Implementation of an optimal first-order method for strongly convex total variation regularization

Jensen, Tobias Lindstrøm; Jørgensen, Jakob Heide; Hansen, Per Christian; Jensen, Søren Holdt

Published in:
BIT Numerical Mathematics

DOI (link to publication from Publisher):
10.1007/s10543-011-0359-8

Publication date:
2012

Document Version
Publisher's PDF, also known as Version of record

BIT Numer Math (2012) 52:329–356
DOI 10.1007/s10543-011-0359-8

BIT

Implementation of an optimal first-order method for strongly convex total variation regularization

T.L. Jensen · J.H. Jørgensen · P.C. Hansen · S.H. Jensen

Received: 7 October 2010 / Accepted: 29 August 2011 / Published online: 24 September 2011

Abstract We present a practical implementation of an optimal first-order method, due to Nesterov, for large-scale total variation regularization in tomographic reconstruction, image deblurring, etc. The algorithm applies to μ-strongly convex objective functions with L-Lipschitz continuous gradient. In the framework of Nesterov both μ and L are assumed known—an assumption that is seldom satisfied in practice. We propose to incorporate mechanisms to estimate locally sufficient μ and L during the iterations. The mechanisms also allow for the application to non-strongly convex functions. We discuss the convergence rate and iteration complexity of several first-order methods, including the proposed algorithm, and we use a 3D tomography problem to compare the performance of these methods. In numerical simulations we demonstrate the advantage in terms of faster convergence when estimating the strong convexity parameter μ for solving ill-conditioned problems to high accuracy, in com-

Communicated by Erkki Somersalo.

This work is part of the project CSI: Computational Science in Imaging, supported by grant no. 274-07-0065 from the Danish Research Council for Technology and Production Sciences.

T.L. Jensen · S.H. Jensen
Department of Electronic Systems, Aalborg University, Niels Jernesvej 12, 9220 Aalborg Ø, Denmark

T.L. Jensen
e-mail: tlj@es.aau.dk

S.H. Jensen
e-mail: shj@es.aau.dk

J.H. Jørgensen · P.C. Hansen (✉)
Department of Informatics and Mathematical Modelling, Technical University of Denmark, Building 321, 2800 Lyngby, Denmark
e-mail: pch@imm.dtu.dk

J.H. Jørgensen
e-mail: jakj@imm.dtu.dk

parison with an optimal method for non-strongly convex problems and a first-order method with Barzilai-Borwein step size selection.

Keywords Optimal first-order optimization methods · Strong convexity · Total variation regularization · Tomography

Mathematics Subject Classification (2000) 65K10 · 65R32

1 Introduction

Large-scale discretizations of inverse problems [22] arise in a variety of applications such as medical imaging, non-destructive testing, and geoscience. Due to the inherent instability of these problems, it is necessary to apply regularization in order to compute meaningful reconstructions, and this work focuses on the use of total variation which is a powerful technique when the sought solution is required to have sharp edges (see, e.g., [14, 36] for applications in image reconstruction).

Many total variation algorithms have already been developed, including time marching [36], fixed-point iteration [40], and various minimization-based methods such as sub-gradient methods [1, 15], interior-point methods for second-order cone programming (SOCP) [20], methods exploiting duality [10, 13, 24], and graph-cut methods [11, 18].

The numerical difficulty of a problem depends on the linear forward operator. Most methods are dedicated either to denoising, where the operator is simply the identity, or to deblurring where the operator is represented by a fast transform. For general linear operators with no exploitable matrix structure, such as in tomographic reconstruction, the selection of algorithms is not as large. Furthermore, the systems that arise in real-world tomography applications, especially in 3D, are so large that memory-requirements preclude the use of second-order methods with quadratic convergence.

Recently, Nesterov's optimal first-order method [30, 31] has been adapted to, and analyzed for, a number of imaging problems [16, 41]. In [41] it is shown that Nesterov's method outperforms standard first-order methods by an order of magnitude, but this analysis does not cover tomography problems. A drawback of Nesterov's algorithm (see, e.g., [12]) is the explicit need for the strong convexity parameter and the Lipschitz constant of the objective function, both of which are generally not available in practice.

This paper describes a practical implementation of Nesterov's algorithm, augmented with efficient heuristic methods to estimate the unknown Lipschitz constant and strong convexity parameter. The Lipschitz constant is handled using backtracking, similar to the technique used in [4]. To estimate the unknown strong convexity parameter—which is more difficult—we propose a heuristic based on adjusting an estimate of the strong convexity parameter using a local strong convexity inequality. Furthermore, we equip the heuristic with a restart procedure to ensure convergence in case of an inadequate estimate.

We call the algorithm *UPN* (Unknown Parameter Nesterov) and compare it with two versions of the well-known gradient projection algorithm; *GP*: a simple version

using a backtracking line search for the stepsize and *GPBB*: a more advanced version using Barzilai-Borwein stepsize selection [2] and the nonmonotone backtracking procedure from [21].

We also compare with a variant of the proposed algorithm, UPN_0, where the strong convexity information is not enforced. UPN_0 is optimal among first-order methods for the class of Lipschitz smooth, convex (but *not* strongly convex) functions. There are several other variants of optimal first-order methods for Lipschitz smooth problems, see, e.g., [4, 7, 27, 29–32, 38] and the overview in [6, 38], but they all share similar practical convergence [6, §6.1]. We therefore consider UPN_0 to represent this class of methods. We have implemented the four algorithms in C with a MEX interface to MATLAB, and the software is available from www.imm.dtu.dk/~pch/TVReg/.

Our numerical tests demonstrate that the proposed method *UPN* is significantly faster than *GP*, as fast as *GPBB* for moderately ill-conditioned problems, and significantly faster for ill-conditioned problems. Compared to UPN_0, *UPN* is consistently faster, when solving to high accuracy.

We start with introductions to the discrete total variation problem, to smooth and strongly convex functions, and to some basic first-order methods in Sects. 2, 3, and 4, respectively. Section 5 introduces important inequalities while the new algorithm is described in Sect. 6. Finally, in Sect. 7 we report our numerical experiments with the proposed method applied to an image deblurring problem and a tomographic reconstruction problem.

Throughout the paper we use the following notation. The smallest singular value of a matrix A is denoted $\sigma_{\min}(A)$. The smallest and largest eigenvalues of a symmetric semi-definite matrix M are denoted by $\lambda_{\min}(M)$ and $\lambda_{\max}(M)$. For an optimization problem, f is the objective function, $x^\star$ denotes a minimizer, $f^\star = f(x^\star)$ is the optimum objective, and x is called an ϵ-suboptimal solution if $f(x) - f^\star \leq \epsilon$.

