Accepted for publication in Journal of Mathematical Imaging and Vision (JMIV)

iPiasco: Inertial Proximal Algorithm for strongly convex Optimization

Peter Ochs[1], Thomas Brox[1], and Thomas Pock[2]

[1] University of Freiburg, Germany
{ochs,brox}@cs.uni-freiburg.de

[2] Graz University of Technology, Austria
pock@icg.tugraz.at

January 2015

Abstract

In this paper, we present a forward–backward splitting algorithm with additional inertial term for solving a strongly convex optimization problem of a certain type. The strongly convex objective function is assumed to be a sum of a non-smooth convex and a smooth convex function. This additional knowledge is used for deriving a worst-case convergence rate for the proposed algorithm. It is proved to be an optimal algorithm with linear rate of convergence. For certain problems this linear rate of convergence is better than the provably optimal worst-case rate of convergence for smooth strongly convex functions. We demonstrate the efficiency of the proposed algorithm in numerical experiments and examples from image processing.

1 Introduction

In this paper, we study the convergence rate of an algorithm for minimizing composite objective functions of the form:

$$\min_{x\in X} \; f(x) + g(x)\,,$$

where X is a finite dimensional real vector space. The objective function is composed of a smooth, convex function f and a non-smooth, convex function g. Additionally, we assume that the objective function is strongly convex. On one hand, these assumptions are very restrictive, however, on the other hand, this class of objective functions can be optimized very efficiently. When $g \equiv 0$, there are algorithms that converge with a linear rate, which is the provably optimal worst-case rate for the considered class of problems [21]. Examples are the conjugate gradient method or the Heavy-ball method [27]. We propose an algorithm with characteristics of the Heavy-ball method and prove that if g is strongly convex the convergence rate can be improved.

The Heavy-ball method is an explicit finite differences discretization of the so-called Heavy-ball with friction dynamical system:

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

It differs from the usual gradient step $x^n - \alpha \nabla f(x^n)$ by an additional inertial term $\beta(x^n - x^{n-1})$. This seemingly small difference causes a significant change in the worst-case convergence rate for the class of smooth, strongly convex functions. The precise definition of the convergence rate depends on the Lipschitz constant $L > 0$ of ∇f and the convexity parameter (of the strong convexity) $l > 0$. While the Lipschitz constant provides an upper bound for the "curvature" of the function, the convexity parameter determines a lower bound. Choosing the step size parameters as

$$\alpha = \frac{4}{(\sqrt{L} + \sqrt{l})^2}, \quad \beta = \frac{(\sqrt{L} - \sqrt{l})^2}{(\sqrt{L} + \sqrt{l})^2},$$

the convergence rate is $\mathcal{O}(((\sqrt{L} - \sqrt{l})/(\sqrt{L} + \sqrt{l}))^n)$, while the convergence rate of the gradient descent method is only $\mathcal{O}(((L-l)/(L+l))^n)$.

The conjugate gradient method has the same worst-case rate of convergence as the Heavy-ball method. In fact, the conjugate gradient method for minimizing strongly convex quadratic problems can be expressed as Heavy-ball method. Therefore, it can be seen as a special case of the Heavy-ball method for quadratic problems. Nevertheless, there is an interesting difference. On the one hand, the conjugate gradient method does not require knowledge about the convexity parameters, since the optimal step sizes are computed online. On the other hand, the conjugate gradient method does not generalize well to non-quadratic functions or constraints, whereas the Heavy-ball method does.

The Heavy-ball method shows the same (linear) convergence rate, if the function is optimized over a convex set C, which requires a projection after the update step. Denoting the projection operator by Π_C the iterations become

$$x^{n+1} = \Pi_C(x^n - \alpha \nabla f(x^n) + \beta(x^n - x^{n-1})) \,.$$

The projection can also be interpreted in terms of the proximal mapping for the (convex) indicator function δ_C of the set C. More generally, the proximal mapping for a convex function g is given as

$$\text{prox}_{\alpha g}(\hat{x}) = \arg\min_{x \in X} \frac{\|x - \hat{x}\|^2}{2} + \alpha g(x) \,,$$

and reduces to the projection onto a convex set C, if $g = \delta_C$.

This leads to the generalization of the projected Heavy-ball method, whose convergence rate is investigated in this paper,

$$x^{n+1} = \text{prox}_{\alpha g}(x^n - \alpha \nabla f(x^n) + \beta(x^n - x^{n-1})) \,.$$

The main theorem of this paper proves a linear worst-case convergence rate for this algorithm. In the general case, the rate only depends on the Lipschitz constant $L > 0$ of ∇f, the strong convexity parameter $l > 0$ of f and the strong convexity parameter $m > 0$ of g. When $g = \delta_C$ or $m = 0$, this rate coincides with the optimal rate found for the projected Heavy-ball method. Summarizing, the additional proximal step does not degrade the known convergence rates.

A particularly interesting situation arises, when g is also smooth and has Lipschitz continuous gradient ∇g with constant $M > 0$. In this case, the function $f+g$ may be split into f and g such that terms with high Lipschitz constant enter g. We demonstrate that in some situations an appropriate splitting leads to a convergence rate that is better than the optimal lower bound. This is possible thanks to the well defined structure of our optimization problem. The optimal lower bound is only valid for black-box algorithms.

A numerical experiment shows and compares the efficiency of the proposed algorithm in practice with other optimal methods. Finally, we apply iPiasco to the computer vision problems of denoising and inpainting and compare its convergence against the projected Heavy-ball method.

2 Related Work

In [21], Nesterov derives lower worst-case efficiency bounds for first-order black-box optimization on different classes of objective functions. If the objective function is non-smooth, it is shown that the lower bound on the convergence rate is $\mathcal{O}(1/\sqrt{N})$. This bound is actually attained by the subgradient method [29], which is certainly the most simple algorithm for non-smooth optimization. In [22], it was shown that if the non-smooth objective function has a certain saddle-point structure (in contrast to black box optimization), the convergence rate can be improved via smoothing to $\mathcal{O}(1/N)$. Algorithms that achieve this rate are for example [22, 8, 12, 24, 15, 16].

If the objective function is smooth (i.e. it has a Lipschitz continuous gradient) the lower worst-case efficiency bound is $\mathcal{O}(1/N^2)$. The first algorithm that achieved this rate was presented by Nesterov in [20] and generalized to composite objective functions or saddle-point problems in [8, 6, 20, 24, 15]. Finally, for the class of smooth and strongly convex functions, the lower efficiency bound is given by $\mathcal{O}(\omega^N)$, where $\omega \in (0,1)$ depends on the square root of the condition number of the objective function. Algorithms that converge with the same rate are for example [8, 27, 7, 17].

