (H1), (H2), and (H3).

Remark 1. These conditions are very similar to those in [5]; however, they are not identical. The difference comes from the fact that [5] does not consider a two-step algorithm.

- In [5] the corresponding condition to (H1) (sufficient decrease condition) is $F(x^{n+1}) + a\Delta_{n+1}^2 \le F(x^n)$.
- The corresponding condition to (H2) (relative error condition) is $\|w^{n+1}\|_2 \le b\Delta_{n+1}$. In some sense, our condition (H2) accepts a larger relative error.
- (H3) (continuity condition) in [5] is the same here, but for $(x^{n_j})_{j\in\mathbb{N}}$.

Remark 2. Our proof and the proof in [5] differ mainly in the calculations that are involved; the outline is the same. There is hope of finding an even more general convergence result, which comprises ours and [5].

Lemma 3.5. *Let $F\colon \mathbb{R}^{2N} \to \mathbb{R} \cup \{\infty\}$ be a proper lower semicontinuous function which satisfies the Kurdyka–Lojasiewicz property at some point $z^* = (x^*, x^*) \in \mathbb{R}^{2N}$. Denote by U, η, and $\varphi\colon [0,\eta) \to \mathbb{R}_+$ the objects appearing in Definition 3.3 of the KL property at z^*. Let $\sigma, \rho > 0$ be such that $B(z^*,\sigma) \subset U$ with $\rho \in (0,\sigma)$, where $B(z^*,\sigma) := \{z \in \mathbb{R}^{2N} : \|z - z^*\|_2 < \sigma\}$.*

Furthermore, let $(z^n)_{n\in\mathbb{N}} = (x^n, x^{n-1})_{n\in\mathbb{N}}$ be a sequence satisfying conditions (H1), (H2), and

$$\forall n \in \mathbb{N}: \quad z^n \in B(z^*,\rho) \Rightarrow z^{n+1} \in B(z^*,\sigma) \text{ with } F(z^{n+1}), F(z^{n+2}) \ge F(z^*) . \tag{2}$$

Moreover, the initial point $z^0 = (x^0, x^{-1})$ is such that $F(z^*) \le F(z^0) < F(z^*) + \eta$ and

$$\|x^* - x^0\|_2 + \sqrt{\frac{F(z^0) - F(z^*)}{a}} + \frac{b}{a}\varphi(F(z^0) - F(z^*)) < \frac{\rho}{2} . \tag{3}$$

Then, the sequence $(z^n)_{n\in\mathbb{N}}$ satisfies

$$\forall n \in \mathbb{N} : z^n \in B(z^*,\rho), \quad \sum_{n=0}^{\infty} \Delta_n < \infty, \quad F(z^n) \to F(z^*) \text{ as } n \to \infty ; \tag{4}$$

$(z^n)_{n\in\mathbb{N}}$ converges to a point $\bar{z} = (\bar{x}, \bar{x}) \in B(z^*,\sigma)$ such that $F(\bar{z}) \le F(z^*)$. If, additionally, condition (H3) is satisfied, then $0 \in \partial F(\bar{z})$ and $F(\bar{z}) = F(z^*)$.

Proof. The key points of the proof are the facts that for all $j \ge 1$,

$$z^j \in B(z^*,\rho) \qquad \text{and} \tag{5}$$

$$\sum_{i=1}^{j} \Delta_i \le \frac{1}{2}(\Delta_0 - \Delta_j) + \frac{b}{a}[\varphi(F(z^1) - F(z^*)) - \varphi(F(z^{j+1}) - F(z^*))]. \tag{6}$$

Let us first see that $\varphi(F(z^{j+1}) - F(z^*))$ is well defined. By condition (H1), $(F(z^n))_{n\in\mathbb{N}}$ is nonincreasing, which shows that $F(z^{n+1}) \le F(z^0) < F(z^*) + \eta$. Combining this with (2) implies $F(z^{n+1}) - F(z^*) \ge 0$.

As for $n \ge 1$, the set $\partial F(z^n)$ is nonempty (see condition (H2)); every z^n belongs to $\operatorname{dom} F$. For notational convenience, we define

$$D_n^\varphi := \varphi(F(z^n) - F(z^*)) - \varphi(F(z^{n+1}) - F(z^*))\,.$$

Now, we want to show that for $n \ge 1$ the following holds: if $F(z^n) < F(z^*) + \eta$ and $z^n \in B(z^*, \rho)$, then

$$2\Delta_n \le \tfrac{b}{a} D_n^\varphi + \tfrac{1}{2}(\Delta_n + \Delta_{n-1})\,. \tag{7}$$

Obviously, we can assume that $\Delta_n \ne 0$ (otherwise it is trivial), and therefore (H1) and (2) imply $F(z^n) > F(z^{n+1}) \ge F(z^*)$. The KL inequality shows $w^n \ne 0$, and (H2) shows $\Delta_n + \Delta_{n-1} > 0$. Since $w^n \in \partial F(z^n)$, using the KL inequality and (H2), we obtain

$$\varphi'(F(z^n) - F(z^*)) \ge \frac{1}{\|w^n\|_2} \ge \frac{2}{b(\Delta_{n-1} + \Delta_n)}\,.$$

As φ is concave and increasing ($\varphi' > 0$), condition (H1) and (2) yield

$$D_n^\varphi \ge \varphi'(F(z^n) - F(z^*))(F(z^n) - F(z^{n+1})) \ge \varphi'(F(z^n) - F(z^*)) a\Delta_n^2\,.$$

Combining both inequalities results in

$$(\tfrac{b}{a} D_n^\varphi)\tfrac{1}{2}(\Delta_{n-1} + \Delta_n) \ge \Delta_n^2\,,$$

which by applying $2\sqrt{uv} \le u + v$ establishes (7).

As (2) only implies $z^{n+1} \in B(z^*, \sigma)$, $\sigma > \rho$, we cannot use (7) directly for the whole sequence. However, (5) and (6) can be shown by induction on j. For $j = 0$, (2) yields $z^1 \in B(z^*, \sigma)$ and $F(z^1), F(z^2) \ge F(z^*)$. From condition (H1) with $n = 1$, $F(z^2) \ge F(z^*)$, and $F(z^1) \le F(z^0)$, we infer

$$\Delta_1 \le \sqrt{\frac{F(z^1) - F(z^2)}{a}} \le \sqrt{\frac{F(z^0) - F(z^*)}{a}}\,, \tag{8}$$

which combined with (3) leads to

$$\|x^* - x^1\|_2 \le \|x^0 - x^*\|_2 + \Delta_1 \le \|x^0 - x^*\|_2 + \sqrt{\frac{F(z^0) - F(z^*)}{a}} < \frac{\rho}{2}\,,$$

and therefore $z^1 \in B(z^*, \rho)$. Direct use of (7) with $n = 1$ shows that (6) holds with $j = 1$.

Suppose (5) and (6) are satisfied for $j \ge 1$. Then, using the triangle inequality and (6), we have

$$\begin{aligned}
\|z^* - z^{j+1}\|_2 &\le \|x^* - x^{j+1}\|_2 + \|x^* - x^j\|_2 \\
&\le 2\|x^* - x^0\|_2 + 2\textstyle\sum_{i=1}^j \Delta_i + \Delta_{j+1} \\
&\le 2\|x^* - x^0\|_2 + (\Delta_0 - \Delta_j) + \Delta_{j+1} \\
&\quad 2\tfrac{b}{a}[\varphi(F(z^1) - F(z^*)) - \varphi(F(z^{j+1}) - F(z^*))] \\
&\le 2\|x^* - x^0\|_2 + \Delta_0 + \Delta_{j+1} + 2\tfrac{b}{a}[\varphi(F(z^0) - F(z^*))]\,,
\end{aligned}$$

which shows, using $\Delta_{j+1} \leq \sqrt{\frac{1}{a}(F(z^{j+1}) - F(z^{j+2}))} \leq \sqrt{\frac{1}{a}(F(z^0) - F(z^*))}$ and (3), that $z^{j+1} \in B(z^*, \rho)$. As a consequence, (7), with $n = j+1$, can be added to (6), and we can conclude (6) with $j+1$. This shows the desired induction on j.

Now, the finiteness of the length of the sequence $(x^n)_{n\in\mathbb{N}}$, i.e., $\sum_{i=1}^{\infty} \Delta_i < \infty$, is a consequence of the following estimation, which is implied by (6):

$$\sum_{i=1}^{j} \Delta_i \leq \tfrac{1}{2}\Delta_0 + \tfrac{b}{a}\varphi(F(z^1) - F(z^*)) < \infty \,.$$

Therefore, x^n converges to some $\bar{x}$ as $n \to \infty$, and z^n converges to $\bar{z} = (\bar{x}, \bar{x})$. As φ is concave, φ' is decreasing. Using this and condition (H2) yields $w^n \to 0$ and $F(z^n) \to \zeta \geq F(z^*)$. Suppose we have $\zeta > F(z^*)$; then the KL inequality reads $\varphi'(\zeta - F(z^*))\|w^n\|_2 \geq 1$ for all $n \geq 1$, which contradicts $w^n \to 0$.

Note that, in general, $\bar{z}$ is not a critical point of F, because the limiting subdifferential requires $F(z^n) \to F(\bar{z})$ as $n \to \infty$. When the sequence $(z^n)_{n\in\mathbb{N}}$ additionally satisfies condition (H3), then $\tilde{z} = \bar{z}$, and $\bar{z}$ is a critical point of F, because $F(\bar{z}) = \lim_{n\to\infty} F(z^n) = F(z^*)$. ■

Remark 3. The only difference from [5] with respect to the assumptions is (2). In [5], $z^n \in B(z^*, \rho)$ implies $F(z^{n+1}) \geq F(z^*)$, whereas we require $F(z^{n+1}) \geq F(z^*)$ and $F(z^{n+2}) \geq F(z^*)$. However, as Theorem 3.7 shows, this does not weaken the convergence result compared to [5]. In fact, Corollary 3.6, which assumes $F(z^n) \geq F(z^*)$ for all $n \in \mathbb{N}$ and which is also used in [5], is key in Theorem 3.7.

The next corollary and the subsequent theorem follow as in [5] by replacing the calculation with our conditions.

Corollary 3.6. *Lemma 3.5 holds true if we replace (2) by*

$$\eta < a(\sigma - \rho)^2 \qquad \textit{and} \qquad F(z^n) \geq F(z^*) \quad \forall n \in \mathbb{N} \,.$$

Proof. By condition (H1), for $z^n \in B(z^*, \rho)$, we have

$$\Delta_{n+1}^2 \leq \frac{F(z^{n+1}) - F(z^{n+2})}{a} \leq \frac{\eta}{a} < (\sigma - \rho)^2 \,.$$

Using the triangle inequality on $\|z^{n+1} - z^*\|_2$ shows that $z^{n+1} \in B(z^*, \sigma)$, which implies (2) and concludes the proof. ■

The work that is done in Lemma 3.5 and Corollary 3.6 allows us to formulate an abstract convergence theorem for sequences satisfying conditions (H1), (H2), and (H3). It follows, with a few modifications, as in [5].

Theorem 3.7 (convergence to a critical point). *Let $F\colon \mathbb{R}^{2N} \to \mathbb{R} \cup \{\infty\}$ be a proper lower semicontinuous function and $(z^n)_{n\in\mathbb{N}} = (x^n, x^{n-1})_{n\in\mathbb{N}}$ be a sequence that satisfies* (H1), (H2), *and* (H3). *Moreover, let F have the Kurdyka–Łojasiewicz property at the cluster point $\tilde{x}$ specified in* (H3).

Then, the sequence $(x^n)_{n=0}^{\infty}$ has finite length, i.e., $\sum_{n=1}^{\infty} \Delta_n < \infty$, and converges to $\bar{x} = \tilde{x}$ as $n \to \infty$, where $(\bar{x}, \bar{x})$ is a critical point of F.