2 The discrete total variation reconstruction problem

The Total Variation (TV) of a real function $\mathcal{X}(t)$ with $t \in \Omega \subset \mathbb{R}^p$ is defined as

$$\mathcal{T}(\mathcal{X}) = \int_\Omega \|\nabla \mathcal{X}(t)\|_2 \, dt. \tag{2.1}$$

Note that the Euclidean norm is not squared, which means that $\mathcal{T}(\mathcal{X})$ is non-differentiable. In order to handle this we consider a smoothed version of the TV functional. Two common choices are to replace the Euclidean norm of the vector z by either $(\|z\|_2^2 + \beta^2)^{1/2}$ or the Huber function

$$\Phi_\tau(z) = \begin{cases} \|z\|_2 - \frac{1}{2}\tau, & \text{if } \|z\|_2 \geq \tau, \\ \frac{1}{2\tau}\|z\|_2^2, & \text{else.} \end{cases} \tag{2.2}$$

In this work we use the latter, which can be considered a prox-function smoothing [31] of the TV functional [5]; thus, the approximated TV functional is given by

$$\mathcal{T}_\tau(\mathcal{X}) = \int_\Omega \Phi_\tau(\nabla \mathcal{X}) \, dt. \tag{2.3}$$

In this work we consider the case $t \in \mathbb{R}^3$. To obtain a discrete version of the TV reconstruction problem, we represent $\mathcal{X}(t)$ by an $N = m \times n \times l$ array X, and we let $x = \text{vec}(X)$. Each element or voxel of the array X, with index j, has an associated matrix (a discrete differential operator) $D_j \in \mathbb{R}^{3\times N}$ such that the vector $D_j\, x \in \mathbb{R}^3$ is the forward difference approximation to the gradient at x_j. By stacking all D_j we obtain the matrix D of dimensions $3N \times N$:

$$D = \begin{pmatrix} D_1 \\ \vdots \\ D_N \end{pmatrix}. \tag{2.4}$$

We use periodic boundary conditions in D, which ensures that only a constant x has a TV of 0. Other choices of boundary conditions could easily be implemented.

When the discrete approximation to the gradient is used and the integration in (2.3) is replaced by summations, the discrete and smoothed TV function is given by

$$T_\tau(x) = \sum_{j=1}^{N} \Phi_\tau(D_j x). \tag{2.5}$$

The gradient $\nabla T_\tau(x) \in \mathbb{R}^N$ of this function is given by

$$\nabla T_\tau(x) = \sum_{j=1}^{N} D_j^T D_j\, x / \max\{\tau, \|D_j x\|_2\}. \tag{2.6}$$

We assume that the sought reconstruction has voxel values in the range $[0, 1]$, so we wish to solve a bound-constrained problem, i.e., having the feasible region $\mathcal{Q} = \{x \in \mathbb{R}^N \mid 0 \le x_j \le 1, \forall j\}$. Given a linear system $A\, x \approx b$ where $A \in \mathbb{R}^{M\times N}$ and $N = mnl$, we define the associated *discrete TV regularization problem* as

$$x^\star = \underset{x\in\mathcal{Q}}{\text{argmin}}\ \phi(x), \quad \phi(x) = \frac{1}{2}\|A\, x - b\|_2^2 + \alpha\, T_\tau(x), \tag{2.7}$$

where $\alpha > 0$ is the TV regularization parameter. This is the problem we want to solve, for the case where the linear system of equations arises from discretization of an inverse problem.

3 Smooth and strongly convex functions

To set the stage for the algorithm development in this paper, we consider the convex optimization problem $\min_{x\in\mathcal{Q}} f(x)$ where f is a convex function and $\mathcal{Q}$ is a convex set. We recall that a continuously differentiable function f is convex if

$$f(x) \ge f(y) + \nabla f(y)^T (x - y), \quad \forall x, y \in \mathbb{R}^N. \tag{3.1}$$

Definition 3.1 A continuously differentiable convex function f is said to be *strongly convex* with *strong convexity parameter* μ if there exists a $\mu > 0$ such that

$$f(x) \geq f(y) + \nabla f(y)^T (x - y) + \frac{1}{2}\mu \|x - y\|_2^2, \quad \forall x, y \in \mathbb{R}^N. \tag{3.2}$$

Definition 3.2 A continuously differentiable convex function f has *Lipschitz continuous* gradient with *Lipschitz constant* L, if

$$f(x) \leq f(y) + \nabla f(y)^T (x - y) + \frac{1}{2}L \|x - y\|_2^2, \quad \forall x, y \in \mathbb{R}^N. \tag{3.3}$$

Remark 3.1 The condition (3.3) is equivalent [30, Theorem 2.1.5] to the more standard way of defining Lipschitz continuity of the gradient, namely, through convexity and the condition $\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2, \forall x, y \in \mathbb{R}^N$.

Remark 3.2 Lipschitz continuity of the gradient is a smoothness requirement on f. A function f that satisfies (3.3) is said to be smooth, and L is also known as the *smoothness constant*.

The set of functions that satisfy (3.2) and (3.3) is denoted $\mathcal{F}_{\mu,L}$. It is clear that $\mu \leq L$ and also that if $\mu_1 \geq \mu_0$ and $L_1 \leq L_0$ then $f \in \mathcal{F}_{\mu_1,L_1} \Rightarrow f \in \mathcal{F}_{\mu_0,L_0}$. Given fixed choices of μ and L, we introduce the ratio $Q = L/\mu$ (sometimes referred to as the "modulus of strong convexity" [28] or the "condition number for f" [30]) which is an upper bound for the condition number of the Hessian matrix. The number Q plays a major role for the convergence rate of the methods we will consider.

Lemma 3.1 *For the quadratic function* $f(x) = \frac{1}{2}\|Ax - b\|_2^2$ *with* $A \in \mathbb{R}^{M\times N}$ *we have*

$$L = \|A\|_2^2, \qquad \mu = \lambda_{\min}\left(A^T A\right) = \begin{cases} \sigma_{\min}(A)^2, & \text{if } \operatorname{rank}(A) = N, \\ 0, & \text{else}, \end{cases} \tag{3.4}$$

and if $\operatorname{rank}(A) = N$ *then* $Q = \kappa(A)^2$, *the square of the condition number of* A.

Proof Follows from $f(x) = f(y) + (Ay - b)^T A(x - y) + \frac{1}{2}(x - y)^T A^T A(x - y)$, the second order Taylor expansion of f about y, where equality holds for quadratic f. □

Lemma 3.2 *For the smoothed TV function* (2.5) *we have*

$$L = \|D\|_2^2/\tau, \qquad \mu = 0, \tag{3.5}$$

where $\|D\|_2^2 \leq 12$ *in the 3D case.*

Proof The result for L follows from [31, Theorem 1] since the smoothed TV functional can be written as [5, 16]

$$T_\tau(x) = \max_u \left\{ u^T Dx - \frac{\tau}{2}\|u\|_2^2 : \|u_i\|_2 \leq 1, \ \forall i = 1, \ldots, N \right\}$$

with $u = (u_1^T, \ldots, u_N^T)^T$ stacked according to D. The inequality $\|D\|_2^2 \leq 12$ follows from a straightforward extension of the proof in the Appendix of [16]. For μ pick $y = \alpha e \in \mathbb{R}^N$ and $x = \beta e \in \mathbb{R}^N$, where $e = (1, \ldots, 1)^T$, and $\alpha \neq \beta \in \mathbb{R}$. Then we get $T_\tau(x) = T_\tau(y) = 0$, $\nabla T_\tau(y) = 0$ and obtain

$$\frac{1}{2}\mu\|x - y\|_2^2 \leq T_\tau(x) - T_\tau(y) - \nabla T_\tau(y)^T(x - y) = 0,$$

and hence $\mu = 0$. □

Theorem 3.1 *For the function $\phi(x)$ defined in (2.7) we have a strong convexity parameter $\mu = \lambda_{\min}(A^T A)$ and Lipschitz constant $L = \|A\|_2^2 + \alpha\|D\|_2^2/\tau$. If $\operatorname{rank}(A) < N$ then $\mu = 0$, otherwise $\mu = \sigma_{\min}(A)^2 > 0$ and*

$$Q = \kappa(A)^2 + \frac{\alpha}{\tau}\frac{\|D\|_2^2}{\sigma_{\min}(A)^2}, \tag{3.6}$$

where $\kappa(A) = \|A\|_2/\sigma_{\min}(A)$ is the condition number of A.