In his seminal work [27], Polyak investigates multi-step schemes to accelerate the gradient method. He assigned special interest to a certain two-step method, the Heavy-ball method. It differs from the usual gradient method by adding an inertial term. 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 Heavy-ball idea is a simple but efficient way to improve the convergence of an algorithm. Therefore, this method was investigated, generalized and modified in several ways. In [3, 2], the Heavy-ball method was extended to maximal monotone operators. In a subsequent work [19], it has been applied to a forward–backward splitting algorithm, again in the general framework of maximal monotone operators. In [4] a time second-order dynamical system related to the Heavy-ball with friction system is studied. It is shown to be equivalent to a coupled first-order system that becomes an inertial forward–backward splitting method when it is discretized. The Heavy-ball method seems also appealing in the non-convex setting. In [30], it was generalized to smooth non-convex functions and in [25] to a class of structured non-smooth non-convex optimization problems.

Modifications of the Heavy-ball method appear in the convex setting for instance in the popular accelerated gradient method of Nesterov [21, 24], or in the proximal forward–backward splitting method in [10], where the difference is the computation of the gradient. While the Heavy-ball method uses the point from the preceding iteration, Nesterov's method computes the gradient at points that are extrapolated by the inertial force. On strongly convex functions, both methods are equally fast (up to constants), but the convergence of Nesterov's accelerated gradient method can be shown to be faster on smooth convex functions [13]. Another two-step algorithm is iterative shrinkage/thresholding (TwIST) [7], which is an acceleration of IST algorithm [12]. They assume $f(x) = \frac{1}{2}\|y - Kx\|_2^2$ for a given data vector y and a linear operator K. Linear convergence is proved under the condition that K is positive definite and thus invertible. This results in f being strongly convex, and therefore is met as a special case in our formulation.

3 Preliminaries

In this paper, we consider a structured (non-smooth) strongly convex optimization problem on a finite dimensional real vector space X with inner product $\langle \cdot, \cdot \rangle$ and norm $\|\cdot\| := \sqrt{\langle \cdot, \cdot \rangle}$. The objective function $h\colon X \to \mathbb{R} \cup \{+\infty\}$ is assumed to be proper lower semi-continuous (lsc) and extended valued and bounded from below by some value $\underline{h} > -\infty$. Its *domain* is defined as $\operatorname{dom} h := \{x \in X : h(x) < +\infty\}$. Moreover, let the objective be composed of a proper lsc convex (possibly non-smooth) function $g\colon X \to \mathbb{R} \cup \{+\infty\}$ and a proper convex function $f\colon X \to R \cup \{+\infty\}$ that is twice continuously differentiable on $\operatorname{dom} g$ and has Lipschitz continuous gradient on $\operatorname{dom} g$; and let f or g be strongly convex.

$$\min_{x \in X} \; h(x) \,, \qquad h(x) := f(x) + g(x) \,, \tag{1}$$

In order to simplify notation, we define two classes of functions.

- The class $\mathscr{C}_L^{k,p}(C)$, $k, p \in \mathbb{N}$, $L > 0$, $C \subset X$ contains functions that are k-times continuously differentiable on C and the p-th derivative is Lipschitz continuous on C with Lipschitz constant L (also denoted as L-Lipschitz continuous gradient).

  Example 1. A function $f \in \mathscr{C}_L^{1,1}(C)$, $C \subset X$ is continuously differentiable on C with Lipschitz continuous gradient, i.e. there exists a constant $L > 0$

such that

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|\,, \quad \forall x, y \in C\,. \tag{2}$$

- The class of convex functions $X \to \mathbb{R} \cup \{+\infty\}$ with *convexity parameter* or *modulus* $m \ge 0$ is denoted by $\mathscr{S}_m := \mathscr{S}_m(X)$. For $m > 0$ the functions of $\mathscr{S}_m$ are strongly convex. This class is defined by the following *subgradient inequality*: It holds that $\operatorname{dom} g = \operatorname{dom} \partial g := \{x \in X : \partial g(x) \neq \emptyset\}$, and for any $y \in \operatorname{dom} \partial g$

$$g(x) \ge g(y) + \langle \xi_y, x - y \rangle + \frac{m}{2}\|x - y\|^2, \quad \forall x \in X, \forall \xi_y \in \partial g(y)\,. \tag{3}$$

- As a subclass, we define $\mathscr{S}^{1,1}_{m,M}(C) := \mathscr{S}_m \cap \mathscr{C}^{1,1}_M(C)$, $M \ge m$, $C \subset X$, which contains (strongly) convex functions with M-Lipschitz continuous first derivative on C.

In the proposed algorithm and the convergence analysis, we make use of the proximal map.

Definition 1 (Proximal map)**.** *Let $g \in \mathscr{S}_0$ be a proper lower semi-continuous convex function, $\hat{x} \in X$, and $\alpha > 0$. Then,* $\operatorname{prox}_{\alpha g}(\hat{x})$ *is the unique point in X satisfying*

$$\operatorname{prox}_{\alpha g}(\hat{x}) = \arg\min_{x \in X} \frac{\|x - \hat{x}\|^2}{2} + \alpha g(x)\,,$$

The operator $\operatorname{prox}_{\alpha g} \colon X \to X$ *is denoted as* proximal mapping*—or* proximity operator*—of g.*

Note that for a proper lower semi-continuous convex function g it holds that (see [5, Proposition 16.34])

$$\operatorname{prox}_{\alpha g} = (I + \alpha \partial g)^{-1}\,,$$

where I denotes the identity operator or identity matrix.

We collect some important (basic) properties of function in $\mathscr{C}^{1,1}_L(C)$ or $\mathscr{S}_m$ with $L > 0$, $m \ge 0$, and $C \subset X$.

- For any $y \in C$, $f \in \mathscr{C}^{1,1}_L(C)$, it holds that (see, for example, [21, Lemma 1.2.3])

$$f(x) \le f(y) + \langle \nabla f(y), x - y \rangle + \frac{L}{2}\|x - y\|^2, \quad \forall x \in C\,. \tag{4}$$

- For $g \in \mathscr{S}_m$, $m \ge 0$, and for any $x, y \in \operatorname{dom} \partial g$ we have (direct consequence from the subgradient inequality)

$$\langle x - y, \xi_x - \xi_y \rangle \ge m\|x - y\|^2, \quad \forall \xi_x \in \partial g(x), \forall \xi_y \in \partial g(y)\,.$$

- Let $g \in \mathscr{S}_m$, $m > 0$, $\alpha > 0$. Then $\alpha g \in \mathscr{S}_{\alpha m}$ and $\operatorname{prox}_{\alpha g}$ is $(1+\alpha m)$-cocoercive, meaning that for all $x, y \in X$: ([5, Proposition 23.11])

  $$\left\langle \operatorname{prox}_{\alpha g}(x) - \operatorname{prox}_{\alpha g}(y), x - y \right\rangle \geq (1+\alpha m)\|\operatorname{prox}_{\alpha g}(x) - \operatorname{prox}_{\alpha g}(y)\|^2 ,$$

  and $\operatorname{prox}_{\alpha g}$ is $(1+\alpha m)^{-1}$-Lipschitz continuous, i.e. for all $x, y \in X$:

  $$\|\operatorname{prox}_{\alpha g}(x) - \operatorname{prox}_{\alpha g}(y)\| \leq (\alpha m + 1)^{-1}\|x - y\| \,. \tag{5}$$

In the following convergence analysis the *spectral radius* of a matrix T, which is defined as the maximal magnitude of its eigenvalues

$$\rho(T) := \max\{|\lambda| : \ \lambda \text{ is eigenvalue of } T\}$$

is of importance. When the iteration matrix of an algorithm has only eigenvalues smaller than 1, i.e., the spectral radius is less than 1, the sequence induced by this iteration matrix converges. Moreover, the matrix norm reveals an asymptotic relationship to the spectral radius of the matrix. This is an important result from linear algebra (see, for example, [26, Sec. 2.1]), which was originally proved by Gelfand [14].