Proof. By condition (H3), we have $z^{n_j} \to \bar{z} = \tilde{z}$ and $F(z^{n_j}) \to F(\bar{z})$ for a subsequence $(z^{n_j})_{n\in\mathbb{N}}$. This together with the nondecreasingness of $(F(z^n))_{n\in\mathbb{N}}$ (by condition (H1)) imply that $F(z^n) \to F(\bar{z})$ and $F(z^n) \geq F(\bar{z})$ for all $n \in \mathbb{N}$. The KL property around $\bar{z}$ states the existence of quantities φ, U, and η as in Definition 3.3. Let $\sigma > 0$ be such that $B(\bar{z}, \sigma) \subset U$ and $\rho \in (0, \sigma)$. Shrink η such that $\eta < a(\sigma - \rho)^2$ (if necessary). As φ is continuous, there exists $n_0 \in \mathbb{N}$ such that $F(z^n) \in [F(\bar{z}), F(\bar{z}) + \eta)$ for all $n \geq n_0$ and

$$\|x^* - x^{n_0}\|_2 + \sqrt{\frac{F(z^{n_0}) - F(z^*)}{a}} + \frac{b}{a}\varphi(F(z^{n_0}) - F(z^*)) < \frac{\rho}{2}.$$

Then, the sequence $(y^n)_{n\in\mathbb{N}}$ defined by $y^n = z^{n_0+n}$ satisfies the conditions in Corollary 3.6, which concludes the proof. ■

4. The proposed algorithm: iPiano.

4.1. The optimization problem. We consider a structured nonsmooth, nonconvex optimization problem with a proper lower semicontinuous extended valued function $h\colon \mathbb{R}^N \to \mathbb{R} \cup \{+\infty\}$, $N \geq 1$:

$$\min_{x\in\mathbb{R}^N} h(x) = \min_{x\in\mathbb{R}^N} f(x) + g(x)\,, \tag{9}$$

which is composed of a C^1-smooth (possibly nonconvex) function $f\colon \mathbb{R}^N \to \mathbb{R}$ with L-Lipschitz continuous gradient on $\operatorname{dom} g$, $L > 0$, and a convex (possibly nonsmooth) function $g\colon \mathbb{R}^N \to \mathbb{R} \cup \{+\infty\}$. Furthermore, we require h to be coercive, i.e., $\|x\|_2 \to +\infty$ implies $h(x) \to +\infty$, and bounded from below by some value $\underline{h} > -\infty$.

The proposed algorithm, which is stated in subsection 4.3, seeks a critical point $x^* \in \operatorname{dom} h$ of h, which is characterized by the necessary first-order optimality condition $0 \in \partial h(x^*)$. In our case, this is equivalent to

$$-\nabla f(x^*) \in \partial g(x^*)\,.$$

This equivalence is explicitly verified in the next subsection, where we collect some details and state some basic properties which are used in the convergence analysis in subsection 4.5.

4.2. Preliminaries. Consider the function f first. It is required to be C^1-smooth with L-Lipschitz continuous gradient on $\operatorname{dom} g$; i.e., there exists a constant $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2 \quad \forall x, y \in \operatorname{dom} g\,. \tag{10}$$

This directly implies that $\operatorname{dom} h = \operatorname{dom} g$ is a nonempty convex set, as $\operatorname{dom} g \subset \operatorname{dom} f$. This property of f plays a crucial role in our convergence analysis due to the following lemma (stated as in [5]).

Lemma 4.1 (descent lemma). *Let $f\colon \mathbb{R}^N \to \mathbb{R}$ be a C^1-function with L-Lipschitz continuous gradient ∇f on $\operatorname{dom} g$. Then for any $x, y \in \operatorname{dom} g$ it holds that*

$$f(x) \leq f(y) + \langle \nabla f(y), x - y\rangle + \frac{L}{2}\|x - y\|_2^2\,. \tag{11}$$

Proof. See, for example, [35]. ∎

We assume that the function g is a proper lower semicontinuous convex function with an efficient to compute proximal map.

Definition 4.2 (proximal map). *Let g be a proper lower semicontinuous convex function. Then, we define the* proximal map

$$(I + \alpha\partial g)^{-1}(\hat{x}) := \arg\min_{x\in\mathbb{R}^N} \frac{\|x-\hat{x}\|_2^2}{2} + \alpha g(x)\,,$$

where $\alpha > 0$ is a given parameter, I is the identity map, and $\hat{x} \in \mathbb{R}^N$.

An important (basic) property that the convex function g contributes to the convergence analysis is the following.

Lemma 4.3. *Let g be a proper lower semicontinuous convex function; then it holds for any $x, y \in \operatorname{dom} g$, $s \in \partial g(x)$ that*

$$g(y) \geq g(x) + \langle s, y - x\rangle\,. \tag{12}$$

Proof. This result follows directly from the convexity of g. ∎

Finally, consider the optimality condition $0 \in \partial h(x^*)$ in more detail. The following proposition proves the equivalence to $-\nabla f(x^*) \in \partial g(x^*)$. The proof is mainly based on Definition 3.2 of the limiting subdifferential.

Proposition 4.4. *Let h, f, and g be as before; i.e., let $h = f + g$ with f continuously differentiable and g convex. Sometimes, h is then called a C^1-perturbation of a convex function. Then, for $x \in \operatorname{dom} h$ holds*

$$\partial h(x) = \nabla f(x) + \partial g(x)\,.$$

Proof. We first prove "$\subset$". Let $\xi^h \in \partial h(x)$; i.e., there is a sequence $(y_k)_{k=0}^\infty$ such that $y_k \to x$, $h(y_k) \to h(x)$, and $\xi_k^h \to \xi^h$, where $\xi_k^h \in \widehat{\partial} h(y_k)$. We want to show that $\xi^g := \xi^h - \nabla f(x) \in \partial g(x)$. As $f \in C^1$ and $\xi^h \in \partial h(x)$, we have

$$\begin{aligned} y_k &\overset{k\to\infty}{\longrightarrow} x, \\ g(y_k) = h(y_k) - f(y_k) &\overset{k\to\infty}{\longrightarrow} h(x) - f(x) = g(x), \\ \xi_k^g := \xi_k^h - \nabla f(y_k) &\overset{k\to\infty}{\longrightarrow} \xi^h - \nabla f(x) =: \xi^g\,. \end{aligned}$$

It remains to show that $\xi_k^g \in \widehat{\partial} g(y_k)$. First, remember that lim inf is superadditive; i.e., for two sequences $(a_n)_{n=0}^\infty$, $(b_n)_{n=0}^\infty$ in $\mathbb{R}$ it is $\liminf_{n\to\infty}(a_n+b_n) \geq \liminf_{n\to\infty} a_n + \liminf_{n\to\infty} b_n$. However, convergence of a_n implies $\liminf_{n\to\infty}(a_n+b_n) = \lim_{n\to\infty} a_n + \liminf_{n\to\infty} b_n$. This fact together with $f \in C^1$ allows us to conclude

$$\begin{aligned} 0 &\leq \liminf \left(h(y_k') - h(y_k) - \left\langle y_k' - y_k, \xi_k^h\right\rangle\right) / \|y_k' - y_k\|_2 \\ &\leq \liminf \left(f(y_k') - f(y_k) + g(y_k') - g(y_k) - \left\langle y_k' - y_k, \nabla f(y_k) + \xi_k^g\right\rangle\right) / \|y_k' - y_k\|_2 \\ &= \lim \left(f(y_k') - f(y_k) - \langle y_k' - y_k, \nabla f(y_k)\rangle\right) / \|y_k' - y_k\|_2 \\ &\quad + \liminf \left(g(y_k') - g(y_k) - \left\langle y_k' - y_k, \xi_k^g\right\rangle\right) / \|y_k' - y_k\|_2 \\ &= \liminf \left(g(y_k') - g(y_k) - \left\langle y_k' - y_k, \xi_k^g\right\rangle\right) / \|y_k' - y_k\|_2\,, \end{aligned}$$

where lim inf and lim are over $y'_k \to y_k, y'_k \neq y_k$. Therefore, $\xi_k^g \in \widehat{\partial} g(y_k)$. The other inclusion "$\supset$" is trivial. ∎

As a consequence, a critical point can also be characterized by the following definition.

Definition 4.5 (proximal residual). *Let f and g be as before. Then, we define the* proximal residual

$$r(x) := x - (I + \partial g)^{-1}(x - \nabla f(x)) .$$

It can be easily seen that $r(x) = 0$ is equivalent to $x = (I + \partial g)^{-1}(x - \nabla f(x))$ and $(I + \partial g)(x) = (I - \nabla f)(x)$, which is the first-order optimality condition. The proximal residual is defined with respect to a fixed step size of 1. The rationale behind this becomes obvious when g is the indicator function of a convex set. In this case, a small residual could be caused by small step sizes as the reprojection onto the convex set is independent of the step size.

4.3. The generic algorithm. In this paper, we propose an algorithm, iPiano, with the generic formulation in Algorithm 1. It is a forward-backward splitting algorithm incorporating an inertial force. In the forward step, α_n determines the step size in the direction of the gradient of the differentiable function f. The step in gradient direction is aggregated with the inertial force from the previous iteration weighted by β_n. Then, the backward step is the solution of the proximity operator for the function g with the weight α_n.

Algorithm 1. *Inertial proximal algorithm for nonconvex optimization (iPiano)*
- *Initialization: Choose a starting point $x^0 \in \operatorname{dom} h$ and set $x^{-1} = x^0$. Moreover, define sequences of step size parameter $(\alpha_n)_{n=0}^\infty$ and $(\beta_n)_{n=0}^\infty$.*
- *Iterations $(n \geq 0)$: Update*

$$x^{n+1} = (I + \alpha_n \partial g)^{-1}(x^n - \alpha_n \nabla f(x^n) + \beta_n(x^n - x^{n-1})) . \tag{13}$$

In order to make the algorithm specific and convergent, the step size parameters must be chosen appropriately. What "appropriately" means will be specified in subsection 4.4 and proved in subsection 4.5.

4.4. Rules for choosing the step size. In this subsection, we propose several strategies for choosing the step sizes. This will make it easier to implement the algorithm. One may choose among the following variants of step size rules depending on the knowledge about the objective function.

Constant step size scheme. The most simple strategy, which requires the most knowledge about the objective function, is outlined in Algorithm 2. All step size parameters are chosen a priori and are constant.

Remark 4. Observe that our law on α, β is equivalent to the law found in [45] for minimizing a smooth nonconvex function. Hence, our result can be seen as an extension of their work to the presence of an additional nonsmooth convex function.

Backtracking. The case where we have only limited knowledge about the objective function occurs more frequently. It can be very challenging to estimate the Lipschitz constant of ∇f beforehand. Using backtracking the Lipschitz constant can be estimated automatically. A

Algorithm 2. *Inertial proximal algorithm for nonconvex optimization with constant parameter (ciPiano)*

- *Initialization: Choose* $\beta \in [0, 1)$*, set* $\alpha < 2(1-\beta)/L$*, where* L *is the Lipschitz constant of* ∇f*, choose ,* $x^0 \in \operatorname{dom} h$*, and set* $x^{-1} = x^0$*.*
- *Iterations* ($n \geq 0$)*: Update* x^n *as follows:*

$$x^{n+1} = (I + \alpha\partial g)^{-1}(x^n - \alpha\nabla f(x^n) + \beta(x^n - x^{n-1})) \tag{14}$$

sufficient condition that the Lipschitz constant at iteration n to $n+1$ must satisfy is

$$f(x^{n+1}) \leq f(x^n) + \left\langle \nabla f(x^n), x^{n+1} - x^n \right\rangle + \frac{L_n}{2}\|x^{n+1} - x^n\|_2^2 . \tag{15}$$

Although there are different strategies for determining L_n, the most common is to define an increment variable $\eta > 1$ and to look for the minimal $L_n \in \{L_{n-1}, \eta L_{n-1}, \eta^2 L_{n-1}, \ldots\}$ satisfying (15). Sometimes it is also feasible to decrease the estimated Lipschitz constant after a few iterations. A possible strategy is as follows: if $L_n = L_{n-1}$, then search for the minimal $L_n \in \{\eta^{-1}L_{n-1}, \eta^{-2}L_{n-1}, \ldots\}$ satisfying (15).

In Algorithm 3 we propose an algorithm with variable step sizes. Any strategy for estimating the Lipschitz constant may be used. When changing the Lipschitz constant from one iteration to another, all step size parameters must be adapted. The rules for adapting the step sizes will be justified during the convergence analysis in subsection 4.5.