Proof Assume $\operatorname{rank}(A) = N$ and consider $f(x) = g(x) + h(x)$ with $g \in \mathcal{F}_{\mu_g, L_g}$ and $h \in \mathcal{F}_{\mu_h, L_h}$. Then $f \in \mathcal{F}_{\mu_f, L_f}$, where $\mu_f = \mu_g + \mu_h$ and $L_f = L_g + L_h$. From μ_f and L_f and using Lemmas 3.1 and 3.2 with $g(x) = \frac{1}{2}\|Ax - b\|_2^2$ and $h(x) = \alpha T_\tau(x)$ we obtain the condition number for ϕ given in (3.6). If $\operatorname{rank}(A) < N$ then the matrix $A^T A$ has at least one zero eigenvalue, and thus $\mu = 0$. □

Remark 3.3 Due to the inequalities used to derive (3.6), there is no guarantee that the given μ and L are the tightest possible for ϕ.

4 Some basic first-order methods

A basic first-order method is the gradient projection method of the form

$$x^{(k+1)} = P_{\mathcal{Q}}\big(x^{(k)} - p_k \nabla f\big(x^{(k)}\big)\big), \quad k = 0, 1, 2, \ldots, \tag{4.1}$$

where $P_{\mathcal{Q}}$ is the Euclidean projection onto the convex set $\mathcal{Q}$ [30]. The following theorem summarizes the convergence properties.

Theorem 4.1 *Let $f \in \mathcal{F}_{\mu,L}$, $p_k = 1/L$ and $x^\star \in \mathcal{Q}$ be the constrained minimizer of f, then for the gradient projection method (4.1) we have*

$$f\big(x^{(k)}\big) - f^\star \leq \frac{L}{2k}\|x^{(0)} - x^\star\|_2^2. \tag{4.2}$$

Moreover, if $\mu \neq 0$ then

$$f\big(x^{(k)}\big) - f^\star \leq \left(1 - \frac{\mu}{L}\right)^k \big(f\big(x^{(0)}\big) - f^\star\big). \tag{4.3}$$

Proof The two bounds follow from [39] and [28, §7.1.4], respectively. □

To improve the convergence of the gradient (projection) method, Barzilai and Borwein [2] suggested a scheme in which the step $p_k \nabla f(x^{(k)})$ provides a simple and computationally cheap approximation to the Newton step $(\nabla^2 f(x^{(k)}))^{-1} \nabla f(x^{(k)})$. For general unconstrained problems with $f \in \mathcal{F}_{\mu,L}$, possibly with $\mu = 0$, non-monotone line search combined with the Barzilai-Borwein (BB) strategy produces algorithms that converge [35]; but it is difficult to give a precise iteration complexity for such algorithms. For strictly quadratic unconstrained problems the BB strategy requires $\mathcal{O}(Q \log \epsilon^{-1})$ iterations to obtain an ϵ-suboptimal solution [17]. In [19] it was argued that, in practice, $\mathcal{O}(Q \log \epsilon^{-1})$ iterations "is the best that could be expected". This comment is also supported by the statement in [30, p. 69] that all "reasonable step-size rules" have the same iteration complexity as the standard gradient method. Note that the classic gradient method (4.1) has $\mathcal{O}(L/\epsilon)$ complexity for $f \in \mathcal{F}_{0,L}$. To summarize, when using the BB strategy we should not expect better complexity than $\mathcal{O}(L/\epsilon)$ for $f \in \mathcal{F}_{0,L}$, and $\mathcal{O}(Q \log \epsilon^{-1})$ for $f \in \mathcal{F}_{\mu,L}$.

In Algorithm 1 we give the (conceptual) algorithm *GPBB*, which implements the BB strategy with non-monotone line search [8, 42] using the backtracking procedure from [21] (initially combined in [35]). The algorithm needs the real parameter $\sigma \in [0, 1]$ and the nonnegative integer K, the latter specifies the number of iterations over which an objective decrease is guaranteed.

An alternative approach is to consider first-order methods with optimal complexity. The optimal complexity is defined as the worst-case complexity for a first-order method applied to any problem in a certain class [28, 30] (there are also more technical aspects involving the problem dimensions and a black-box assumption). In this paper we focus on the classes $\mathcal{F}_{0,L}$ and $\mathcal{F}_{\mu,L}$.

Algorithm 1: *GPBB*

input : $x^{(0)}$, K
output: $x^{(k+1)}$

$p_0 = 1$;
for $k = 0, 1, 2, \ldots$ **do**
  `// BB strategy`
  if $k > 0$ **then**
    $p_k \leftarrow \frac{\|x^{(k)} - x^{(k-1)}\|_2^2}{(x^{(k)} - x^{(k-1)})^T (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)}))}$;
  $\beta \leftarrow 0.95$;
  $\bar{x} \leftarrow P_{\mathcal{Q}}(x^{(k)} - \beta p_k \nabla f(x^{(k)}))$;
  $\hat{f} \leftarrow \max\{f(x^{(k)}), f(x^{(k-1)}), \ldots, f(x^{(k-K)})\}$;
  while $f(\bar{x}) \geq \hat{f} - \sigma \nabla f(x^{(k)})^T (x^{(k)} - \bar{x})$ **do**
    $\beta \leftarrow \beta^2$;
    $\bar{x} \leftarrow P_{\mathcal{Q}}(x^{(k)} - \beta p_k \nabla f(x^{(k)}))$;
  $x^{(k+1)} \leftarrow \bar{x}$;

Recently there has been a great deal of interest in optimal first-order methods for convex optimization problems with $f \in \mathcal{F}_{0,L}$ [3, 38]. For this class it is possible to reach an ϵ-suboptimal solution within $\mathcal{O}(\sqrt{L/\epsilon})$ iterations. Nesterov's methods can be used as stand-alone optimization algorithm, or in a composite objective setup [4, 32, 38], in which case they are called accelerated methods (because the designer violates the black-box assumption). Another option is to apply optimal first-order methods to a smooth approximation of a non-smooth function leading to an algorithm with $\mathcal{O}(1/\epsilon)$ complexity [31]; for practical considerations, see [5, 16].

Optimal methods specific for the function class $\mathcal{F}_{\mu,L}$ with $\mu > 0$ are also known [29, 30]; see also [32] for the composite objective version. However, these methods have gained little practical consideration; for example in [32] all the simulations are conducted with $\mu = 0$. Optimal methods require $\mathcal{O}(\sqrt{Q}\log\epsilon^{-1})$ iterations while the classic gradient method requires $\mathcal{O}(Q\log\epsilon^{-1})$ iterations [28, 30]. For quadratic problems, the conjugate gradient method achieves the same iteration complexity as the optimal first-order method [28].

In Algorithm 2 we state the basic optimal method *Nesterov* [30] with known μ and L; it requires an initial $\theta_0 \geq \sqrt{\mu/L}$. Note that it uses two sequences of vectors, $x^{(k)}$ and $y^{(k)}$. The convergence rate is provided by the following theorem.