Lemma 1 (Gelfand's formula)**.** *It holds that $\rho(A) = \lim_{n\to\infty} \|A^n\|^{1/n}$, i.e., the spectral radius of A gives the asymptotic growth rate of $\|A^n\|$: For every $\varepsilon > 0$ there is $c = c(\varepsilon)$ such that $\|A^n\| \leq c(\rho(A)+\varepsilon)^n$ for all $n \in \mathbb{N}$.*

4 The proposed algorithm

We propose Algorithm 1 including the definition of the step size parameters, which emerge from the subsequent convergence analysis.

4.1 Convergence analysis

Let us now analyze Algorithm 1.

Lemma 2. *Let $a, b \in \mathbb{R}$. If*

$$(1-\sqrt{1-a})^2 \leq b \leq (1+\sqrt{1-a})^2 \,, \tag{7}$$

then the matrix

$$T := \begin{pmatrix} a+b & -b \\ 1 & 0 \end{pmatrix}$$

has two complex eigenvalues with squared magnitude b.

Proof. In order to obtain the eigenvalues of T, we compute the roots of the characteristic polynomial $v^2 - (a+b)v + b$ in the variable v. The eigenvalues v_1, v_2 are

$$v_{1,2} = \frac{a+b}{2} \pm \sqrt{\frac{(a+b)^2}{4} - b} \,.$$

Algorithm 1.

inertial **p**rox**i**mal **a**lgorithm for **s**trongly **c**onvex **o**ptimization (iPiasco)

- *Optimization problem*: $\min_{x\in X} h(x) = \min_{x\in X} f(x) + g(x)$ with

$$f \in \mathscr{S}^{2,1}_{l,L}(\operatorname{dom} g),\ L \geq l \geq 0\,, \quad g \in \mathscr{S}_m,\ m \geq 0, \quad m + l > 0\,.$$

- *Step-size parameter*: Define $\alpha > 0$ and $\beta \in [0,1)$ by

$$\alpha = \frac{4}{(\sqrt{l+m}+\sqrt{L+m})^2 - 4m}\,,\ \beta = \frac{(\sqrt{m+L}-\sqrt{m+l})^2}{(\sqrt{m+L}+\sqrt{m+l})^2 - 4m}$$

- *Initialization*: Choose a starting point $x^0 \in \operatorname{dom} h$ and set $x^{-1} = x^0$.
- *Iterations* ($n \geq 0$): Update

$$x^{n+1} = \operatorname{prox}_{\alpha g}(x^n - \alpha\nabla f(x^n) + \beta(x^n - x^{n-1}))\,. \tag{6}$$

Now, we try to find the condition for obtaining complex roots. The discriminant of the quadratic characteristic polynomial needs to be negative. A few elementary transformation steps yield the condition $(b + (a-2))^2 \leq 4 - 4a$, which is equivalent to (7). Under the assumption that this inequality holds T has two complex roots with squared magnitude

$$|v_{1,2}|^2 = v_{1,2}\bar{v}_{1,2} = \left(\tfrac{a+b}{2} \pm \sqrt{\tfrac{(a+b)^2}{4} - b}\right)\left(\tfrac{a+b}{2} \mp \sqrt{\tfrac{(a+b)^2}{4} - b}\right) = b\,,$$

which concludes the proof, where $\bar{v}_{1,2}$ denotes the complex conjugate. □

Now, we are equipped to start with the analysis of the worst-case convergence rate of iPiasco as proposed in Algorithm 1. The theoretical convergence rates that are discussed in this paper always refer to the worst-case performance. For notational convenience, we define $z^n := (x^n - x^*, x^{n-1} - x^*)^\top$.

Theorem 1. *Let the functions f and g with parameter l, L, and m; and the step size parameter α and β be as in Algorithm 1. Moreover, let $(x_n)_{n\in\mathbb{N}}$ be generated by Algorithm 1 and $x^* := \lim_{n\to\infty} x^n$ be the unique global optimum. Then, for every $\varepsilon > 0$ there is $c = c(\varepsilon)$ such that*

$$\|z^{n+1}\| \leq c(q+\varepsilon)^n\|z^n\| \quad \textit{for all } n \in \mathbb{N}\,, \quad \textit{where } q = \frac{\sqrt{m+L}-\sqrt{m+l}}{\sqrt{m+L}+\sqrt{m+l}}\,.$$

Proof. First, we note that $x^* \in \operatorname{dom} h$ exists and it is unique due to the properties of h. As x^* is a stationary point of h, hence $0 \in \partial h(x^*)$, x^* is a fixed point of $\operatorname{prox}_{\alpha g} \circ (I - \alpha\nabla f)$ ([5, Proposition 25.1(iv)]), i.e.,

$$x^* = \operatorname{prox}_{\alpha g}(x^* - \alpha\nabla f(x^*) + \beta(x^* - x^*))\,.$$

Combining and (6) with the $(1+\alpha m)^{-1}$-Lipschitz continuity of the (strongly) convex function αg from (5), we observe

$$\|x^{n+1}-x^*\| \le \tilde{m}_\alpha \|(1+\beta)(x^n-x^*) - \alpha(\nabla f(x^n) - \nabla f(x^*)) - \beta(x^{n-1}-x^*)\| \,,$$

where $\tilde{m}_\alpha := (1+\alpha m)^{-1}$. As f is twice continuously differentiable and $\operatorname{dom} f$ is convex, there exists a matrix B such that the following mean value theorem holds

$$\nabla f(x^n) - \nabla f(x^*) = \left(\int_0^1 B_t \, dt\right)(x^n - x^*), \quad B_t := \nabla^2 f(x^n + t(x^* - x^n))\,,$$

where $\nabla^2 f$ denotes the second derivative of f. Plugging both results together, we conclude

$$\|z^{n+1}\| \le \sup_{t\in[0,1]} \|A_t z^n\|, \quad \text{where } A_t := \begin{pmatrix} \tilde{m}_\alpha((1+\beta)I - \alpha B_t) & -\beta I \tilde{m}_\alpha \\ I & 0 \end{pmatrix} .$$

Now, we analyze the spectral radius of the matrix A_t. As the decisive fact is that the eigenvalues of B_t are in $[l, L]$, which is independent of t, from now on we drop the subscript t. Denoting the eigenvalues of B by λ_i, $i = 1, \ldots, N$, where $N \in \mathbb{N}$ is the dimension of X, it is not too difficult to show (using the right permutation matrix) that A is similar to a block diagonal matrix with blocks of size 2×2 of the form

$$T_\lambda := \begin{pmatrix} \tilde{m}_\alpha(1+\beta-\alpha\lambda) & -\tilde{m}_\alpha\beta \\ 1 & 0 \end{pmatrix} ,$$

where from now on λ stands for one of the eigenvalues λ_i. Consequently, the spectral radius of A is given by the maximal spectral radius of T_λ for all λ.