Algorithm 3. *Inertial proximal algorithm for nonconvex optimization with backtracking (biPiano)*

- *Initialization: Choose* $\delta \geq c_2 > 0$ *with* c_2 *close to* 0 *(e.g.,* $c_2 := 10^{-6}$*) and* $x^0 \in \operatorname{dom} h$*, and set* $x^{-1} = x^0$*.*
- *Iterations* ($n \geq 0$)*: Update* x^n *as follows:*

$$x^{n+1} = (I + \alpha_n\partial g)^{-1}(x^n - \alpha_n\nabla f(x^n) + \beta_n(x^n - x^{n-1}))\,, \tag{16}$$

where $L_n > 0$ *satisfies* (15) *and*

$$\beta_n = (b-1)\Big/\Big(b - \frac{1}{2}\Big), \qquad b := \Big(\delta + \frac{L_n}{2}\Big)\Big/\Big(c_2 + \frac{L_n}{2}\Big),$$
$$\alpha_n = 2(1-\beta_n)/(2c_2 + L_n)\,.$$

Lazy backtracking. Algorithm 4 presents another alternative to Algorithm 1. It is related to Algorithms 2 and 3 in the following way. Algorithm 4 makes use of the Lipschitz continuity of ∇f in the sense that the Lipschitz constant is always finite. As a consequence, using backtracking with only increasing Lipschitz constants, after a finite number of iterations $n_0 \in \mathbb{N}$ the estimated Lipschitz constant will no longer change, and starting from this iteration the constant step size rules as in Algorithm 2 are applied. Using this strategy, the results that will be proved in the convergence analysis are satisfied only as soon as the Lipschitz constant is high enough and no longer changing.

Algorithm 4. *Nonmonotone inertial proximal algorithm for nonconvex optimization with backtracking (nmiPiano)*

- *Initialization: Choose* $\beta \in [0,1)$, $L_{-1} > 0$, $\eta > 1$, *and* $x^0 \in \operatorname{dom} h$, *and set* $x^{-1} = x^0$.
- *Iterations* ($n \geq 0$): *Update* x^n *as follows:*

$$x^{n+1} = (I + \alpha_n \partial g)^{-1}(x^n - \alpha_n \nabla f(x^n) + \beta(x^n - x^{n-1})), \tag{17}$$

where $L_n \in \{L_{n-1}, \eta L_{n-1}, \eta^2 L_{n-1}, \dots\}$ *is minimal and satisfies*

$$f(x^{n+1}) \leq f(x^n) + \langle \nabla f(x^n), x^{n+1} - x^n \rangle + \frac{L_n}{2}\|x^{n+1} - x^n\|_2^2 \tag{18}$$

and $\alpha_n < 2(1-\beta)/L_n$.

General rule of choosing the step sizes. Algorithm 5 defines the general rules that the step size parameters must satisfy.

Algorithm 5. *Inertial proximal algorithm for nonconvex optimization (iPiano)*

- *Initialization: Choose* $c_1, c_2 > 0$ *close to* 0, $x^0 \in \operatorname{dom} h$, *and set* $x^{-1} = x^0$.
- *Iterations* ($n \geq 0$): *Update*

$$x^{n+1} = (I + \alpha_n \partial g)^{-1}(x^n - \alpha_n \nabla f(x^n) + \beta_n(x^n - x^{n-1})), \tag{19}$$

where $L_n > 0$ *is the local Lipschitz constant satisfying*

$$f(x^{n+1}) \leq f(x^n) + \langle \nabla f(x^n), x^{n+1} - x^n \rangle + \frac{L_n}{2}\|x^{n+1} - x^n\|_2^2, \tag{20}$$

and $\alpha_n \geq c_1$, $\beta_n \geq 0$ *are chosen such that* $\delta_n \geq \gamma_n \geq c_2$ *is defined by*

$$\delta_n := \frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{2\alpha_n} \quad \text{and} \quad \gamma_n := \frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{\alpha_n} \tag{21}$$

and $(\delta_n)_{n=0}^{\infty}$ *is monotonically decreasing.*

It contains Algorithms 2, 3, and 4 as special instances. This is easily verified for Algorithms 2 and 4. For Algorithm 3 the step size rules are derived from the proof of Lemma 4.6.

As Algorithm 5 is the most general algorithm, let us now analyze its behavior.

4.5. Convergence analysis. In all of what follows, let $(x^n)_{n=0}^{\infty}$ be the sequence generated by Algorithm 5 and with parameters satisfying the algorithm's requirements. Furthermore, for more convenient notation we abbreviate $H_\delta(x,y) := h(x) + \delta\|x-y\|_2^2$, $\delta \in \mathbb{R}$, and $\Delta_n := \|x^n - x^{n-1}\|_2$. Note that for $x = y$ it is $H_\delta(x,y) = h(x)$.

Let us first verify that the algorithm makes sense. We have to show that the requirements for the parameters are not contradictory, i.e., that it is possible to choose a feasible set of parameters. In the following lemma, we will only show the existence of such a parameter set;

however, the proof helps us to formulate specific step size rules.

Lemma 4.6. *For all $n \geq 0$, there are $\delta_n \geq \gamma_n$, $\beta_n \in [0,1)$, and $\alpha_n < 2(1-\beta_n)/L_n$. Furthermore, given $L_n > 0$, there exists a choice of parameters α_n and β_n such that additionally $(\delta_n)_{n=0}^\infty$ is monotonically decreasing.*

Proof. By the algorithm's requirements we have

$$\delta_n = \frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{2\alpha_n} \geq \frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{\alpha_n} = \gamma_n > 0\,.$$

The upper bound for β_n and α_n comes from rearranging $\gamma_n \geq c_2$ to $\beta_n \leq 1 - \alpha_n L_n/2 - c_2\alpha_n$ and $\alpha_n \leq 2(1-\beta_n)/(L_n + 2c_2)$, respectively.

The last statement follows by incorporating the descent property of δ_n. Let $\delta_{-1} \geq c_2$ be chosen initially. Then, the descent property of $(\delta_n)_{n=0}^\infty$ requires one of the equivalent statements

$$\delta_{n-1} \geq \delta_n \quad \Leftrightarrow \quad \delta_{n-1} \geq \frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{2\alpha_n} \quad \Leftrightarrow \quad \alpha_n \geq \frac{1 - \frac{\beta_n}{2}}{\delta_{n-1} + \frac{L_n}{2}}$$

to be true. An upper bound on α_n is obtained by

$$\gamma_n \geq c_2 \quad \Leftrightarrow \quad \alpha_n \leq \frac{1-\beta_n}{c_2 + \frac{L_n}{2}}\,.$$

The only thing that remains to show is that there exist $\alpha_n > c_1$ and $\beta_n \in [0,1)$ such that these two relations are fulfilled. Consider the condition for a nonnegative gap between the upper and lower bounds for α_n:

$$\frac{1-\beta_n}{c_2 + \frac{L_n}{2}} - \frac{1 - \frac{\beta_n}{2}}{\delta_{n-1} + \frac{L_n}{2}} \geq 0 \quad \Leftrightarrow \quad \frac{\delta_{n-1} + \frac{L_n}{2}}{c_2 + \frac{L_n}{2}} \geq \frac{1 - \frac{\beta_n}{2}}{1-\beta_n}\,.$$

Defining $b := (\delta_{n-1} + \frac{L_n}{2})/(c_2 + \frac{L_n}{2}) \geq 1$, it is easily verified that there exists $\beta_n \in [0,1)$ satisfying the equivalent condition

$$\frac{b-1}{b-\frac{1}{2}} \geq \beta_n\,. \tag{22}$$

As a consequence, the existence of a feasible α_n follows, and the descent property for δ_n holds. ■

In the following proposition, we state a result which will be very useful. Although iPiano does not imply a descent property of the function values, we construct a majorizing function that enjoys a monotonic descent property. This function reveals the connection to the Lyapunov direct method for convergence analysis as used in [45].

Proposition 4.7.

(a) *The sequence $(H_{\delta_n}(x^n, x^{n-1}))_{n=0}^\infty$ is monotonically decreasing and thus converging. In particular, it holds that*

$$H_{\delta_{n+1}}(x^{n+1}, x^n) \leq H_{\delta_n}(x^n, x^{n-1}) - \gamma_n \Delta_n^2\,. \tag{23}$$

(b) *It holds that* $\sum_{n=0}^{\infty} \Delta_n^2 < \infty$ *and, thus,* $\lim_{n\to\infty} \Delta_n = 0$.

Proof.

(a) From (19) it follows that

$$\frac{x^n - x^{n+1}}{\alpha_n} - \nabla f(x^n) + \frac{\beta_n}{\alpha_n}(x^n - x^{n-1}) \in \partial g(x^{n+1}).$$

Now using $x = x^{n+1}$ and $y = x^n$ in (11) and (12) and summing both inequalities it follows that

$$\begin{aligned} h(x^{n+1}) &\le h(x^n) - \left(\frac{1}{\alpha_n} - \frac{L_n}{2}\right)\Delta_{n+1}^2 + \frac{\beta_n}{\alpha_n}\left\langle x^{n+1} - x^n, x^n - x^{n-1}\right\rangle \\ &\le h(x^n) - \left(\frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{2\alpha_n}\right)\Delta_{n+1}^2 + \frac{\beta_n}{2\alpha_n}\Delta_n^2 , \end{aligned}$$

where the second line follows from $2\langle a, b\rangle \le \|a\|_2^2 + \|b\|_2^2$ for vectors $a, b \in \mathbb{R}^N$. Then, a simple rearrangement of the terms shows

$$h(x^{n+1}) + \delta_n \Delta_{n+1}^2 \le h(x^n) + \delta_n \Delta_n^2 - \gamma_n \Delta_n^2 ,$$

which establishes (23) as δ_n is monotonically decreasing. Obviously, the sequence $(H_{\delta_n}(x^n, x^{n-1}))_{n=0}^{\infty}$ is monotonically decreasing if and only if $\gamma_n \ge 0$, which is true by the algorithm's requirements. By assumption, h is bounded from below by some constant $\underline{h} > -\infty$; hence $(H_{\delta_n}(x^n, x^{n-1}))_{n=0}^{\infty}$ converges.

(b) Summing up (23) from $n = 0, \dots, N$ yields (note that $H_{\delta_n}(x^0, x^{-1}) = h(x^0)$)

$$\begin{aligned} \sum_{n=0}^{N} \gamma_n \Delta_n^2 &\le \sum_{n=0}^{N} H_{\delta_n}(x^n, x^{n-1}) - H_{\delta_{n+1}}(x^{n+1}, x^n) \\ &= h(x^0) - H_{\delta_{N+1}}(x^{N+1}, x^N) \le h(x^0) - \underline{h} < \infty . \end{aligned}$$

Letting N tend to ∞ and remembering that $\gamma_N \ge c_2 > 0$ holds implies the statement. ■

Remark 5. The function H_δ is a Lyapunov function for the dynamical system described by the Heavy-ball method. It corresponds to a discretized version of the kinetic energy of the Heavy-ball with friction.

In the following theorem, we state our general convergence results about Algorithm 5.

Theorem 4.8.

(a) *The sequence* $(h(x^n))_{n=0}^{\infty}$ *converges.*

(b) *There exists a converging subsequence* $(x^{n_k})_{k=0}^{\infty}$.

(c) *Any limit point* $x^* := \lim_{k\to\infty} x^{n_k}$ *is a critical point of* (9) *and* $h(x^{n_k}) \to h(x^*)$ *as* $k \to \infty$.

Proof.

(a) This follows from the Squeeze theorem as for all $n \ge 0$ it holds that

$$H_{-\delta_n}(x^n, x^{n-1}) \le h(x^n) \le H_{\delta_n}(x^n, x^{n-1}),$$

and thanks to Proposition 4.7(4.7) and (4.7) it holds that

$$\lim_{n\to\infty} H_{-\delta_n}(x^n, x^{n-1}) = \lim_{n\to\infty} H_{\delta_n}(x^n, x^{n-1}) - 2\delta_n \Delta_n^2 = \lim_{n\to\infty} H_{\delta_n}(x^n, x^{n-1}) .$$

(b) By Proposition 4.7(4.7) and $H_{\delta_0}(x^0, x^{-1}) = h(x^0)$ it is clear that the whole sequence $(x^n)_{n=0}^\infty$ is contained in the level set $\{x \in \mathbb{R}^N : \underline{h} \le h(x) \le h(x^0)\}$, which is bounded thanks to the coercivity of h and $\underline{h} = \inf_{x\in\mathbb{R}^N} h(x) > -\infty$. Using the Bolzano–Weierstrass theorem, we deduce the existence of a converging subsequence $(x^{n_k})_{k=0}^\infty$.