Theorem 4.2 *If $f \in \mathcal{F}_{\mu,L}$, $1 > \theta_0 \geq \sqrt{\mu/L}$, and $\gamma_0 = \frac{\theta_0(\theta_0 L - \mu)}{1-\theta_0}$, then for algorithm Nesterov we have*

$$f\big(x^{(k)}\big) - f^\star \leq \frac{4L}{(2\sqrt{L} + k\sqrt{\gamma_0})^2}\left(f\big(x^{(0)}\big) - f^\star + \frac{\gamma_0}{2}\|x^{(0)} - x^\star\|_2^2\right). \tag{4.4}$$

Moreover, if $\mu \neq 0$ then

$$f\big(x^{(k)}\big) - f^\star \leq \left(1 - \sqrt{\frac{\mu}{L}}\right)^k \left(f\big(x^{(0)}\big) - f^\star + \frac{\gamma_0}{2}\|x^{(0)} - x^\star\|_2^2\right). \tag{4.5}$$

Proof See [30, (2.2.19), Theorem 2.2.3] and Appendix A for an alternative proof. □

Except for different constants Theorem 4.2 mimics the result in Theorem 4.1, with the crucial differences that the denominator in (4.4) is squared and μ/L in (4.5) has

Algorithm 2: *Nesterov*

input : $x^{(0)}$, μ, L, θ_0
output: $x^{(k+1)}$

$y^{(0)} \leftarrow x^{(0)}$;
for $k = 0, 1, 2, \ldots$ **do**
  $x^{(k+1)} \leftarrow P_{\mathcal{Q}}(y^{(k)} - L^{-1}\nabla f(y^{(k)}))$;
  $\theta_{k+1} \leftarrow$ positive root of $\theta^2 = (1-\theta)\theta_k^2 + \frac{\mu}{L}\theta$;
  $\beta_k \leftarrow \theta_k(1-\theta_k)/(\theta_k^2 + \theta_{k+1})$;
  $y^{(k+1)} \leftarrow x^{(k+1)} + \beta_k(x^{(k+1)} - x^{(k)})$;

a square root. Comparing the convergence rates in Theorems 4.1 and 4.2, we see that the rates are linear but differ in the linear rate, Q^{-1} and $\sqrt{Q^{-1}}$, respectively. For ill-conditioned problems, it is important whether the complexity is a function of Q or $\sqrt{Q}$, see, e.g., [28, §7.2.8], [7]. This motivates the interest in specialized optimal first-order methods for solving ill-conditioned problems.

5 First-order inequalities for the gradient map

For unconstrained convex problems the (norm of the) gradient is a measure of how close we are to the minimum, through the first-order optimality condition, cf. [9]. For constrained convex problems $\min_{x\in\mathcal{Q}} f(x)$ there is a similar quantity, namely, the *gradient map* defined by

$$G_\nu(x) = \nu(x - P_{\mathcal{Q}}(x - \nu^{-1}\nabla f(x))). \tag{5.1}$$

Here $\nu > 0$ is a parameter and ν^{-1} can be interpreted as the step size of a gradient step. The gradient map is a generalization of the gradient to constrained problems in the sense that if $\mathcal{Q} = \mathbb{R}^N$ then $G_\nu(x) = \nabla f(x)$, and the equality $G_\nu(x^\star) = 0$ is a necessary and sufficient optimality condition [39]. In what follows we review and derive some important first-order inequalities which will be used to analyze the proposed algorithm. We start with a rather technical result.

Lemma 5.1 *Let $f \in \mathcal{F}_{\mu,L}$, fix $x \in \mathcal{Q}$, $y \in \mathbb{R}^N$, and set $x^+ = P_{\mathcal{Q}}(y - \bar{L}^{-1}\nabla f(y))$, where $\bar{\mu}$ and $\bar{L}$ are related to x, y and x^+ by the inequalities*

$$f(x) \geq f(y) + \nabla f(y)^T(x - y) + \frac{1}{2}\bar{\mu}\|x - y\|_2^2, \tag{5.2}$$

$$f(x^+) \leq f(y) + \nabla f(y)^T(x^+ - y) + \frac{1}{2}\bar{L}\|x^+ - y\|_2^2. \tag{5.3}$$

Then

$$f\left(x^+\right) \leq f(x) + G_{\bar{L}}(y)^T(y - x) - \frac{1}{2}\bar{L}^{-1}\|G_{\bar{L}}(y)\|_2^2 - \frac{1}{2}\bar{\mu}\|y - x\|_2^2. \tag{5.4}$$

Proof Follows directly from [30, Theorem 2.2.7]. □

Note that if $f \in \mathcal{F}_{\mu,L}$, then in Lemma 5.1 we can always select $\bar{\mu} = \mu$ and $\bar{L} = L$ to ensure that the inequalities (5.2) and (5.3) are satisfied. However, for specific x, y and x^+, there can exist $\bar{\mu} \geq \mu$ and $\bar{L} \leq L$ such that (5.2) and (5.3) hold. We will use these results to design an algorithm for unknown parameters μ and L.

The lemma can be used to obtain the following lemma. The derivation of the bounds is inspired by similar results for composite objective functions in [32], and the second result is similar to [30, Corollary 2.2.1].

Lemma 5.2 *Let $f \in \mathcal{F}_{\mu,L}$, fix $y \in \mathbb{R}^N$, and set $x^+ = P_{\mathcal{Q}}(y - \bar{L}^{-1}\nabla f(y))$. Let $\bar{\mu}$ and $\bar{L}$ be selected in accordance with (5.2) and (5.3) respectively. Then*

$$\frac{1}{2}\bar{\mu}\|y - x^\star\|_2 \leq \|G_{\bar{L}}(y)\|_2. \tag{5.5}$$

If $y \in \mathcal{Q}$ then

$$\frac{1}{2}\bar{L}^{-1}\|G_{\bar{L}}(y)\|_2^2 \leq f(y) - f\left(x^+\right) \leq f(y) - f^\star. \tag{5.6}$$

Proof From Lemma 5.1 with $x = x^\star$ we use $f(x^+) \geq f^\star$ and obtain

$$\frac{1}{2}\bar{\mu}\|y - x^\star\|_2^2 \leq G_{\bar{L}}(y)^T\left(y - x^\star\right) - \frac{1}{2}\bar{L}^{-1}\|G_{\bar{L}}(y)\|_2^2 \leq \|G_{\bar{L}}(y)\|_2\|y - x^\star\|_2,$$

and (5.5) follows; (5.6) follows from Lemma 5.1 using $y = x$ and $f^\star \leq f(x^+)$. □

As mentioned in the beginning of the section, the results of the corollary say that we can relate the norm of the gradient map at y to the error $\|y - x^*\|_2$ as well as to $f(y) - f^*$. This motivates the use of the gradient map in a stopping criterion:

$$\|G_{\bar{L}}(y)\|_2 \leq \bar{\epsilon}, \tag{5.7}$$

where y is the current iterate, and $\bar{L}$ is linked to this iterate using (5.3). The parameter $\bar{\epsilon}$ is a user-specified tolerance based on the requested accuracy. Lemma 5.2 is also used in the following section to develop a restart criterion to ensure convergence.

6 Nesterov's method with parameter estimation

The parameters μ and L are explicitly needed in *Nesterov*. In case of an unregularized least-squares problem we can in principle compute μ and L as the smallest and largest squared singular value of A, though it might be computational expensive. When a regularization term is present it is unclear whether the tight μ and L can be computed at all. Bounds can be obtained using the result in Theorem 3.1.

A practical approach is to estimate μ and L during the iterations. To this end, we introduce the estimates μ_k and L_k of μ and L in each iteration k. We discuss first how to choose L_k, then μ_k, and finally we state the complete algorithm *UPN* and its convergence properties.