Under the assumption that the spectral radius of A is less than 1, we can show the statement of convergence using Lemma 1.

It remains to show that the setting of α and β yields $\rho(A) < 1$. If we set $a = \tilde{m}_\alpha(1-\alpha\lambda)$ and $b = \tilde{m}_\alpha\beta$ in Lemma 2, then the right inequality in (7) is trivially satisfied as $b < 1$. The left inequality in (7) requires $\rho(T_\lambda) = \sqrt{\tilde{m}_\alpha\beta} \ge |1 - \sqrt{1+\alpha\lambda\tilde{m}_\alpha - \tilde{m}_\alpha}|$. Now, the next step is to determine α and β such that (7) is met and the spectral radius becomes minimal.

Since $\lambda \in [l, L]$, we have

$$(1-\sqrt{1+\alpha\lambda\tilde{m}_\alpha - \tilde{m}_\alpha})^2 \le \max\{(1-\sqrt{1+\alpha\tilde{m}_\alpha l - \tilde{m}_\alpha})^2, (1-\sqrt{1+\alpha\tilde{m}_\alpha L - \tilde{m}_\alpha})^2\} \,.$$

Hence, we determine α such that the right hand side is minimal[1]. Then, setting β such that $\tilde{m}_\alpha\beta$ equals the right hand side (using the determined α) yields the best estimate for the convergence rate $q = \sqrt{\tilde{m}_\alpha\beta}$.

For notational convenience, we define

$$\varrho_{l,m}(\alpha) := 1 - \sqrt{1 - \frac{1-\alpha l}{1+\alpha m}} \,.$$

[1]In general, this is analytically very challenging because $\tilde{m}_\alpha$ also depends on α.

Then, we have to determine $\alpha > 0$ such that $(\varrho_{l,m}(\alpha))^2 = (\varrho_{L,m}(\alpha))^2$. It can be easily seen that $\varrho_{l,m}(\alpha) = \varrho_{L,m}(\alpha)$ is solved by $\alpha = 0$, which is not feasible. The expression $\varrho_{l,m}(\alpha) = -\varrho_{L,m}(\alpha)$ is equivalent to (note that $\alpha > 0$ and $1 + \alpha m > 0$)

$$2\sqrt{1+\alpha m} = \sqrt{\alpha}(\sqrt{l+m} + \sqrt{L+m})\,.$$

As both sides are positive, squaring and solving for α proves the expression for α. Then plugging-in the computed α verifies the term for β. □

Theorem 1 covers several special cases, which are interesting in their own right. In the following, we state some of these special cases. The proofs directly follow from Theorem 1 by plugging-in the parameters and some simple and brief calculation.

First we discuss the trivial case $L = l$. This yields $q = 0$ and the problem can be solved with a single iteration. The step size parameters are $\alpha = 1/L$ and $\beta = 0$. The function f is a simple quadratic function with circular level sets; The Lipschitz upper bound is exact. Therefore, the minimization problem

$$x^{n+1} = \arg\min_x g(x) + \frac{L}{2}\|x - (x^n - \tfrac{1}{L}\nabla f(x^n))\|^2\,,$$

which is an equivalent representation of a single iteration step of iPiasco (6), coincides with the original problem. This formulation shows that after the quadratic function f is minimized (by the step $x^n - \frac{1}{L}\nabla f(x^n)$) evaluating the proximal term solves the whole problem.

If, additionally, $m = 0$, then a single gradient descent step with step size $1/L$ solves the problem.

In the following, we consider the special case $m = 0$ and $l \leq L$. From [21], the Heavy-ball method is known as an optimal method for the class of strongly convex and two times continuously differentiable functions with Lipschitz continuous gradient. Its worst-case convergence rate coincides with the estimated optimal lower bound from Nesterov. The next corollary reveals that adding a non-smooth convex function g to the function f iPiasco still shows the same worst-case convergence rate as the Heavy-ball method. Hence, in this sense, iPiasco is optimal for the class of strongly convex (possibly non-smooth) functions.

Corollary 1. *Let the same assumptions hold as in Theorem 1, but with $l > 0$ (f strongly convex) and $m = 0$ (g only convex). The quantities α, β, q in Algorithm 1 simplify to*

$$\alpha = \frac{4}{(\sqrt{L}+\sqrt{l})^2}, \quad \beta = \left(\frac{\sqrt{L}-\sqrt{l}}{\sqrt{L}+\sqrt{l}}\right)^2, \quad \text{and} \quad q = \sqrt{\beta}\,.$$

Then, for every $\varepsilon > 0$ there is $c = c(\varepsilon)$ such that $\|z^{n+1}\| \leq c(q+\varepsilon)^n\|z^n\|$ for all $n \in \mathbb{N}$.

Remark 1. Denoting $Q_f := L/l$ as the ratio between the Lipschitz constant for ∇f and the strong convexity parameter for f, we can rewrite the spectral radius of the iteration matrix, denoted T here, for Corollary 1 as

$$\rho(T) = \frac{\sqrt{Q_f} - 1}{\sqrt{Q_f} + 1}\,,$$

which establishes a relation between the conditioning of the objective function and the convergence rate.

Remark 2. The worst-case convergence rate of the Heavy-ball method is equivalent to the worst-case rate of the conjugate gradient (CG) method. An advantage of CG is that it automatically determines the parameter α and β in each iteration. On the other hand, unlike iPiasco, CG cannot handle an additional non-smooth convex term in the objective function.

There is also an interesting new variant of Algorithm 1, where only the function g is required to be strongly convex. The convergence rate can benefit from an additional strongly convex term g.