(c) To show that each limit point $x^* := \lim_{j\to\infty} x^{n_j}$ is a critical point of (9), recall that the subdifferential (1) is closed [40]. Define

$$\xi^j := \frac{x^{n_j} - x^{n_j+1}}{\alpha_{n_j}} - \nabla f(x^{n_j}) + \frac{\beta_{n_j}}{\alpha_{n_j}}(x^{n_j} - x^{n_j-1}) + \nabla f(x^{n_j+1}).$$

Then, the sequence $(x^{n_j}, \xi^j) \in \operatorname{Graph}(\partial h) := \{(x,\xi) \in \mathbb{R}^N \times \mathbb{R}^N \,|\, \xi \in \partial h(x)\}$. Furthermore, it holds that $x^* = \lim_{j\to\infty} x^{n_j}$, and due to Proposition 4.7(4.7), the Lipschitz continuity of ∇f, and

$$\|\xi^j - 0\|_2 \le \frac{1}{\alpha_{n_j}}\Delta_{n_j+1} + \frac{\beta_{n_j}}{\alpha_{n_j}}\Delta_{n_j} + \|\nabla f(x^{n_j+1}) - \nabla f(x^{n_j})\|_2$$

it holds that $\lim_{j\to\infty} \xi^j = 0$. It remains to show that $\lim_{j\to\infty} h(x^{n_j}) = h(x^*)$. By the closure property of the subdifferential ∂h we have $(x^*, 0) \in \operatorname{Graph}(\partial h)$, which means that x^* is a critical point of h.

The continuity statement about the limiting process as $j \to \infty$ follows by the lower semicontinuity of g, the existence $\lim_{j\to\infty} \xi^j = 0$, and the convexity property in Lemma 4.3

$$\limsup_{j\to\infty} g(x^{n_j}) = \limsup_{j\to\infty} g(x^{n_j}) + \left\langle \xi^j, x^* - x^{n_j} \right\rangle \le g(x^*) \le \liminf_{j\to\infty} g(x^{n_j}).$$

The first equality holds because the subadditivity of lim sup becomes an equality when the limit exists for one of the two summed sequences[2]; here $\lim_{j\to\infty} \left\langle \xi^j, x^* - x^{n_j} \right\rangle = 0$ exists. Moreover, as f is differentiable it is also continuous; thus $\lim_{j\to\infty} f(x^{n_j}) = f(x^*)$. This implies $\lim_{j\to\infty} h(x^{n_j}) = h(x^*)$. ■

Remark 6. The convergence properties shown in Theorem 4.8 should be the basic requirements of any algorithm. Very loosely speaking, the theorem states that the algorithm ends up in a meaningful solution. It allows us to formulate stopping conditions, e.g., the residual between successive function values.

Now, using Theorem 3.7, we can verify the convergence of the sequence $(x^n)_{n\in\mathbb{N}}$ generated by Algorithm 5. We assume that after a finite number of steps the sequence $(\delta_n)_{n\in\mathbb{N}}$ is constant and consider the sequence $(x^n)_{n\in\mathbb{N}}$ starting from this iteration (again denoted by $(x^n)_{n\in\mathbb{N}}$). For example, if δ_n is determined relative to the Lipschitz constant, then as the Lipschitz constant can be assumed constant after a finite number of iterations, δ_n is also constant starting from this iteration.

Theorem 4.9 (convergence of iPiano to a critical point). *Let $(x^n)_{n\in\mathbb{N}}$ be generated by Algorithm 5, and let $\delta_n = \delta$ for all $n \in \mathbb{N}$. Then, the sequence $(x^{n+1}, x^n)_{n\in\mathbb{N}}$ satisfies* (H1), (H2), *and* (H3) *for the function $H_\delta \colon \mathbb{R}^{2N} \to \mathbb{R} \cup \{\infty\}$, $(x,y) \mapsto h(x) + \delta\|x-y\|_2^2$.*

[2]In general, the existence of $(\xi^j)_{j=0}^\infty$ is not guaranteed. Compared to the general case, additionally $\lim_{j\to\infty} \xi^j = 0$ is known here.

Moreover, if $H_\delta(x,y)$ has the Kurdyka–Łojasiewicz property at a cluster point (x^*,x^*), then the sequence $(x^n)_{n\in\mathbb{N}}$ has finite length, $x^n \to x^*$ as $n\to\infty$, and (x^*,x^*) is a critical point of H_δ; hence x^* is a critical point of h.

Proof. First, we verify that assumptions (H1), (H2), and (H3) are satisfied. We consider the sequence $z^n = (x^n, x^{n-1})$ for all $n\in\mathbb{N}$ and the proper lower semicontinuous function $F = H_\delta$.

- Condition (H1) is proved in Proposition 4.7(a) with $a = c_2 \le \gamma_n$.
- To prove condition (H2), consider $w^{n+1} := (w_x^{n+1}, w_y^{n+1})^\top \in \partial H_\delta(x^{n+1}, x^n)$ with $w_x^{n+1} \in \partial g(x^{n+1}) + \nabla f(x^{n+1}) + 2\delta(x^{n+1}-x^n)$ and $w_y^{n+1} = -2\delta(x^{n+1}-x^n)$. The Lipschitz continuity of ∇f and using (19) to specify an element from $\partial g(x^{n+1})$ imply

$$\begin{array}{rcl} \|w^{n+1}\|_2 & \le & \|w_x^{n+1}\|_2 + \|w_y^{n+1}\|_2 \\ & \le & \|\nabla f(x^{n+1}) - \nabla f(x^n)\|_2 + (\frac{1}{\alpha_n} + 4\delta)\|x^{n+1}-x^n\|_2 \\ & & + \frac{\beta_n}{\alpha_n}\|x^n - x^{n-1}\|_2 \\ & \le & \frac{1}{\alpha_n}(\alpha_n L_n + 1 + 4\alpha_n\delta)\Delta_{n+1} + \frac{1}{\alpha_n}\beta_n\Delta_n\,. \end{array}$$

  As $\alpha_n L_n \le 2(1-\beta_n) \le 2$ and $\delta\alpha_n = 1 - \frac{1}{2}\alpha_n L_n - \frac{1}{2}\beta_n \le 1$, setting $b = \frac{7}{c_1}$ verifies condition (H2), i.e., $\|w^{n+1}\|_2 \le b(\Delta_n + \Delta_{n+1})$.
- In Theorem 4.8(4.8) it is proved that there exists a subsequence $(x^{n_j+1})_{j\in\mathbb{N}}$ of $(x^n)_{n\in\mathbb{N}}$ such that $\lim_{j\to\infty} h(x^{n_j+1}) = h(x^*)$. Proposition 4.7(4.7) shows that $\Delta_{n+1}\to 0$ as $n\to\infty$; hence $\lim_{j\to\infty} x^{n_j} = x^*$. As the term $\delta\|x-y\|_2^2$ is continuous in x and y, we deduce

$$\lim_{j\to\infty} H(x^{n_j+1}, x^{n_j}) = \lim_{j\to\infty} h(x^{n_j+1}) + \delta\|x^{n_j+1} - x^{n_j}\|_2 = H(x^*,x^*) = h(x^*)\,.$$

Now, the abstract convergence Theorem 3.7 concludes the proof. ■

The next corollary makes use of the fact that semialgebraic functions (Definition 3.4) have the Kurdyka–Łojasiewicz property.

Corollary 4.10 (convergence of iPiano for semialgebraic functions). *Let h be a semialgebraic function. Then, $H_\delta(x,y)$ is also semialgebraic. Furthermore, let $(x^n)_{n\in\mathbb{N}}$, $(\delta_n)_{n\in\mathbb{N}}$, $(x^{n+1},x^n)_{n\in\mathbb{N}}$ be as in Theorem 4.9. Then the sequence $(x^n)_{n\in\mathbb{N}}$ has finite length, $x^n\to x^*$ as $n\to\infty$, and x^* is a critical point of h.*

Proof. As h and $\delta\|x-y\|_2$ are semialgebraic, $H_\delta(x,y)$ is semialgebraic and has the KL property. Then, Theorem 4.9 concludes the proof. ■

4.6. Convergence rate. In the following, we are interested in determining a convergence rate with respect to the proximal residual from Definition 4.5. Since all preceding estimations are according to $\|x^{n+1}-x^n\|_2$, we establish the relation to $\|r(x)\|_2$ first. The following lemmas about the monotonicity and the nonexpansiveness of the proximity operator turn out to be very useful for that. Coarsely speaking, Lemma 4.11 states that the residual is sublinearly increasing. Lemma 4.12 formulates a standard property of the proximal operator.

Lemma 4.11 (proximal monotonicity). *Let $y,z\in\mathbb{R}^N$, and $\alpha>0$. Define the functions*

$$p_g(\alpha) := \frac{1}{\alpha}\|(I+\alpha\partial g)^{-1}(y-\alpha z) - y\|_2$$

and

$$q_g(\alpha) := \|(I + \alpha\partial g)^{-1}(y - \alpha z) - y\|_2.$$

Then, $p_g(\alpha)$ is a decreasing function of α, and $q_g(\alpha)$ increasing in α.

Proof. See, e.g., [36, Lemma 1] or [44, Lemma 4]. ■

Lemma 4.12 (nonexpansiveness). *Let g be a convex function and $\alpha > 0$; then, for all $x, y \in \operatorname{dom} g$ we obtain the nonexpansiveness of the proximity operator*

$$\|(I + \alpha\partial g)^{-1}(x) - (I + \alpha\partial g)^{-1}(y)\|_2 \le \|x - y\|_2 \quad \forall x, y \in \mathbb{R}^N. \tag{24}$$

Proof. The lemma is a well-known fact. See, for example, [6]. ■

The two preceding lemmas allow us to establish the following relation.

Lemma 4.13. *We have the following bound:*

$$\sum_{n=0}^{N} \|r(x^n)\|_2 \le \frac{2}{c_1} \sum_{n=0}^{N} \|x^{n+1} - x^n\|_2. \tag{25}$$

Proof. First, we observe the relations $1 \le \alpha \Rightarrow q_g(1) \le q_g(\alpha)$ and $1 \ge \alpha \Rightarrow p_g(1) \le p_g(\alpha) = \frac{1}{\alpha} q_g(\alpha)$, which are based on Lemma 4.11. Then, invoking the nonexpansiveness of the proximity operator (Lemma 4.12), we obtain

$$\begin{aligned} \beta_n \|x^n - x^{n-1}\|_2 &\ge \|x^n - \alpha_n \nabla f(x^n) + \beta_n(x^n - x^{n-1}) - (x^n - \alpha_n \nabla f(x^n))\|_2 \\ &\ge \|x^{n+1} - (I + \alpha_n\partial g)^{-1}(x^n - \alpha_n \nabla f(x^n))\|_2 \,. \end{aligned} \tag{26}$$

This allows us to compute the following lower bound:

$$\begin{aligned} \|x^{n+1} - x^n\|_2 &\ge \|x^{n+1} - x^n\|_2 - \beta_n \|x^n - x^{n-1}\|_2 \\ &\quad + \|x^{n+1} - (I + \alpha_n\partial g)^{-1}(x^n - \alpha_n \nabla f(x^n))\|_2 \\ &\ge \|x^n - (I + \alpha_n\partial g)^{-1}(x^n - \alpha_n \nabla f(x^n))\|_2 - \beta_n \|x^n - x^{n-1}\|_2 \\ &\ge \min(1, \alpha_n)\|r(x^n)\|_2 - \|x^n - x^{n-1}\|_2 \\ &\ge c_1 \|r(x^n)\|_2 - \|x^n - x^{n-1}\|_2 \,, \end{aligned}$$