To ensure convergence, the main inequalities (A.6) and (A.7) must be satisfied. Hence, according to Lemma 5.1 we need to choose L_k such that

$$f(x^{(k+1)}) \leq f(y^{(k)}) + \nabla f(y^{(k)})^T(x^{(k+1)} - y^{(k)}) + \frac{1}{2}L_k\|x^{(k+1)} - y^{(k)}\|_2^2. \tag{6.1}$$

This is easily accomplished using *backtracking* on L_k [4]. The scheme, *BT*, takes the form given in Algorithm 3, where $\rho_L > 1$ is an adjustment parameter. If the loop

Algorithm 3: *BT*

input : $y, \bar{L}$
output: $x, \tilde{L}$
$\tilde{L} \leftarrow \bar{L}$;
$x \leftarrow P_{\mathcal{Q}}(y - \tilde{L}^{-1}\nabla f(y))$;
while $f(x) > f(y) + \nabla f(y)^T(x-y) + \frac{1}{2}\tilde{L}\|x-y\|_2^2$ **do**
  $\tilde{L} \leftarrow \rho_L \tilde{L}$;
  $x \leftarrow P_{\mathcal{Q}}(y - \tilde{L}^{-1}\nabla f(y))$;

is executed n_{BT} times, the dominant computational cost of *BT* is $n_{BT} + 2$ function evaluations and 1 gradient evaluation.

For choosing the estimate μ_k we introduce the auxiliary variable $\mu_k^\star$ as the value that causes Definition 3.1 (of strong convexity) for $x^\star$ and $y^{(k)}$ to hold with equality

$$f(x^\star) = f(y^{(k)}) + \nabla f(y^{(k)})^T(x^\star - y^{(k)}) + \frac{1}{2}\mu_k^\star \|x^\star - y^{(k)}\|_2^2. \tag{6.2}$$

From (A.7) with Lemma 5.1 and (A.8) we find that we must choose $\mu_k \leq \mu_k^\star$ to obtain a convergent algorithm. However, as $x^\star$ is, of course, unknown, this task is not straightforward, if at all possible. Instead, we propose a *heuristic* where we select μ_k such that

$$f(x^{(k)}) \geq f(y^{(k)}) + \nabla f(y^{(k)})^T(x^{(k)} - y^{(k)}) + \frac{1}{2}\mu_k \|x^{(k)} - y^{(k)}\|_2^2. \tag{6.3}$$

This is indeed possible since $x^{(k)}$ and $y^{(k)}$ are known iterates. Furthermore, we want the estimate μ_k to be decreasing in order to approach a better estimate of μ. This can be achieved by the choice

$$\mu_k = \min\{\mu_{k-1}, M(x^{(k)}, y^{(k)})\}, \tag{6.4}$$

where we have defined the function

$$M(x,y) = \begin{cases} \frac{f(x) - f(y) - \nabla f(y)^T(x-y)}{\frac{1}{2}\|x-y\|_2^2}, & \text{if } x \neq y, \\ \infty, & \text{else.} \end{cases} \tag{6.5}$$

In words, the heuristic chooses the largest μ_k that satisfies (3.2) for $x^{(k)}$ and $y^{(k)}$, as long as μ_k is not larger than μ_{k-1}. The heuristic is simple and computationally inexpensive and we have found that it is effective for determining a useful estimate. Unfortunately, convergence of *Nesterov* equipped with this heuristic is not guaranteed, since the estimate can be too large. To ensure convergence we include a restart procedure *RUPN* that detects if μ_k is too large, inspired by the approach in [32, §5.3] for composite objectives. *RUPN* is given in Algorithm 4.

Algorithm 4: *RUPN*

$\gamma_1 = \theta_1(\theta_1 L_1 - \mu_1)/(1-\theta_1)$;
if $\mu_k \neq 0$ **and** inequality (6.9) not satisfied **then**
  abort execution of *UPN*;
  restart *UPN* with input $(x^{(k+1)},\ \rho_\mu \mu_k,\ L_k,\ \bar{\epsilon})$;

To analyze the restart strategy, assume that μ_i for all $i = 1, \ldots, k$ are *small enough*, i.e., they satisfy $\mu_i \leq \mu_i^\star$ for $i = 1, \ldots, k$, and μ_k satisfies

$$f(x^\star) \geq f(x^{(0)}) + \nabla f(x^{(0)})^T (x^\star - x^{(0)}) + \frac{1}{2}\mu_k \|x^\star - x^{(0)}\|_2^2. \tag{6.6}$$

When this holds we have the convergence result (using (A.9))

$$f(x^{(k+1)}) - f^\star \leq \prod_{i=1}^{k} (1 - \sqrt{\mu_i / L_i}) \left(f(x^{(1)}) - f^\star + \frac{1}{2}\gamma_1 \|x^{(1)} - x^\star\|_2^2 \right). \tag{6.7}$$

We start from iteration $k = 1$ for reasons which will presented shortly (see Appendix A for details and definitions). If the algorithm uses a projected gradient step from the initial $x^{(0)}$ to obtain $x^{(1)}$, the rightmost factor of (6.7) can be bounded as

$$\begin{aligned}
&f\left(x^{(1)}\right) - f^\star + \frac{1}{2}\gamma_1 \|x^{(1)} - x^\star\|_2^2 \\
&\quad \leq G_{L_0}\left(x^{(0)}\right)^T \left(x^{(0)} - x^\star\right) - \frac{1}{2} L_0^{-1} \left\| G_{L_0}\left(x^{(0)}\right) \right\|_2^2 + \frac{1}{2}\gamma_1 \|x^{(1)} - x^\star\|_2^2 \\
&\quad \leq \left\| G_{L_0}\left(x^{(0)}\right) \right\|_2 \|x^{(0)} - x^\star\|_2 - \frac{1}{2} L_0^{-1} \left\| G_{L_0}\left(x^{(0)}\right) \right\|_2^2 + \frac{1}{2}\gamma_1 \|x^{(0)} - x^\star\|_2^2 \\
&\quad \leq \left(\frac{2}{\mu_k} - \frac{1}{2L_0} + \frac{2\gamma_1}{\mu_k^2} \right) \left\| G_{L_0}\left(x^{(0)}\right) \right\|_2^2.
\end{aligned} \tag{6.8}$$

Here we used Lemma 5.1, and the fact that a projected gradient step reduces the Euclidean distance to the solution [30, Theorem 2.2.8]. Using Lemma 5.2 we arrive at the bound, where $\tilde{L}_{k+1}$ is defined in Algorithm *UPN*:

$$\frac{1}{2}\tilde{L}_{k+1}^{-1} \left\| G_{\tilde{L}_{k+1}}\left(x^{(k+1)}\right) \right\|_2^2 \leq \prod_{i=1}^{k} \left(1 - \sqrt{\frac{\mu_i}{L_i}} \right) \left(\frac{2}{\mu_k} - \frac{1}{2L_0} + \frac{2\gamma_1}{\mu_k^2} \right) \left\| G_{L_0}\left(x^{(0)}\right) \right\|_2^2. \tag{6.9}$$

If the algorithm detects that (6.9) is not satisfied, it can only be because there was at least one μ_i for $i = 1, \ldots, k$ which was *not small enough*. If this is the case, we restart the algorithm with a new $\bar{\mu} \leftarrow \rho_\mu \mu_k$, where $0 < \rho_\mu < 1$ is a parameter, using the current iterate $x^{(k+1)}$ as initial vector.