Corollary 2. *Let the same assumptions hold as in Theorem 1, but let only g be strongly convex and f be convex, i.e., $m > 0$, $l = 0$, and $L > 0$. The quantities α, β, q in Algorithm 1 simplify to*

$$\alpha = \frac{4}{(\sqrt{m} + \sqrt{L+m})^2 - 4m}, \quad \beta = \frac{(\sqrt{m+L} - \sqrt{m})^2}{(\sqrt{m+L} + \sqrt{m})^2 - 4m},$$
$$\text{and} \quad q = \frac{\sqrt{m+L} - \sqrt{m}}{\sqrt{m+L} + \sqrt{m}}\,.$$

Then, for every $\varepsilon > 0$ there is $c = c(\varepsilon)$ such that $\|z^{n+1}\| \le c(q+\varepsilon)^n \|z^n\|$ for all $n \in \mathbb{N}$.

Summarizing, we obtain optimal linear convergence for iPiasco in the case, where f is convex and two times continuously differentiable, g is convex, and any of the two functions f or g is strongly convex, additionally.

Let us discuss the results that we obtained so far more in detail. For a moment, suppose the function $g \in \mathscr{S}^{1,1}_{m,M}$ with convexity parameter $m > 0$ and Lipschitz constant $M > 0$. Then, the function h is strongly convex, continuously differentiable and has a Lipschitz continuous gradient. The lower complexity bound from [21] reads

$$q(l, L, m, M) = \frac{\sqrt{L+M} - \sqrt{l+m}}{\sqrt{L+M} + \sqrt{l+m}}\,,$$

which is increasing whenever M is increasing ($\lim_{M\to\infty} q(l, L, m, M) = 1$). As the convergence rate for iPiasco, a value in $(0, 1)$, is independent of the Lipschitz

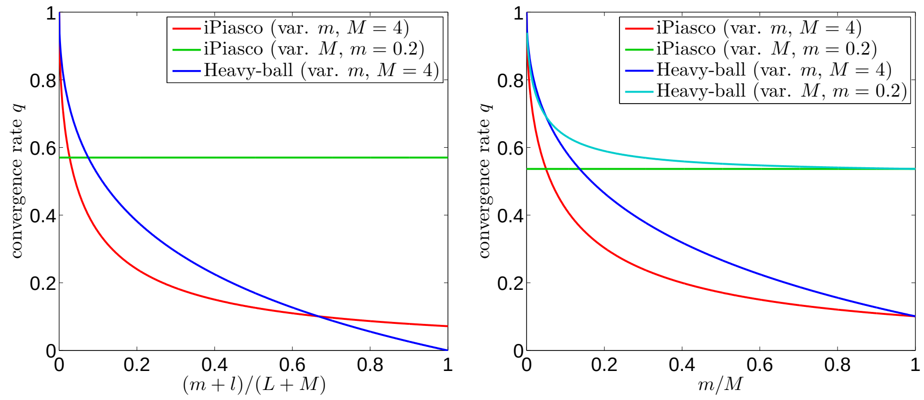
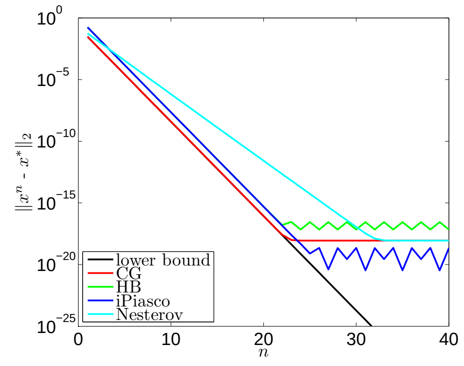


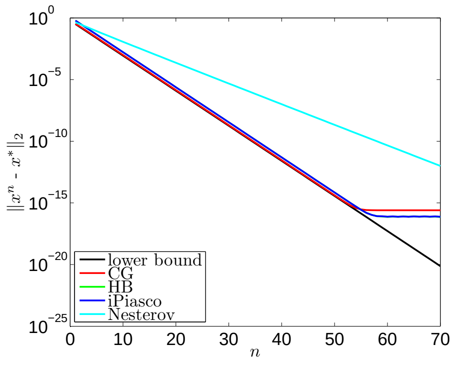
Figure 1: Convergence rates q for strongly convex and differentiable objectives with Lipschitz continuous gradient. Lower values of q are better. Let $f \in \mathscr{S}^{2,1}_{0,2}$, i.e., $l = 0$ and $L = 2$. iPiasco makes use of the splitting $h = f + g$, $g \in \mathscr{S}^{1,1}_{m,M}$, where the parameters M and m are adapted according to the x-axis, i.e., the (inverse) condition number of h (left plot) and the (inverse) condition number of g (right plot), respectively. Harder problems are closer to the left. In the case, where M is fixed and m tends to 0, the convergence rates tend to 1, because h tends to be not strongly convex. On the other hand, when m is fixed and M is high iPiasco is applicable, whereas the Heavy-ball method's rate tends to 1 (cyan line in the right plot), because the Lipschitz constant becomes unbounded. Observe, that the convergence rate for iPiasco is independent of M. Finally, for any condition numbers of g not equal to 0 or 1 the convergence rate of iPiasco outperforms the Heavy-ball method. This is impressive, as the Heavy-ball method coincides with the theoretical lower bound for this class of functions proved by Nesterov [21] for black-box algorithms.

constant of ∇g a good decomposition moves the term causing a high Lipschitz constant into the function g. It is clear that there is a certain value M for which the convergence rate of iPiasco outperforms the rate of the Heavy-ball method, see Figure 1. In other words: *iPiasco can beat the theoretical lower complexity bound from Nesterov [21] for strongly convex and twice differentiable functions with Lipschitz continuous gradient.*

At first glance, this result seems to be an error, as it is impossible to outperform the theoretical lower complexity bound. However, this bound holds true only for black-box algorithms. Here, we seemingly very efficiently explore the composition of two functions contributing good properties to the objective. Subsection 4.2.2 shows a practical example where this fact improves the performance.



(a) $l_h/L_h = 0.5$



(b) $l_h/L_h = 0.1$

Figure 2: Comparison of the linear convergence rates on Nesterov's worst-case function (8) for smooth, strongly convex problems using a problem size of "$\infty \approx 100000$". As predicted by theory, the conjugate gradient (CG) and the heavy ball (HB) method coincide with the theoretical lower bound. For larger values of l_h/L_h, iPiasco is a bit worse compared to CG and HB, but for smaller values of l_h/L_h (harder problems), iPiasco basically coincides with CG and HB. See also Figure 1. Note that Nesterov's optimal method is significantly slower.