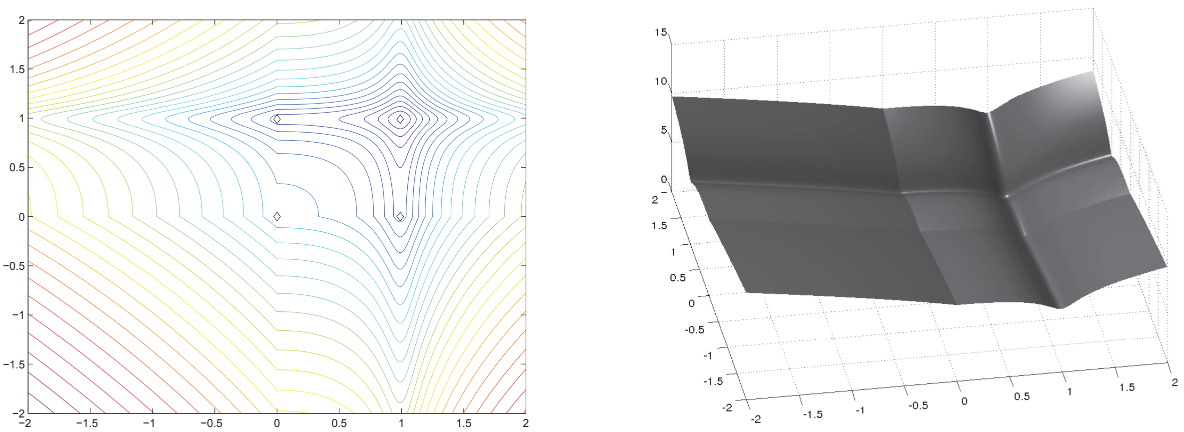
where the first inequality arises from adding zero and using (26), the second uses the triangle inequality, and the next applies Lemma 4.11 and $\beta_n < 1$. Now, summing both sides from $n = 0, \dots, N$ and using $x^{-1} = x^0$ the statement easily follows. ■

Next, we prove a global $\mathcal{O}(1/n)$ convergence rate for $\|x^{n+1} - x^n\|_2^2$ and the residuum $\|r(x^n)\|_2^2$ of the algorithm. The residuum provides an error measure of being a fixed point and hence a critical point of the problem. We first define the error μ_N to be the smallest squared ℓ_2 norm of successive iterates and, analogously, the error μ'_N

$$\mu_N := \min_{0 \le n \le N} \|x^n - x^{n-1}\|_2^2 \quad \text{and} \quad \mu'_N := \min_{0 \le n \le N} \|r(x^n)\|_2^2 \,.$$

Theorem 4.14. *Algorithm 5 guarantees that for all $N \ge 0$*

$$\mu'_N \le \frac{2}{c_1} \mu_N \quad \textit{and} \quad \mu_N \le c_2^{-1} \frac{h(x^0) - \underline{h}}{N + 1}.$$



(a) Contour plot of $h(x)$ (b) Energy landscape of $h(x)$

Figure 1. *Contour plot (left) and energy landscape (right) of the nonconvex function h shown in (27). The four diamonds mark stationary points of the function h.*

Proof. In view of Proposition 4.7(4.7), and the definition of γ_N in (21), summing up both sides of (23) for $n = 0, \dots, N$ and using that $\delta_N > 0$ from (21), we obtain

$$\underline{h} \leq h(x^0) - \sum_{n=0}^{N} \gamma_n \|x^n - x^{n-1}\|_2^2 \leq h(x^0) - (N+1) \min_{0 \leq n \leq N} \gamma_n \mu_N \,.$$

As $\gamma_n > c_2$, a simple rearrangement invoking Lemma 4.13 concludes the proof. ∎

Remark 7. The convergence rate $\mathcal{O}(1/N)$ for the squared ℓ_2 norm of our error measures is equivalent to stating a convergence rate $\mathcal{O}(1/\sqrt{N})$ for the error in the ℓ_2 norm.

Remark 8. A similar result can be found in [36] for the case $\beta = 0$.

5. Numerical experiments. In all of the following experiments, let $u, u^0 \in \mathbb{R}^N$ be vectors of dimension $N \in \mathbb{N}$, where N depends on the respective problem. In the case of an image, N is the number of pixels.

5.1. Ability to overcome spurious stationary points. Let us present some of the qualitative properties of the proposed algorithm. For this, we consider minimizing the following simple problem:

$$\min_{x \in \mathbb{R}^N} h(x) := f(x) + g(x)\,, \quad f(x) = \frac{1}{2} \sum_{i=1}^{N} \log(1 + \mu(x_i - u_i^0)^2)\,, \quad g(x) = \lambda \|x\|_1\,, \tag{27}$$

where x is the unknown vector, u^0 is some given vector, and $\lambda, \mu > 0$ are some free parameters. A contour plot and the energy landscape of h in the case of $N = 2$, $\lambda = 1$, $\mu = 100$, and $u^0 = (1, 1)^\top$ is depicted in Figure 1. It turns out that the function h has four stationary points, i.e., points $\bar{x}$, such that $0 \in \nabla f(\bar{x}) + \partial g(\bar{x})$. These points are marked by small black diamonds. Clearly the function f is nonconvex but has a Lipschitz continuous gradient with

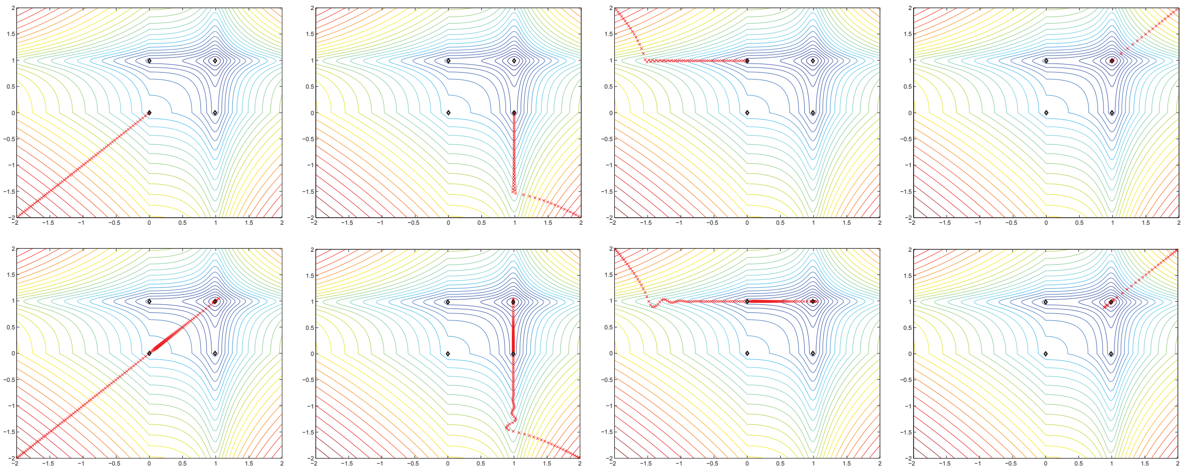


Figure 2. *The first row shows the result of the iPiano algorithm for four different starting points when using $\beta = 0$; the second row shows the results when using $\beta = 0.75$. While the algorithm without an inertial term gets stuck in unwanted local stationary points in three of four cases, the algorithm with an inertial term always succeeds in converging to the global optimum.*

components

$$\nabla f(x)_i = \mu \frac{x_i - u_i^0}{1 + \mu(x_i - u_i^0)^2}.$$

The Lipschitz constant of ∇f is easily computed as $L = \mu$. The function g is nonsmooth but convex, and the proximal operator with respect to g is given by the well-known shrinkage operator

$$(I + \alpha \partial g)^{-1}(y) = \max(0, |y| - \alpha\lambda) \cdot \operatorname{sgn}(y)\,, \tag{28}$$

where all operations are understood componentwise. Let us test the performance of the proposed algorithm on the example shown in Figure 1. We set $\alpha = 2(1-\beta)/L$. Figure 2 shows the results of using the iPiano algorithm for different settings of the extrapolation factor β. We observe that iPiano with $\beta = 0$ is strongly attracted by the closest stationary points, while switching on the inertial term can help to overcome the spurious stationary points. The reason for this desired property is that while the gradient might vanish at some points, the inertial term $\beta(x^n - x^{n-1})$ is still strong enough to drive the sequence out of the stationary region. Clearly, there is no guarantee that iPiano always avoids spurious stationary points. iPiano is not designed to find the global optimum. However, our numerical experiments suggest that in many cases, iPiano finds lower energies than the respective algorithm without inertial term. A similar observation about the Heavy-ball method is described in [8].

5.2. Image processing applications. It is well known that nonconvex regularizers are better models for many image processing and computer vision problems; see, e.g., [9, 21, 25, 41]. However, convex models are still preferred over nonconvex models, since they can be efficiently optimized using convex optimization algorithms. In this section, we demonstrate the applicability of the proposed algorithm to solving a class of nonconvex regularized variational models. We present examples for natural image denoising and linear diffusion based image compression. We show that iPiano can be easily adapted to all of these problems and yields state-of-the-art results.

5.2.1. Student-t regularized image denoising. In this subsection, we investigate the task of natural image denoising. For this we exploit an optimized Markov random field (MRF) model (see [13]) and make use of the iPiano algorithm to solve it. In order to evaluate the performance of iPiano, we compare it to the well-known bound constrained limited memory quasi Newton method (L-BFGS) [29].[3] As an error measure, we use the energy difference

$$\mathcal{E}^n = h^n - h^*, \tag{29}$$

where h^n is the energy of the current iteration n and h^* is the energy of the true solution. Clearly, this error measure makes sense only when different algorithms can achieve the same true energy h^*, which is in general wrong for nonconvex problems. In our image denoising experiments, however, we find that all tested algorithms find the same solution, independent of the initialization. This can be explained by the fact that the learning procedure [13] also delivers models that are relatively easy to optimize, since otherwise they would have resulted in a bad training error. In order to compute a true energy h^*, we run the iPiano algorithm with a proper β (e.g., $\beta = 0.8$) for enough iterations ($\sim$1000 iterations). We run all the experiments in MATLAB on a 64-bit Linux server with 2.53GHz CPUs.

The MRF image denoising model based on learned filters is formulated as

$$\min_{u\in\mathbb{R}^N} \sum_{i=1}^{N_f} \vartheta_i \Phi(K_i u) + g_{1,2}(u, u^0)\,, \tag{30}$$

where u and $u^0 \in \mathbb{R}^N$ denote the sought solution and the noisy input image, respectively, Φ is the nonconvex penalty function, $\Phi(K_i u) = \sum_p \varphi((K_i u)_p)$, K_i are learned, linear operators with the corresponding weights ϑ_i, and N_f is the number of the filters. The linear operators K_i are implemented as two-dimensional convolutions of the image u with small (e.g., 7×7) filter kernels k_i, i.e., $K_i u = k_i * u$. The function $g_{1,2}$ is the data term, which depends on the respective problem. In the case of Gaussian noise, $g_{1,2}$ is given as

$$g_2(u, u^0) = \frac{\lambda}{2}\|u - u^0\|_2^2\,,$$

and for the impulse noise (e.g., salt and pepper noise), $g_{1,2}$ is given as

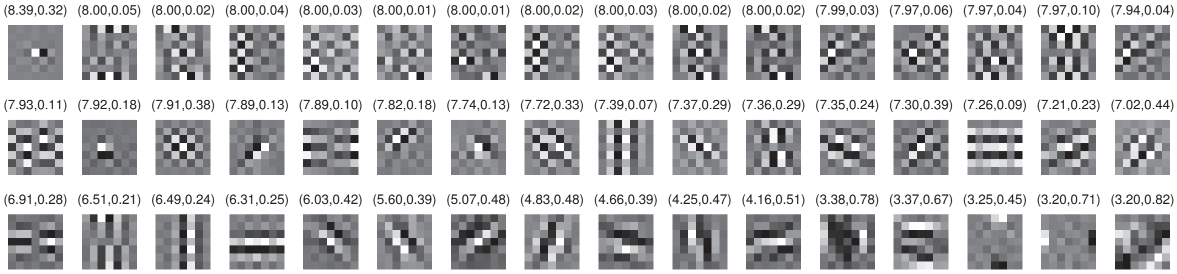
$$g_1(u, u^0) = \lambda\|u - u^0\|_1\,.$$

The parameter $\lambda > 0$ is used to define the tradeoff between regularization and data fitting.