The complete algorithm *UPN* (Unknown-Parameter Nesterov) is given in Algorithm 5. *UPN* is based on Nesterov's optimal method where we have included backtracking on L_k and the heuristic (6.4). An initial vector $x^{(0)}$ and initial parameters

Algorithm 5: *UPN*

input : $x^{(0)}, \bar{\mu}, \bar{L}, \bar{\epsilon}$
output: $x^{(k+1)}$ or $\tilde{x}^{(k+1)}$
1 $[x^{(1)}, L_0] \leftarrow BT(x^{(0)}, \bar{L})$;
2 $\mu_0 = \bar{\mu},\ y^{(1)} \leftarrow x^{(1)},\ \theta_1 \leftarrow \sqrt{\mu_0/L_0}$;
3 **for** $k = 1, 2, \ldots$ **do**
4 $\quad [x^{(k+1)}, L_k] \leftarrow BT(y^{(k)}, L_{k-1})$;
5 $\quad [\tilde{x}^{(k+1)}, \tilde{L}_{k+1}] \leftarrow BT(x^{(k+1)}, L_k)$;
6 $\quad$ **if** $\|G_{\tilde{L}_{k+1}}(x^{(k+1)})\|_2 \le \bar{\epsilon}$ **then abort**, **return** $\tilde{x}^{(k+1)}$;
7 $\quad$ **if** $\|G_{L_k}(y^{(k)})\|_2 \le \bar{\epsilon}$ **then abort**, **return** $x^{(k+1)}$;
8 $\quad \mu_k \leftarrow \min\{\mu_{k-1}, M(x^{(k)}, y^{(k)})\}$;
9 $\quad$ *RUPN*;
10 $\quad \theta_{k+1} \leftarrow$ positive root of $\theta^2 = (1-\theta)\theta_k^2 + (\mu_k/L_k)\,\theta$;
11 $\quad \beta_k \leftarrow \theta_k(1-\theta_k)/(\theta_k^2 + \theta_{k+1})$;
12 $\quad y^{(k+1)} \leftarrow x^{(k+1)} + \beta_k(x^{(k+1)} - x^{(k)})$;

$\bar{\mu} \ge \mu$ and $\bar{L} \le L$ must be specified along with the requested accuracy $\bar{\epsilon}$. The changes from *Nesterov* to *UPN* are at the following lines:

1: Initial projected gradient step to obtain the bound (6.8) and thereby the bound (6.9) used for the restart criterion.
5: Extra projected gradient step explicitly applied to obtain the stopping criterion $\|G_{\tilde{L}_{k+1}}(x^{(k+1)})\|_2 \le \bar{\epsilon}$.
6,7: Used to relate the stopping criterion in terms of $\bar{\epsilon}$ to ϵ, see Appendix B.3.
8: The heuristic choice of μ_k in (6.4).
9: The restart procedure for inadequate estimates of μ.

We note that in a practical implementation, the computational work involved in one iteration step of *UPN* may—in the worst case situation—be twice that of one iteration of *GPBB*, due to the two calls to *BT*. However, it may be possible to implement these two calls more efficiently than naively calling *BT* twice. We will instead focus on the iteration complexity of *UPN* given in the following theorem.

Theorem 6.1 *Algorithm UPN, applied to $f \in \mathcal{F}_{\mu,L}$ under conditions $\bar{\mu} \ge \mu$, $\bar{L} \le L$, $\bar{\epsilon} = \sqrt{(\mu/2)\,\epsilon}$, stops using the gradient map magnitude measure and returns an ϵ-suboptimal solution with iteration complexity*

$$\mathcal{O}\big(\sqrt{Q}\log Q\big) + \mathcal{O}\big(\sqrt{Q}\log \epsilon^{-1}\big). \tag{6.10}$$

Proof See Appendix B. □

The term $\mathcal{O}(\sqrt{Q}\log Q)$ in (6.10) follows from application of several inequalities involving the problem dependent parameters μ and L to obtain the overall bound

(6.9). Algorithm *UPN* is suboptimal since the optimal complexity is $\mathcal{O}(\sqrt{Q}\log\epsilon^{-1})$ but it has the advantage that it can be applied to problems with unknown μ and L.

7 Numerical experiments

7.1 An image deblurring example

We exemplify the use of the algorithm *UPN* to solve a total variation regularized image deblurring problem, where the goal is to determine a sharp image x from a blurred and noisy one $b = Ax + e$. The matrix A models linear motion blur, which renders A sparse, and we use reflexive boundary conditions. For this type of blur no fast transform can be exploited. We add Gaussian noise e with relative noise level $\|e\|_2/\|b\|_2 = 0.01$ and reconstruct using $\alpha = 5.0$ and the default setting of $\tau = 10^{-4} \cdot 255$, where $[0, 255]$ is the dynamic pixel intensity range. The result is shown in Fig. 1. We recognize well-known features of TV-regularized reconstructions: Sharp edges are well-preserved, while fine texture has been over-regularized and has a "patchy" appearance.

To investigate the convergence of the methods, we need the true minimizer $x^\star$ with $\phi(x^\star) = \phi^\star$, which is unknown for the test problem. However, for comparison it is enough to use a reference solution much closer to the true minimizer than the iterates. Thus, to compare the accuracy of the solutions obtained with the accuracy parameter $\bar{\epsilon}$, we use a reference solution computed with accuracy $(\bar{\epsilon} \cdot 10^{-2})$, and with abuse of notation we use $x^\star$ to denote this reference solution.

In Fig. 1 both *UPN* and UPN_0 are seen to be faster than *GP* and *GPBB*, and for a high-accuracy solution *UPN* also outperforms UPN_0. For *UPN*, *GP* and *GPBB* we observe linear rates of convergence, but *UPN* converges much faster. UPN_0 shows a sublinear convergence rate, however the initial phase is steep enough that it takes *UPN* almost 1000 iterations to catch up. We note that the potential of *UPN* seems to be in the case where a high-accuracy solution is needed.

Having demonstrated the performance of the proposed algorithm in an image deblurring problem, we focus in the remainder on a 3D tomography test problem, for which we further study the convergence behavior including the influence of the regularization and smoothing parameters.

7.2 Experiments with 3D tomographic reconstruction

Tomography problems arise in numerous areas, such as medical imaging, nondestructive testing, materials science, and geophysics [23, 26, 33]. These problems amount to reconstructing an object from its projections along a number of specified directions, and these projections are produced by X-rays, seismic waves, or other "rays" penetrating the object in such a way that their intensity is partially absorbed by the object. The absorption thus gives information about the object.

The following generic model accounts for several applications of tomography. We consider an object in 3D with linear attenuation coefficient $\mathcal{X}(t)$, with $t \in \Omega \subset \mathbb{R}^3$.