4.2 Numerical analysis

Now, we consider the behavior of iPiasco when it is applied to Nesterov's worst-case function for smooth and strongly convex problems [21]. Let $Q_h := L_h/l_h > 1$ be the condition number for the function h with $L_h > 0$, the Lipschitz constant of ∇h, and $l_h > 0$, the convexity parameter. Then, the worst-case function from Nesterov reads

$$h_{\mathrm{wc}}(x) := \frac{l_h(Q_h - 1)}{8}\left((x_1)^2 + \sum_{i=1}^{\infty}(x_i - x_{i+1})^2 - 2x_1\right) + \frac{l_h}{2}\|x\|_2^2\,, \qquad (8)$$

where we consider here $\|\cdot\|_2 = \|\cdot\|_{\ell_2}$, the norm in the space of infinite sequences. Figure 2 presents a comparison of iPiasco to the Heavy-ball method and the conjugate gradient method based on this function. The splitting of iPiasco is by setting $g = \frac{l_h}{2}\|x\|_2^2$ and $f = h - g$. The resulting parameters for estimating the convergence rate are $L = L_h$, $m = l_h$, and $l = 0$. The parameters for the Heavy-ball method (as a special instance of iPiasco) are $m = 0$, $L = L_h$, and $l = l_h$.

Since our algorithm can basically deal with the same class of composite objective functions as for example studied in [24], iPiasco can be seen as an improvement of Nesterov's optimal gradient method for minimizing strongly convex composite objective functions.

4.2.1 The dual Huber–ROF model

In this subsection, we consider the dual problem of the Huber norm regularized variant of the Rudin–Osher–Fatemi model [28] for image denoising. We define the decomposition

$$f(p) = \frac{1}{2}\|\nabla^\top p - \lambda u^0\|_2^2, \quad g(p) = \frac{\varepsilon}{2}\|p\|_2^2 + \delta_{\{\|p\|_\infty \le 1\}}(p)\,, \tag{9}$$

where $p \in \mathbb{R}^{2N}$ is the dual vector for the ROF problem, $u^0 \in \mathbb{R}^N$ is the noisy input image, $\lambda, \varepsilon > 0$, $\delta_{\{\|p\|_\infty \le 1\}}$ denotes the indicator function of the set $\{\|p\|_\infty \le 1\}$, and $\nabla^\top$ denotes the transposed of the gradient operator. Obviously, this model is strongly convex with modulus ε. In our decomposition, g is strongly convex and f is two times continuously differentiable. The decomposition defined above is suitable for iPiasco, which yields an optimal worst-case convergence rate for this class of problems. If we shift the quadratic penalty term $\frac{\varepsilon}{2}\|p\|_2^2$ from function g to the function f, we obtain another variant of iPiasco, which for the sake of discrimination is denoted iPiasco (projected). There is no best choice for decomposing the dual Huber–ROF model. This is explained by the term $\frac{\varepsilon}{2}\|p\|_2^2$: the Lipschitz constant and the convexity parameter are the same (cf. Figure 1).

We compare our method against the optimal methods: TwIST [7] and Primal–Dual [9] (in this work a slightly better convergence rate for the primal–dual algorithm in [8] for strongly convex and smooth problems is proved). Moreover, we compare against the linearly converging Algorithm 2.2.11 from [21], and the (regarding the convergence rate for this problem) suboptimal methods FISTA [2] [6], a very recent algorithm with unknown convergence rate IFB [4] (parameters: $a_{\text{IFB}} = b_{\text{IFB}} = L/4$ and $\lambda_{\text{IFB}} = 4/L$; L is the Lipschitz constant of ∇f), the primal–dual Algorithm 3.5 from [11] (denoted Primal–Dual-CDV10, with parameters: $\gamma_{\text{CDV10}} = 2/L$ and $\lambda_{\text{CDV10}} = 0.75$), and the proximal forward–backward splitting method in [10] (denoted ProxFB with parameter $\gamma_{\text{ProxFB}} = 1.99/L$ and $\lambda_{\text{ProxFB}} = 0.99$).

Note that the differences in the computational cost between all methods for one iteration are negligible. Also regarding storage, the methods are comparable: iPiasco, iPiasco (projected), TwIST, Nesterov's method, and FISTA require 3 optimization variables of dimension $2N$; IFB and ProxFB require 2 variables in $\mathbb{R}^{2N}$; Primal–Dual-CDV10 requires 2 variables in $\mathbb{R}^{2N}$ and 1 in $\mathbb{R}^N$; Primal–Dual [9] requires 1 variable in $\mathbb{R}^{2N}$ and 2 in $\mathbb{R}^N$; the cost for the derivative operator, input variables and parameters are the same for all algorithms. For 2D and 3D image processing tasks, storage is usually not a problem, but for huge scale machine learning problems, where even the evaluation of a single gradient is a problem (see for example [23]), the additional storage of inertial methods might become an issue.

[2] Although FISTA is an accelerated method, a comparison is not completely fair since it does not exploit the strong convexity of the problem.

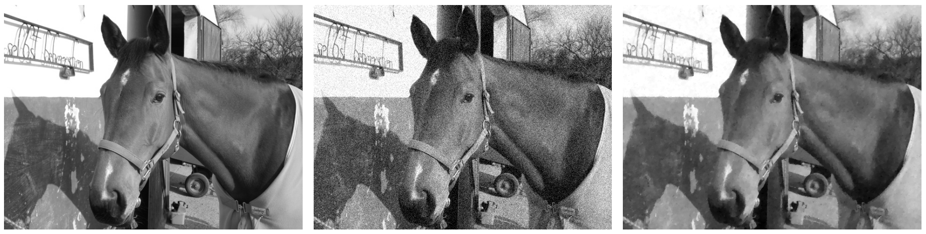


Figure 3: A denoising example with the dual Huber–ROF model (9). The objective function is strongly convex and the result on the right is obtained using optimal methods for this class of problems. Among these methods are two versions of iPiasco arising from different decompositions of the objective and a scheme from Nesterov.

	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}	10^{-12}	10^{-14}
iPiasco	25	48	71	92	114	135	157
iPiasco (proj.)	25	48	71	92	114	135	157
TwIST	28	52	74	96	117	139	160
Primal-Dual	36	78	120	162	dnc	dnc	dnc
Nesterov	36	79	121	162	dnc	dnc	dnc
Primal-Dual-CDV10	58	148	dnc	dnc	dnc	dnc	dnc
FISTA	52	dnc	dnc	dnc	dnc	dnc	dnc
ProxFB	106	dnc	dnc	dnc	dnc	dnc	dnc
IFB	108	dnc	dnc	dnc	dnc	dnc	dnc

Table 1: Table corresponding to Figure 4. The first row shows error thresholds (for the normalized error). The entries in the table show the number of required iterations to fall below the respective error threshold. "dnc" means that the threshold was not reached within 200 iterations. The optimal methods iPiasco, TwIST, and iPiasco (proj.) clearly perform best.

In the following experiment we set $\lambda = 0.05$ and $\varepsilon = 0.1$, which yields a good denoising quality. See Figure 3 for the result of this problem. The parameters of all algorithms were chosen to perform best for these λ, ε.