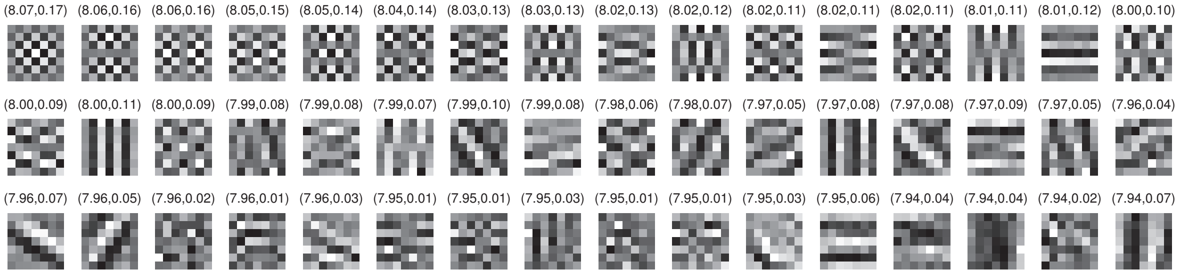
In this paper, we consider the following nonconvex penalty function, which is derived from the Student-t distribution:

$$\varphi(t) = \log(1 + t^2)\,. \tag{31}$$

[3] We make use of the implementation distributed at http://www.cs.toronto.edu/~liam/software.shtml.



(a) Learned filters for the MRF-ℓ_2 model



(b) Learned filters for the MRF-ℓ_1 model

Figure 3. 48 *learned filters of size* 7×7 *for two different MRF denoising models. The first number in the bracket is the weights* ϑ_i*, and the second one is the norm* $\|k_i\|_2$ *of the filters.*

Concerning the filters k_i, for the ℓ_2 model (MRF-ℓ_2), we make use of the filters learned in [13] by using a bilevel learning approach. The filters are shown in Figure 3(a) together with the corresponding weights ϑ_i. For the MRF-ℓ_1 denoising model, we employ the same bilevel learning algorithm to train a set of optimal filters specialized for the ℓ_1 data term and input images degraded by salt and pepper noise. Since the bilevel learning algorithm requires a twice continuously differentiable model, we replace the ℓ_1 norm by a smooth approximation during training. The learned filters for the MRF-ℓ_1 model together with the corresponding weights ϑ_i are shown in Figure 3(b).

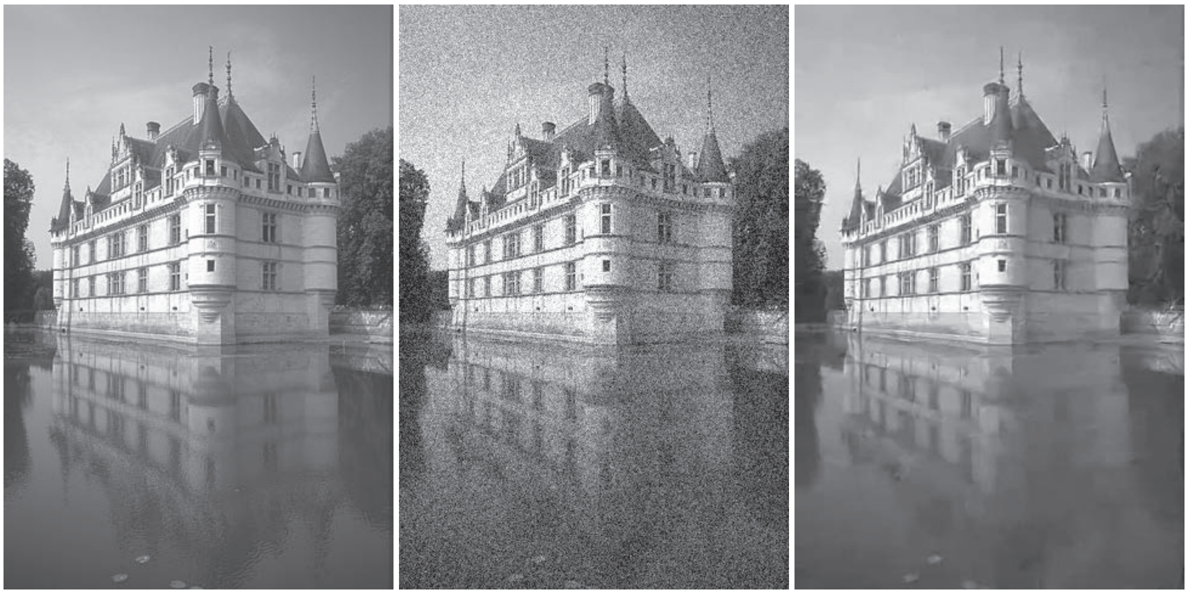
Let us now explain how to solve (30) using the iPiano algorithm. Casting (30) in the form of (9), we see that $f(u) = \sum_{i=1}^{N_f} \vartheta_i \Phi(K_i u)$ and $g(u) = g_{1,2}(u, u^0)$. Thus, we have

$$\nabla f(u) = \sum_{i=1}^{N_f} \vartheta_i K_i^\top \Phi'(K_i u)\,,$$

where $\Phi'(K_i u) = [\varphi'((K_i u)_1)\,, \varphi'((K_i u)_2), \ldots, \varphi'((K_i u)_p)]^\top$ and $\varphi'(t) = 2t/(1+t^2)$. The proximal map with respect to g simply poses pointwise operations. For the case of g_2, it is given by

$$u = (I + \alpha\partial g)^{-1}(\hat{u}) \Longleftrightarrow u_p = \frac{\hat{u}_p + \alpha\lambda u_p^0}{1+\alpha\lambda}\,, \quad p = 1, \ldots, N,$$

and for the function g_1, it is given by the well-known soft shrinkage operator (28), which in



(a) Clean image (b) Noisy image ($\sigma = 25$) (c) Denoised image

Figure 4. *Natural image denoising by using student-t regularized MRF model (MRF-ℓ_2). The noisy version is corrupted by additive zero mean Gaussian noise with $\sigma = 25$.*

the case of the MRF-ℓ_1 model becomes

$$u = (I + \alpha\partial g)^{-1}(\hat{u}) \iff$$
$$u_p = \max(0,\ |\hat{u}_p - u_p^0| - \alpha\lambda) \cdot \operatorname{sgn}(\hat{u}_p - u_p^0) + u_p^0\,, \quad p = 1, \dots, N.$$

Now, we can make use of our proposed algorithm to solve the nonconvex optimization problems. In order to evaluate the performance of iPiano, we compare it to L-BFGS. To use L-BFGS, we merely need the gradient of the objective function with respect to u. For the MRF-ℓ_2 model, calculating the gradients is straightforward. However, in the case of the MRF-ℓ_1 model, due to the nonsmooth function g, we cannot directly use L-BFGS. Since L-BFGS can easily handle box constraints, we can get rid of the nonsmooth function ℓ_1 norm by introducing two box constraints.

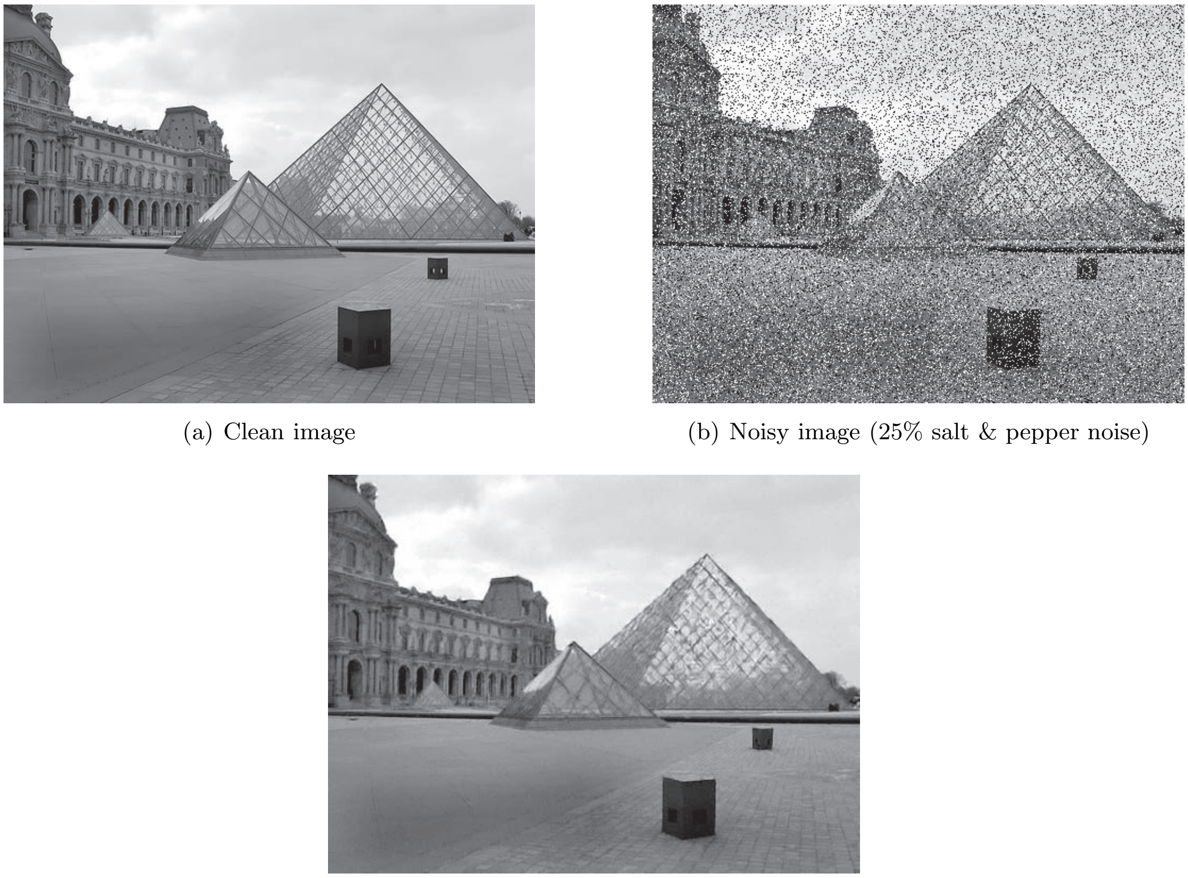
Lemma 5.1. *The MRF-ℓ_1 model can be equivalently written as the bound-constraint problem*

$$\min_{w,v} \sum_{i=1}^{N_f} \vartheta_i \Phi(K_i(w + v)) + \lambda\, 1^\top (v - w) \quad s.t. \quad w \le u^0/2,\ v \ge u^0/2\,. \tag{32}$$

Proof. It is well known that the ℓ_1 norm $\|u - u^0\|_1$ can be equivalently expressed as

$$\|u - u^0\|_1 = \min_t 1^\top t \quad \text{s.t.} \quad t \ge u - u^0\,, \quad t \ge -u + u^0\,,$$

where $t \in \mathbb{R}^N$ and the inequalities are understood pointwise. Letting $w = (u - t)/2 \in \mathbb{R}^N$ and $v = (u + t)/2 \in \mathbb{R}^N$, we find $u = w + v$ and $t = v - w$. Substituting u and t back into (30) while using the above formulation of the ℓ_1 norm yields the desired transformation. ∎



(a) Clean image

(b) Noisy image (25% salt & pepper noise)

(c) Denoised image

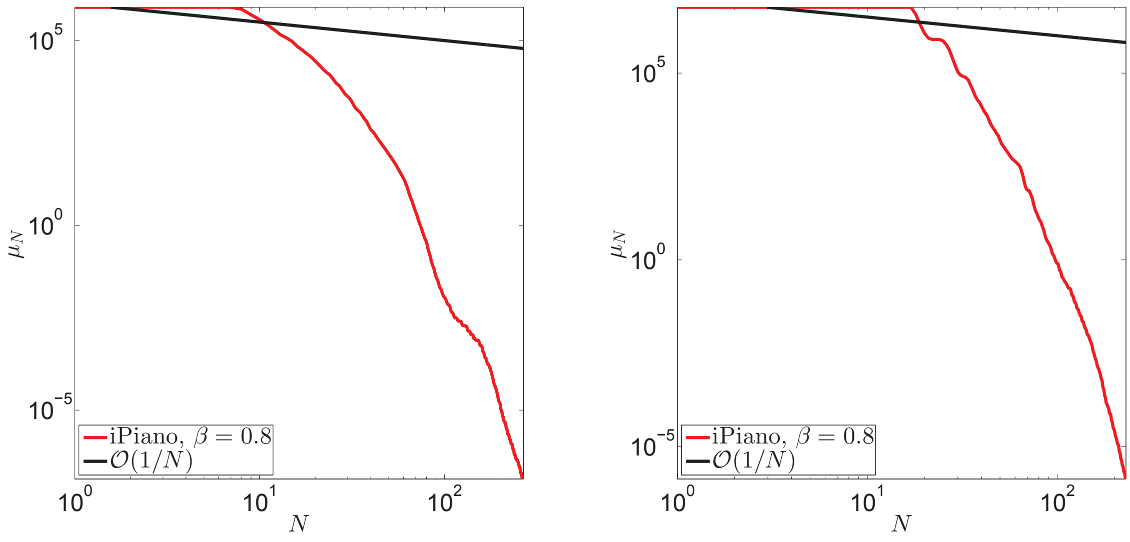
Figure 5. *Natural image denoising in the case of impulse noise by using the MRF-ℓ_1 model. The noisy version is corrupted by* 25% *salt and pepper noise.*

Figures 4 and 5, respectively, show a denoising example using the MRF-ℓ_2 model and the MRF-ℓ_1 model. In both experiments, we use the iPiano version with backtracking (Algorithm 4) with the following parameter settings:

$$L_{-1} = 1,\ \eta = 1.2,\ \alpha_n = 1.99(1-\beta)/L_n\,,$$

where β is a free parameter to be evaluated in the experiment. In order to make use of possible larger step sizes in practice, we use the following trick: when the inequality (15) is fulfilled, we decrease the evaluated Lipschitz constant L_n slightly by setting $L_n = L_n/1.05$.