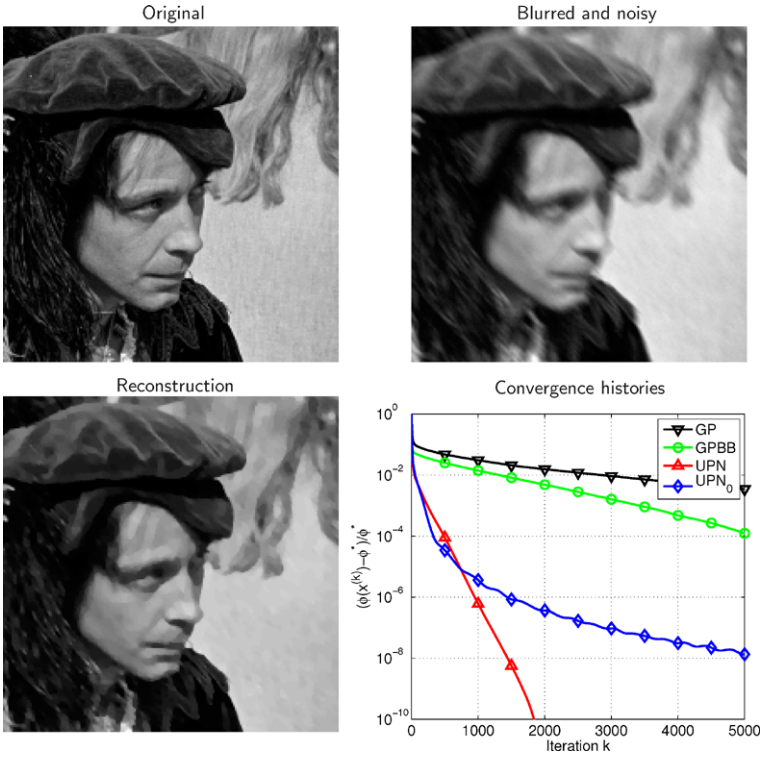

Fig. 1 Example of total variation deblurring for motion blur with reflexive boundary conditions. Methods are Gradient Projection (*GP*), Gradient Projection Barzilai-Borwein (*GPBB*), Unknown Parameter Nesterov (*UPN*), and *UPN* with $\mu_k = 0$ (UPN_0). Both *UPN* and UPN_0 are much faster than *GP* and *GPBB*, and for a high-accuracy solution *UPN* also outperforms UPN_0

The intensity decay b_i of a ray along the line ℓ_i through Ω is governed by a line integral,

$$b_i = \log(I_0/I_i) = \int_{\ell_i} \mathcal{X}(t)\, d\ell = b_i, \tag{7.1}$$

where I_0 and I_i are the intensities of the ray before and after passing through the object. When a large number of these line integrals are recorded, then we are able to reconstruct an approximation of the function $\mathcal{X}(t)$.

We discretize the problem as described in Sect. 2, such that $\mathcal{X}$ is approximated by a piecewise constant function in each voxel in the domain $\Omega = [0, 1] \times [0, 1] \times [0, 1]$. Then the line integral along ℓ_i is computed by summing the contributions from all the voxels penetrated by ℓ_i. If the path length of the ith ray through the jth voxel is denoted by a_{ij}, then we obtain the linear equations

$$\sum_{j=1}^{N} a_{ij} x_j = b_i, \quad i = 1, \ldots, M, \tag{7.2}$$

Fig. 2 *Left*: Two orthogonal slices through the 3D Shepp-Logan phantom discretized on a 43^3 grid used in our test problems. *Middle*: Central horizontal slice. *Right*: Example of solution for $\alpha = 1$ and $\tau = 10^{-4}$. A less smooth solution can be obtained using a smaller α. Original voxel/pixel values are 0.0, 0.2, 0.3 and 1.0. Color range in display is set to [0.1, 0.4] for better contrast

Table 1 Specifications of the two test problems; the object domain consists of $m \times n \times l$ voxels and each projection is a $p \times p$ image. Any zero rows have been purged from A

Problem	$m = n = l$	p	Projections	Dimensions of A	Rank
T1	43	63	37	99361×79507	=79507
T2	43	63	13	33937×79507	<79507

where M is the number of rays or measurements and N is the number of voxels. This is a linear system of equations $A\,x = b$ with a sparse coefficient matrix $A \in \mathbb{R}^{M \times N}$.

A widely used test image in medical tomography is the "Shepp-Logan phantom," which consists of a number superimposed ellipses. In the MATLAB function `shepplogan3d` [37] this 2D image is generalized to 3D by superimposing ellipsoids instead. The voxels are in the range [0, 1], and Fig. 2 shows an example with $43 \times 43 \times 43$ voxels.

We construct the matrix A for a parallel-beam geometry with orthogonal projections of the object along directions well distributed over the unit sphere. The projection directions are the direction vectors of so-called *Lebedev quadrature* points on the unit sphere, and the directions are evenly distributed over the sphere; we use the MATLAB implementation `getLebedevSphere` [34]. For setting up the tomography system matrix for a parallel beam geometry, we use the Matlab implementation `tomobox` [25].

This section describes our numerical experiments with the four methods *UPN*, UPN_0, *GP* and *GPBB* applied to the TV regularization problem (2.7). We use the two test problems listed in Table 1, which are representative across a larger class of problems (other directions, number of projections, noise levels, etc.) that we have run simulations with. The smallest eigenvalue of $A^T A$ for *T1* is $2.19 \cdot 10^{-5}$ (as computed by MATLAB's `eigs`), confirming that $\text{rank}(A) = N$ for *T1*. We emphasize that this computation is only conducted to support the analysis of the considered problems since—as we have argued in the introduction—it carries a considerable computational burden to compute. In all simulations we create noisy data from an exact object x_{exact} through the forward mapping $b = Ax_{\text{exact}} + e$, subject to additive Gaussian

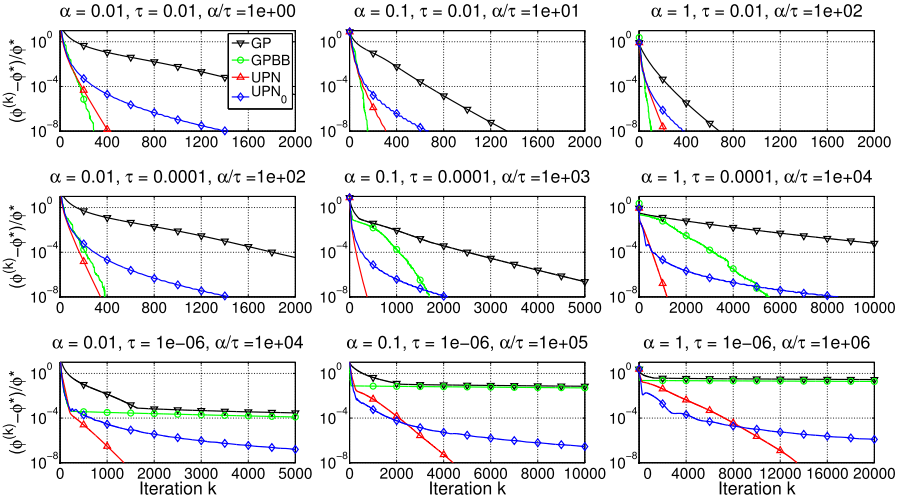

Fig. 3 Convergence histories $(\phi(x^{(k)}) - \phi^\star)/\phi^\star$ vs. k for *T1* with $\alpha = 0.01$, 0.1 and 1 and $\tau = 10^{-2}$, 10^{-4} and 10^{-6}. Methods are Gradient Projection (*GP*), Gradient Projection Barzilai-Borwein (*GPBB*), Unknown Parameter Nesterov (*UPN*), and *UPN* with $\mu_k = 0$ (UPN_0). As the ratio α/τ increases, which implies an increased Q and a computationally more difficult problem, *UPN* and UPN_0 scale significantly better. For high accuracy solutions *UPN* is always competitive

white noise of relative noise level $\|e\|_2/\|b\|_2 = 0.01$. As initial point of the optimization algorithm we use the fifth iteration of the conjugate gradient method applied to the least squares problem.