The theoretical estimates for the linear convergence rates are 0.8 for iPiasco, iPiasco (proj.), and TwIST, 0.943 for Nesterov's algorithm, and 0.894 for Primal–Dual [9]. Figure 4 and Table 1 show that the convergence of iPiasco (proj.), iPiasco, and TwIST are practically the same. This result also highlights the importance of (optimal) convergence rates. The linearly converging algorithm from Nesterov and Primal–Dual [9] perform equally well, but significantly worse than iPiasco. The methods FISTA, IFB, and Primal–Dual–CDV10 from [11] are clearly outperformed; They have suboptimal rates of convergence.

In order to base our conclusions on a larger experimental evidence, we performed additional experiments with different parameters λ and ε. The results gave us essentially the same conclusion about the performance of the different

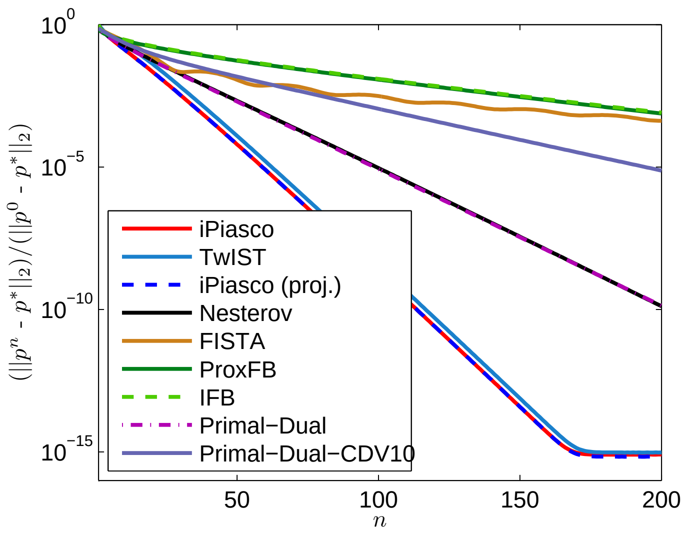


Figure 4: The three optimal methods iPiasco, iPiasco (proj.), and TwIST perform best on the dual Huber–ROF model (9).

algorithms.

4.2.2 An inpainting problem

Recently, [18] has shown that inpainting by a linear diffusion model is able to compete with jpeg for lossy image compression, particularly for cartoon like images. We consider an approximation of their reconstruction step defined by the decomposition

$$f(u) = \frac{1}{2}\|\nabla u\|_2^2, \quad g(u) = \frac{1}{2}\lambda\|c \cdot (u - u^0)\|_2^2 + \frac{1}{2}\varepsilon\|u\|_2^2\,, \tag{10}$$

where "$\cdot$" denotes the coordinate-wise product, $u, c, u^0 \in \mathbb{R}^N$, and $c_i = 1$ if pixel i of the original image u_i^0 was stored and $c_i = 0$ otherwise, i.e. the original pixel value u_i^0 is known only if $c_i = 1$. The linear operator implements a discrete gradient operator. The positive parameter λ allows for denoising the original pixel values, and $\varepsilon > 0$ is a small numerical number to make the problem strongly convex. This is a very difficult problem, which is related to Nesterov's worst-case function.

Using the proposed decomposition (10) the Lipschitz constant of the gradient of f is $L = 8$, the strong convexity parameter is $l = 0$, for g the Lipschitz constant is $M = \lambda + \varepsilon$ and the strong convexity parameter is $m = \varepsilon$. As the (inverse) condition number of g is smaller than 1, the convergence rate of iPiasco is expected to be better than the optimal rate of the Heavy-ball method (cf. Figure 1). In fact, by setting $\varepsilon = 10^{-4}$, $\lambda = 10$ the convergence rate for the Heavy-ball method is 0.995297, whereas the rate for our iPiasco is 0.992954.

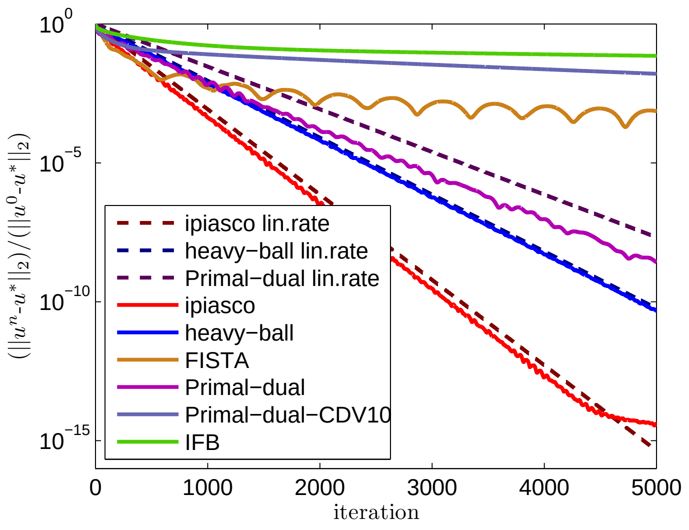


Figure 5: Convergence of the Heavy-ball method and iPiasco for the inpainting problem (10). The theoretical and practical convergence of iPiasco is much better then the convergence of the Heavy-ball method. Note that the worst-case convergence rate of the Heavy-ball method coincides with the optimal convergence rate determined by Nesterov [21] for the general class of smooth strongly convex objectives. Since we exploit the structure of the inpainting problem very well, we can achieve a better convergence rate. Several other methods are clearly outperformed.

The parameters ε and λ are chosen for a good reconstruction quality. The convergence of both methods is shown in Figure 5 and Table 2, together with the several other methods mentioned in the following. The inpainting result is shown in Figure 6.

We compare also against other methods with no known or worse convergence rate. We evaluate the linearly converging primal–dual algorithm (Primal–Dual) from [9], FISTA [6], IFB [4], the primal–dual algorithm (Primal–Dual–CDV10) from [11], and the proximal forward–backward splitting method (ProxFB) from [10] with the same parameters as in the preceding subsection. Figure 5 does not show ProxFB as it performed almost identical to IFB. As discussed for the dual Huber–ROF denoising problem, the differences in computational cost and storage consumption of these methods are negligible.

Not only the theoretical rate of convergence for iPiasco is better than the one for the Heavy-ball method and the other methods, but also the practical convergence is significantly better. Note that increasing the parameter λ the convergence rate for the Heavy-ball becomes worse, whereas the convergence rate of iPiasco is independent of λ. For $\lambda \to \infty$ the Heavy-ball method can only be applied by explicitly handling the constraints $u_i = u_i^0$ for $c_i = 1$ as boundary

	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}	10^{-12}	10^{-14}
iPiasco	553	1202	1850	2496	3148	3801	4459
Heavy-ball	924	1909	2880	3851	4838	dnc	dnc
Primal-dual	951	2122	3390	4595	dnc	dnc	dnc
FISTA	578	dnc	dnc	dnc	dnc	dnc	dnc
Primal-dual-CDV10	dnc	dnc	dnc	dnc	dnc	dnc	dnc
IFB	dnc	dnc	dnc	dnc	dnc	dnc	dnc
ProxFB	dnc	dnc	dnc	dnc	dnc	dnc	dnc

Table 2: Table corresponding to Figure 5. The first row shows error thresholds (for the normalized error). The entries in the table show the number of required iterations to fall below the respective error threshold. “dnc” means that the threshold was not reached within 5000 iterations. Our method iPiasco clearly outperforms all other methods.