For the MRF-ℓ_2 denoising experiments, we initialized u using the noisy image itself; however, for the MRF-ℓ_1 denoising model, we initialized u using a zero image. We found that this initialization strategy usually gives good convergence behavior for both algorithms. For both denoising examples, we run the algorithms until the error $\mathcal{E}^n$ decreases to a certain predefined threshold tol. We then record the required number of iterations and the run time. We summarize the results of the iPiano algorithm with different settings and L-BFGS in Tables 1 and



(a) MRF-ℓ_2 model (b) MRF-ℓ_1 model

Figure 6. *Convergence rates for the MRF-ℓ_2 and -ℓ_1 models. The figures plot the minimal residual norm μ_N, which also bounds the proximal residual μ'_N. Note that the empirical convergence rate is much faster compared to the worst case rate (see Theorem 4.14).*

Table 1

The number of iterations and the run time necessary for reaching the corresponding error for iPiano and L-BFGS to solve the MRF-ℓ_2 model. T_1 is the run time of iPiano with $\beta = 0.8$, and T_2 shows the run time of L-BFGS.

	iPiano with different β							L-BFGS	
tol	0.00	0.20	0.40	0.60	0.80	0.95	T_1(s)	iter.	T_2(s)
10^3	260	182	116	66	56	214	34.073	43	18.465
10^2	372	256	164	94	67	257	40.199	55	22.803
10^1	505	344	222	129	79	299	47.177	66	27.054
10^0	664	451	290	168	98	342	59.133	79	32.143
10^{-1}	857	579	371	216	143	384	85.784	93	36.926
10^{-2}	1086	730	468	271	173	427	103.436	107	41.939
10^{-3}	1347	904	577	338	199	473	119.149	124	48.272
10^{-4}	1639	1097	697	415	232	524	138.416	139	53.290
10^{-5}	1949	1300	827	494	270	569	161.084	154	58.511

2. From these two tables, one can draw the common conclusion that iPiano with a proper inertial term takes significantly fewer iterations compared to the case without inertial term, and in practice $\beta \approx 0.8$ is generally a good choice.

In Table 1, one can see that the iPiano algorithm with $\beta = 0.8$ takes slightly more iterations and a longer run time to reach a solution of moderate accuracy (e.g., tol $= 10^3$) compared with L-BFGS. However, for highly accurate solutions (e.g., tol $= 10^{-5}$), this gap increases. For the case of the nonsmooth MRF-ℓ_1 model, the result is just the reverse. It is shown in Figure 2

Table 2

The number of iterations and the run time necessary for reaching the corresponding error for iPiano and L-BFGS to solve the MRF-ℓ_1 model. T_1 is the run time of iPiano with $\beta = 0.8$, and T_2 shows the run time of L-BFGS.

	iPiano with different β							L-BFGS	
tol	0.00	0.20	0.40	0.60	0.80	0.95	T_1(s)	iter.	T_2(s)
10^3	390	272	174	96	64	215	43.709	223	102.383
10^2	621	403	256	145	77	260	53.143	246	112.408
10^1	847	538	341	195	96	304	65.679	265	121.303
10^0	1077	682	433	247	120	349	81.761	285	130.846
10^{-1}	1311	835	530	303	143	395	97.060	298	136.326
10^{-2}	1559	997	631	362	164	440	111.579	311	141.876
10^{-3}	1818	1169	741	424	185	485	126.272	327	148.945
10^{-4}	2086	1346	853	489	208	529	142.083	347	157.956
10^{-5}	2364	1530	968	557	233	575	159.493	372	169.674

that for reaching a moderately accurate solution, iPiano with $\beta = 0.8$ consumes significantly fewer iterations and a shorter run time, and for the solution of high accuracy, it still can save much computation.

Figure 6 plots the error μ_N over the number of required iterations N for both the MRF-ℓ_2 and -ℓ_1 models using $\beta = 0.8$. From the plots it becomes obvious that the empirical performance of the iPiano algorithm is much better compared to the worst-case convergence rate of $\mathcal{O}(1/N)$ as provided in Theorem 4.14.

The iPiano algorithm has an additional advantage of simplicity. The iPiano version without backtracking basically relies on matrix vector products (filter operations in the denoising examples) and simple pointwise operations. Therefore, the iPiano algorithm is well suited for a parallel implementation on GPUs which can lead to speedup factors of 20–30.

5.2.2. Linear diffusion based image compression. In this example we apply the iPiano algorithm to linear diffusion based image compression. Recent works [20, 42] have shown that image compression based on linear and nonlinear diffusion can outperform the standard JPEG standard and even the more advanced JPEG 2000 standard when the interpolation points are carefully chosen. Therefore, finding optimal data for interpolation is a key problem in the context of PDE-based image compression. There exist only a few prior works on this topic (see, e.g., [33, 24]), and the very recent approach presented in [24] defines the state-of-the-art.

The problem of finding optimal data for homogeneous diffusion based interpolation is formulated as the following constrained minimization problem:

$$\begin{aligned} &\min_{u,c} \frac{1}{2}\|u-u^0\|_2^2 + \lambda\|c\|_1 \\ &\text{s.t. } C(u-u^0) - (I-C)Lu = 0\,, \end{aligned} \tag{33}$$

where $u^0 \in \mathbb{R}^N$ denotes the ground truth image, $u \in \mathbb{R}^N$ denotes the reconstructed image, and $c \in \mathbb{R}^N$ denotes the inpainting mask, i.e., the characteristic function of the set of points that are chosen for compressing the image. Furthermore, we denote by $C = \operatorname{diag}(c) \in \mathbb{R}^{N\times N}$ the diagonal matrix with the vector c on its main diagonal, by I the identity matrix, and by

$L \in \mathbb{R}^{N\times N}$ the Laplacian operator. Compared to the original formulation [24], we omit a very small quadratic term $\frac{\varepsilon}{2}\|c\|_2^2$, because we find it unnecessary in experiments.

Observe that if $c \in [0,1)^N$, we can multiply the constraint equation in (33) from the left by $(I-C)^{-1}$ such that it becomes

$$E(c)(u-u^0) - Lu = 0\,,$$

where $E(c) = \operatorname{diag}(c_1/(1-c_1),\ldots,c_N/(1-c_N))$. This shows that problem (33) is in fact a reduced formulation of the bilevel optimization problem

$$\begin{aligned}(34)\qquad & \min_c \frac{1}{2}\|u(c)-u^0\|_2^2 + \lambda\|c\|_1 \\ & \text{s.t.}\quad u(c) = \arg\min_u \|Du\|_2^2 + \|E(c)^{\frac{1}{2}}(u-u^0)\|_2^2\,,\end{aligned}$$

where D is the nabla operator and hence $-L = D^\top D$.

Problem (33) is nonconvex due to the nonconvexity of the equality constraint. In [24], the above problem is solved by a successive primal-dual (SPD) algorithm, which successively linearizes the nonconvex constraint and solves the resulting convex problem with the first-order primal-dual algorithm [12]. The main drawback of SPD is that it requires tens of thousands of inner iterations and thousands of outer iterations to reach a reasonable solution. However, as we now demonstrate, iPiano can solve this problem with higher accuracy in 1000 iterations.

Observe that we can rewrite problem (33) by solving u from the constraints equation, which gives

$$u = A^{-1}Cu^0\,,$$

where $A = C + (C-I)L$. In [32], it is shown that the A is invertible as long as at least one element of c is nonzero, which is the case for nondegenerate problems. Substituting the above equation back into (33), we arrive at the following optimization problem, which now depends only on the inpainting mask c:

$$(35)\qquad \min_c \frac{1}{2}\|A^{-1}Cu^0 - u^0\|_2^2 + \lambda\|c\|_1\,.$$

Casting (35) in the form of (9), we have $f(c) = \frac{1}{2}\|A^{-1}Cu^0 - u^0\|_2^2$, and $g(c) = \lambda\|c\|_1$. In order to minimize the above problem using iPiano, we need to calculate the gradient of f with respect to c. This is shown by the following lemma.

Lemma 5.2. *Let*

$$f(c) = \frac{1}{2}\|A^{-1}Cu^0 - u^0\|_2^2\,;$$

then

$$(36)\qquad \nabla f(c) = \operatorname{diag}(-(I+L)u + u^0)(A^\top)^{-1}(u-u^0)\,.$$

Proof. Differentiating both sides of

$$f = \frac{1}{2}\|u-u^0\|_2^2 = \frac{1}{2}\left\langle u-u^0, u-u^0\right\rangle\,,$$

we obtain

$$\mathrm{d}f = \left\langle \mathrm{d}u, u - u^0 \right\rangle . \tag{37}$$

In view of $u = A^{-1}Cu^0$ and $\mathrm{d}A^{-1} = -A^{-1}\mathrm{d}AA^{-1}$, we further have

$$\begin{aligned}
\mathrm{d}u &= \mathrm{d}A^{-1}Cu^0 + A^{-1}\mathrm{d}Cu^0 \\
&= -A^{-1}\mathrm{d}AA^{-1}Cu^0 + A^{-1}\mathrm{d}Cu^0 \\
&= -A^{-1}\mathrm{d}Au + A^{-1}\mathrm{d}Cu^0 \\
&= -A^{-1}\mathrm{d}C(I+L)u + A^{-1}\mathrm{d}Cu^0 \\
&= A^{-1}\mathrm{d}C(-\left(I+L)u + u^0\right) .
\end{aligned}$$

Let $t = -(I+L)u + u^0 \in \mathbb{R}^N$, and since C is a diagonal matrix, we have

$$\mathrm{d}Ct = \mathrm{diag}(\mathrm{d}c)t = \mathrm{diag}(t)\mathrm{d}c ,$$

and hence

$$\mathrm{d}u = A^{-1}\,\mathrm{diag}(t)\mathrm{d}c . \tag{38}$$

By substituting (38) into (37), we obtain

$$\begin{aligned}
\mathrm{d}f &= \left\langle (A^{-1}\,\mathrm{diag}(t))\mathrm{d}c, u - u^0 \right\rangle \\
&= \left\langle \mathrm{d}c, (A^{-1}\,\mathrm{diag}(t))^\top (u - u^0) \right\rangle .
\end{aligned}$$

Finally, the gradient is given by

$$\begin{aligned}
\nabla f &= (A^{-1}\,\mathrm{diag}(t))^\top (u - u^0) \\
&= \mathrm{diag}(-(I+L)u + u^0)(A^\top)^{-1}(u - u^0) . \qquad \blacksquare
\end{aligned} \tag{39}$$

Finally, we need to compute the proximal map with respect to $g(c)$, which is again given by a pointwise application of the shrinkage operator (28).