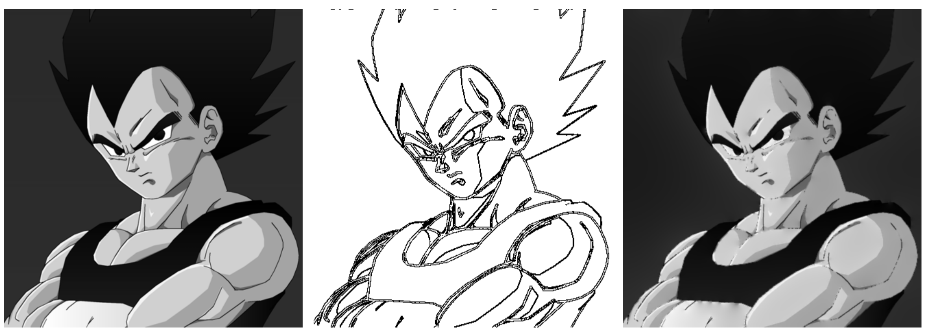


Figure 6: An inpainting example [1] like in the reconstruction step of the image compression method [18]. The pixel values of the original image on the left were stored only at the black pixels of the inpainting mask in the middle image. The image on the right is generated by solving the inpainting problem (10).

conditions in the gradient operator, whereas iPiasco can be applied directly.

5 Conclusions

In this paper, we have considered an algorithm, which we termed iPiasco, for solving a strongly convex optimization problem. The algorithm was shown to benefit from the additional structure that we assume from the optimization problem, namely its being a sum of two convex functions, a smooth one and a non-smooth one. iPiasco combines characteristics of the Heavy-ball method and forward–backward splitting.

In the convergence analysis, we proved the worst-case rate of convergence linear. The worst-case rate of convergence is given by a simple expression and is achieved by the proposed step size parameters. For special cases of iPiasco (the

Heavy-ball or projected Heavy-ball method) the rate of convergence coincides with the known rates of convergence, which are provably optimal. Moreover, there are problems, where a well chosen splitting of the objective function, leads to even better worst-case convergence rates than the optimal ones proved for smooth strongly convex functions.

Numerical experiments have confirmed the theoretical results that have been proved, and highlight the advantage of our algorithm over existing methods. The practical relevance of iPiasco has been demonstrated on image processing problems, where it achieves the best efficiency in the comparison.

An interesting and challenging problem is to obtain a performance estimate which allows for errors in the evaluation of the gradient or the proximal map. Furthermore it will be interesting to study algorithms which automatically adapt the step size parameters α and β to the problem. These problems will be addressed in future work.

6 Acknowledgements

Thomas Pock acknowledges support from the Austrian science fund (FWF) under the START project *BIVISION*, No. Y729. Peter Ochs and Thomas Brox acknowledge funding by the German Research Foundation (DFG grant BR 3815/5-1).

References

[1] Vegeta from Dragon Ball Z. `http://1.bp.blogspot.com/-g3wpzDWE0QI/UbycqlJWjDI/AAAAAAAAAbU/ty0EZl0kqXw/s1600/VEGETA%2B1.jpg`.

[2] F. Alvarez. Weak convergence of a relaxed and inertial hybrid projection-proximal point algorithm for maximal monotone operators in Hilbert space. *SIAM Journal on Optimization*, 14(3):773–782, 2003.

[3] F. Alvarez and H. Attouch. An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping. *Set-Valued Analysis*, 9(1-2):3–11, 2001.

[4] H. Attouch, J. Peypouquet, and P. Redont. A dynamical approach to an inertial forward-backward algorithm for convex minimization. *SIAM Journal on Optimization*, 24(1):232–256, 2014.

[5] H.H. Bauschke and P.L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer, 2011.

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

[7] J.M. Bioucas-Dias and M. Figueiredo. A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image Processing*, 16(12):2992–3004, 2007.

[8] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[9] A. Chambolle and T. Pock. On the ergodic convergence rates of a first-order primal-dual algorithm. 2014. to appear.

[10] P. L. Combettes and V. R. Wajs. Signal Recovery by Proximal Forward-Backward Splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[11] P.L. Combettes, D. Dũng, and B.C. Vũ. Dualization of signal recovery problems. *Set-Valued and Variational Analysis*, 18(3-4):373–404, 2010.

[12] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure Appl. Math.*, 57(11):1413–1457, 2004.

[13] Y. Drori and M. Teboulle. Performance of first-order methods for smooth convex minimization: a novel approach. *Mathematical Programming*, 145(1-2):451–482, 2014.

[14] I. Gelfand. Normierte Ringe. *Rec. Math. [Mat. Sbornik] N.S.*, 9(51):3–24, 1941.

[15] D. Goldfarb and S. Ma. Fast multiple-splitting algorithms for convex optimization. *SIAM Journal on Optimization*, 22(2):533–556, 2012.

[16] B. He and X. Yuan. On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012.

[17] M. Hong and Z.-Q. Luo. On the linear convergence of the alternating direction method of multipliers. *ArXiv e-prints*, August 2012.

[18] M. Mainberger and J. Weickert. Edge-based image compression with homogeneous diffusion. In Xiaoyi Jiang and Nicolai Petkov, editors, *Computer Analysis of Images and Patterns*, volume 5702 of *Lecture Notes in Computer Science*, pages 476–483. Springer Berlin Heidelberg, 2009.

[19] A. Moudafi and M. Oliny. Convergence of a splitting inertial proximal method for monotone operators. *Journal of Computational and Applied Mathematics*, 155:447–454, 2003.

[20] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27:372–376, 1983.

[21] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87 of *Applied Optimization*. Kluwer Academic Publishers, Boston, MA, 2004.

[22] Y. Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, May 2005.

[23] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

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

[25] P. Ochs, Y. Chen, T. Brox, and T. Pock. ipiano: Inertial proximal algorithm for non-convex optimization. *SIAM Journal on Imaging Sciences*, 7(2):1388–1419, 2014.

[26] B. T. Poljak. *Introduction to optimization.* Optimization Software, 1987.

[27] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[28] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[29] N. Z. Shor. *Minimization Methods for Non-differentiable Functions.* Springer-Verlag New York, Inc., New York, NY, USA, 1985.

[30] S.K. Zavriev and F.V. Kostyuk. Heavy-ball method in nonconvex optimization problems. *Computational Mathematics and Modeling*, 4(4):336–341, 1993.