Now, we can make use of the iPiano algorithm to solve problem (35). We set $\beta = 0.8$, which generally performs very well in practice. We additionally accelerate the SPD algorithm used in the previous work [24] by applying the diagonal preconditioning technique [37], which significantly reduces the required iterations for the primal-dual algorithm in the inner loop.

Figure 7 shows examples of finding optimal interpolation data for the three test images. Table 3 summarizes the results of two different algorithms. Regarding the reconstruction quality, we make use of the mean squared error (MSE) as an error measurement to be consistent with previous work; the MSE is computed by

$$MSE(u, u^0) = \frac{1}{N}\sum_{i=1}^{N}(u_i - u_i^0)^2 .$$

From Table 3, one can see that the Successive PD algorithm requires 200×4000 iterations to converge. iPiano needs only 1000 iterations to reach a lower energy. Note that in each iteration of the iPiano algorithm, two linear systems have to be solved. In our implementation we use the MATLAB "backslash" operator, which effectively exploits the strong sparseness of the systems. A lower energy basically implies that iPiano can solve the minimization problem (33) better. Regarding the final compression result, usually the result of iPiano has slightly less density but slightly worse MSE. Following the work [33], we also consider the so-called gray value optimization (GVO) as a postprocessing step to further improve the MSE of the reconstructed images.

Table 3
Summary of two algorithms for three test images.

Test image	Algorithm	Iterations	Energy	Density	MSE	with GVO
trui	iPiano	1000	21.574011	4.98%	17.31	16.89
	SPD	200/4000	21.630280	5.08%	17.06	16.54
peppers	iPiano	1000	20.631985	4.84%	19.50	18.99
	SPD	200/4000	20.758777	4.93%	19.48	18.71
walter	iPiano	1000	10.246041	4.82%	8.29	8.03
	SPD	200/4000	10.278874	4.93%	8.01	7.72

6. Conclusions. In this paper, we have proposed a new optimization algorithm, which we call iPiano. It is applicable to a broad class of nonconvex problems. More specifically, it addresses objective functions, which are composed as a sum of a differentiable (possibly nonconvex) and a convex (possibly nondifferentiable) function. The basic methodologies have been derived from the forward-backward splitting algorithm and the Heavy-ball method.

Our theoretical convergence analysis is divided into two steps. First, we have proved an abstract convergence result about inexact descent methods. Then, we analyzed the convergence of iPiano. For iPiano, we have proved that the sequence of function values converges, that the subsequence of arguments generated by the algorithm is bounded, and that every limit point is a critical point of the problem. Requiring the Kurdyka–Łojasiewicz property for the objective function establishes deeper insights into the convergence behavior of the algorithm. Using the abstract convergence result, we have shown that the whole sequence converges and the unique limit point is a stationary point.

The analysis includes an examination of the convergence rate. A rough upper bound of $O(1/n)$ has been found for the squared proximal residual. Experimentally, iPiano has been shown to have a much faster convergence rate.

Finally, the applicability of the algorithm has been demonstrated, and iPiano achieved state-of-the-art performance. The experiments comprised denoising and image compression. In the first two experiments, iPiano helped in learning a good prior for the problem. In the case of image compression, iPiano has demonstrated its use in a huge optimization problem for computing an optimal mask for a Laplacian PDE based image compression method.

In summary, iPiano has many favorable theoretical properties, is simple, and is efficient. Hence, we recommend it as a standard solver for the considered class of problems.

(a) Test image (256×256) (b) Optimized mask (c) Reconstruction

Figure 7. *Examples of finding optimal inpainting mask for Laplace interpolation based image compression by using iPiano.* First row: *Test image trui of size* 256×256. *Parameter* $\lambda = 0.0036$, *the optimized mask has a density of* 4.98*% and the MSE of the reconstructed image is* 16.89. Second row: *Test image peppers of size* 256×256. *Parameter* $\lambda = 0.0034$, *the optimized mask has a density of* 4.84%, *and the MSE of the reconstructed image is* 18.99. Third row: *Test image walter of size* 256×256. *Parameter* $\lambda = 0.0018$, *the optimized mask has a density of* 4.82%, *and the MSE of the reconstructed image is* 8.03.

Acknowledgment. We are grateful to Joachim Weickert for discussions about the image compression by diffusion problem.

REFERENCES

[1] F. Alvarez, *Weak convergence of a relaxed and inertial hybrid projection-proximal point algorithm for maximal monotone operators in Hilbert space*, SIAM J. Optim., 14 (2004), pp. 773–782.

[2] F. Alvarez and H. Attouch, *An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping*, Set-Valued Anal., 9 (2001), pp. 3–11.

[3] H. Attouch and J. Bolte, *On the convergence of the proximal algorithm for nonsmooth functions involving analytic features*, Math. Program., 116 (2008), pp. 5–16.

[4] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran, *Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality*, Math. Oper. Res., 35 (2010), pp. 438–457.

[5] H. Attouch, J. Bolte, and B. Svaiter, *Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods*, Math. Program., 137 (2013), pp. 91–129.

[6] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, CMS Books Math., Springer, New York, 2011.

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

[8] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed., Athena Scientific, Cambridge, MA, 1999.

[9] M. J. Black and A. Rangarajan, *On the unification of line processes, outlier rejection, and robust statistics with applications in early vision*, Internat. J. Comput. Vis., 19 (1996), pp. 57–91.

[10] J. Bolte, A. Daniilidis, A. Ley, and L. Mazet, *Characterizations of Łojasiewicz inequalities: Subgradient flows, talweg, convexity*, Trans. Amer. Math. Soc., 362 (2010), pp. 3319–3363.

[11] K. Bredies and D. A. Lorenz, *Minimization of Non-smooth, Non-convex Functionals by Iterative Thresholding*, preprint, available online at http://www.uni-graz.at/~bredies/publications.html, 2009.

[12] A. Chambolle and T. Pock, *A first-order primal-dual algorithm for convex problems with applications to imaging*, J. Math. Imaging Vis., 40 (2011), pp. 120–145.

[13] Y.J. Chen, T. Pock, R. Ranftl, and H. Bischof, *Revisiting loss-specific training of filter-based MRFs for image restoration*, in German Conference on Pattern Recognition (GCPR), Lecture Notes in Comput. Sci. 8142, Springer-Verlag, Berlin, 2013, pp. 271–281.

[14] E. Chouzenoux, J.-C. Pesquet, and A. Repetti, *Variable metric forward-backward algorithm for minimizing the sum of a differentiable function and a convex function*, J. Optim. Theory Appl., 2013, available online at http://link.springer.com/article/10.1007/s10957-013-0465-7.

[15] P. L. Combettes and J.-C. Pesquet, *Proximal splitting methods in signal processing*, in Fixed-Point Algorithms for Inverse Problems in Science and Engineering, H.H. Bauschke, R.S. Burachik, P.L. Combettes, V. Elser, D.R. Luke, and H. Wolkowicz, eds., Springer, New York, 2011, pp. 185–212.

[16] P. L. Combettes and V. R. Wajs, *Signal recovery by proximal forward-backward splitting*, Multiscale Model. Simul., 4 (2005), pp. 1168–1200.

[17] Y. Drori and M. Teboulle, *Performance of first-order methods for smooth convex minimization: A novel approach*, Math. Program., 145 (2014), pp. 451–482.

[18] J. Eckstein and D. P. Bertsekas, *On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Math. Program., 55 (1992), pp. 293–318.

[19] M. Fukushima and H. Mine, *A generalized proximal point algorithm for certain non-convex minimization problems*, Internat. J. Systems Sci., 12 (1981), pp. 989–1000.

[20] I. Galic, J. Weickert, M. Welk, A. Bruhn, A. G. Belyaev, and H.-P. Seidel, *Image compression with anisotropic diffusion*, J. Math. Imaging Vis., 31 (2008), pp. 255–269.

[21] S. Geman and D. Geman, *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images*, IEEE Trans. Pattern Anal. Machine Intell., 6 (1984), pp. 721–741.

[22] A. A. Goldstein, *Convex programming in Hilbert space*, Bull. Amer. Math. Soc., 70 (1964), pp. 709–710.

[23] B. He and X. Yuan, *Convergence analysis of primal-dual algorithms for a saddle-point problem: From contraction perspective*, SIAM J. Imaging Sci., 5 (2012), pp. 119–149.

[24] L. Hoeltgen, S. Setzer, and J. Weickert, *An optimal control approach to find sparse data for Laplace interpolation*, in International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR), Lund, Sweden, 2013, pp. 151–164.

[25] J. Huang and D. Mumford, *Statistics of natural images and models*, in International Conference on Computer Vision and Pattern Recognition (CVPR), Fort Collins, CO, 1999, pp. 541–547.
[26] K. Kurdyka, *On gradients of functions definable in o-minimal structures*, Ann. Inst. Fourier, 48 (1998), pp. 769–783.
[27] E. S. Levitin and B.T. Polyak, *Constrained minimization methods*, USSR Comput. Math. Math. Phys., 6 (1966), pp. 1–50.
[28] P. L. Lions and B. Mercier, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal., 16 (1979), pp. 964–979.
[29] D. C. Liu and J. Nocedal, *On the limited memory BFGS method for large scale optimization*, Math. Program., 45 (1989), pp. 503–528.
[30] S. Łojasiewicz, *Une propriété topologique des sous-ensembles analytiques réels*, in Les Équations aux Dérivées Partielles (Paris, 1962), Éditions du centre National de la Recherche Scientifique, Paris, 1963, pp. 87–89.
[31] S. Łojasiewicz, *Sur la géométrie semi- et sous- analytique*, Ann. Inst. Fourier, 43 (1993), pp. 1575–1595.
[32] M. Mainberger, A. Bruhn, J. Weickert, and S. Forchhammer, *Edge-based compression of cartoon-like images with homogeneous diffusion*, Pattern Recognition, 44 (2011), pp. 1859–1873.
[33] M. Mainberger, S. Hoffmann, J. Weickert, C. H. Tang, D. Johannsen, F. Neumann, and B. Doerr, *Optimising spatial and tonal data for homogeneous diffusion inpainting*, in International Conference on Scale Space and Variational Methods in Computer Vision (SSVM), Ein-Gedi, The Dead Sea, Israel, 2011, pp. 26–37.
[34] A. Moudafi and M. Oliny, *Convergence of a splitting inertial proximal method for monotone operators*, J. Comput. Appl. Math., 155 (2003), pp. 447–454.
[35] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, Appl. Optim. 87, Kluwer Academic Publishers, Boston, MA, 2004.
[36] Y. Nesterov, *Gradient methods for minimizing composite functions*, Math. Program., 140 (2013), pp. 125–161.
[37] T. Pock and A. Chambolle, *Diagonal preconditioning for first order primal-dual algorithms in convex optimization*, in Proceedings of the IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 2011, pp. 1762–1769.
[38] B. T. Polyak, *Some methods of speeding up the convergence of iteration methods*, USSR Comput. Math. Math. Phys., 4 (1964), pp. 1–17.
[39] R. T. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM J. Control Optim., 14 (1976), pp. 877–898.
[40] R. T. Rockafellar, *Variational Analysis*, Grundlehren Math. Wiss. 317, Springer, Berlin, Heidelberg, 1998.
[41] S. Roth and M. J. Black, *Fields of experts*, Internat. J. Comput. Vis., 82 (2009), pp. 205–229.
[42] C. Schmaltz, J. Weickert, and A. Bruhn, *Beating the quality of JPEG* 2000 *with anisotropic diffusion*, in Proceedings of the DAGM Symposium, Jena, Germany, 2009, pp. 452–461.
[43] M. V. Solodov, *Convergence analysis of perturbed feasible descent methods*, J. Optim. Theory Appl., 93 (1997), pp. 337–353.
[44] S. Sra, *Scalable nonconvex inexact proximal splitting*, in Proceedings of the 25th Conference on Advances in Neural Information Processing Systems (NIPS), P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, eds., Lake Tahoe, NV, 2012, pp. 539–547.
[45] S. K. Zavriev and F. V. Kostyuk, *Heavy-ball method in nonconvex optimization problems*, Comput. Math. Model., 4 (1993), pp. 336–341.