We compare the algorithm *UPN* with *GP* (the gradient projection method (4.1) with backtracking line search on the step size), *GPBB* and UPN_0. The latter is *UPN* with $\mu_i = 0$ for all $i = 0, \ldots, k$ and $\theta_1 = 1$ and is optimal for the class $\mathcal{F}_{0,L}$.

7.3 Influence of α and τ on the convergence

For a given A the theoretical modulus of strong convexity given in (3.6) varies only with α and τ. We therefore expect better convergence rates (4.3) and (4.5) for smaller α and larger τ. In Fig. 3 we show the convergence histories for *T1* with all combinations of $\alpha = 0.01$, 0.1, 1 and $\tau = 10^{-2}$, 10^{-4}, 10^{-6}.

For low α/τ ratios, i.e., small condition number of the Hessian, *GPBB* and *GP* requires a comparable or smaller number of iterations than *UPN* and UPN_0. As α/τ increases, both *GPBB* and *GP* exhibit slower convergence, while *UPN* is less affected. In all cases *UPN* shows linear convergence, at least in the final stage, while UPN_0 shows sublinear convergence. Due to these observations, we consistently observe that for sufficiently high accuracy, *UPN* requires the lowest number of iterations. This also follows from the theory since *UPN* scales as $\mathcal{O}(\log \epsilon^{-1})$, whereas UPN_0 scales at a higher complexity of $\mathcal{O}(\sqrt{\epsilon^{-1}})$.

We conclude that for small condition numbers there is no gain in using *UPN* compared to *GPBB*. For larger condition numbers, and in particular if a high-accuracy solution is required, *UPN* converges significantly faster. Assume that we were to choose only one of the four algorithms to use for reconstruction across the condition

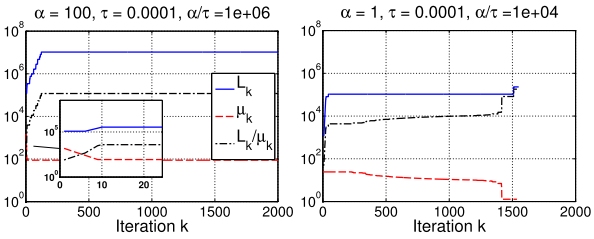

Fig. 4 The μ_k, L_k histories for *T1*. *Left*: $\alpha = 100$ and $\tau = 10^{-4}$. *Right*: $\alpha = 1$ and $\tau = 10^{-4}$

number range. When *UPN* requires the lowest number of iterations, it requires *significantly* fewer, and when not, *UPN* only requires slightly more iterations than the best of the other algorithms. Therefore, *UPN* appears to be the best choice. Obviously, the choice of algorithm also depends on the demanded accuracy of the solution. If only a low accuracy, say $(\phi^{(k)} - \phi^\star)/\phi^\star = 10^{-2}$ is sufficient, all four methods perform more or less equally well.

7.4 Restarts and μ_k and L_k histories

To ensure convergence of *UPN* we introduced the restart functionality *RUPN*. In practice, we almost never observe a restart, e.g., in none of the experiments reported so far a restart occurred. An example where restarts do occur is obtained if we increase α to 100 for *T1* (still $\tau = 10^{-4}$). Restarts occur in the first 8 iterations, and each time μ_k is reduced by a constant factor of $\rho_\mu = 0.7$. In Fig. 4, left, the μ_k and L_k histories are plotted vs. k and the restarts are seen in the zoomed inset as the rapid, constant decrease in μ_k. From the plot we also note that after the decrease in μ_k and an initial increase in L_k, both estimates are constant for the remaining iterations, indicating that the heuristics determine sufficient values.

For comparison the μ_k and L_k histories for *T1* with $\alpha = 1$ and $\tau = 10^{-4}$ are seen in Fig. 4, right. No restarts occurred here, and μ_k decays gradually, except for one final jump, while L_k remains almost constant.

7.5 A non-strongly convex example

Test problem *T2* corresponds to only 13 projections, which causes A to not have full column rank. This leads to $\lambda_{\min}(A^T A) = 0$, and hence $\phi(x)$ is not strongly convex. The optimal convergence rate is therefore given by (4.4); but how does the lack of strong convexity affect *UPN*, which was specifically constructed for strongly convex problems? *UPN* does not recognize that the problem is not strongly convex but simply relies on the heuristic (6.4) at the kth iteration. We investigate the convergence by solving *T2* with $\alpha = 1$ and $\tau = 10^{-4}$. Convergence histories are given in Fig. 5, left. The algorithm *UPN* still converges linearly, although slightly slower than in the *T1* experiment ($\alpha = 1, \tau = 10^{-4}$) in Fig. 3. The algorithms *GP* and *GPBB* converge much more slowly, while at low accuracies UPN_0 is comparable to *UPN*. But the linear convergence makes *UPN* converge faster for high accuracy solutions.

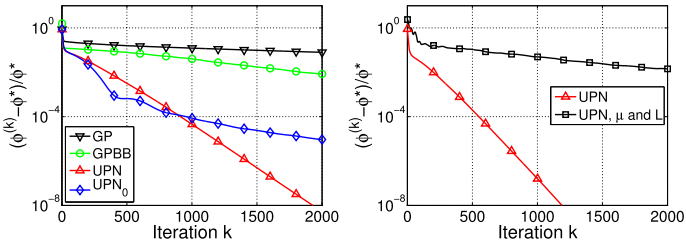

Fig. 5 *Left*: Convergence histories of *GP*, *GPBB*, *UPN* and UPN_0 on *T2* with $\alpha = 1$ and $\tau = 10^{-4}$. *Right*: Convergence histories of *UPN* and *UPN* using true μ and L on *T1* with $\alpha = 1$ and $\tau = 10^{-4}$

7.6 Influence of the heuristic

An obvious question is how the use of the heuristic for estimating μ affects *UPN* compared to *Nesterov*, where μ (and L) are assumed known. From Theorem 3.1 we can compute a strong convexity parameter and a Lipschitz parameter for $\phi(x)$ assuming we know the largest and smallest magnitude eigenvalues of $A^T A$. Recall that these μ and L are not necessarily the tightest possible, according to Remark 3.3. For *T1* we have computed $\lambda_{\max}(A^T A) = 1.52 \cdot 10^3$ and $\lambda_{\min}(A^T A) = 2.19 \cdot 10^{-5}$ (by means of `eigs` in MATLAB). Using $\alpha = 1$, $\tau = 10^{-4}$ and $\|D\|_2^2 \leq 12$ from Lemma 3.2 we fix

$$\mu_k = \lambda_{\min}(A^T A) = 2.19 \cdot 10^{-5}, \qquad L_k = \lambda_{\max}(A^T A) + 12\frac{\alpha}{\tau} = 1.22 \cdot 10^5,$$

for all k, and solve test problem *T1* using *UPN* with the heuristics switched off in favor of these *true* strong convexity and Lipschitz parameters. Convergence histories are plotted in Fig. 5, right.

The convergence is much slower than using *UPN* with the heuristics switched on. We ascribe this behavior to the very large modulus of strong convexity that arise from the true μ and L. It appears that *UPN* works better than the actual degree of strong convexity as measured by μ, by heuristically choosing in each step a μ_k that is sufficient *locally* instead of being restricted to using a *globally* valid μ.

Another question is how much is actually gained in using the heuristic for μ in *UPN* compared to simply using a fixed "guess" throughout the iterations. To answer that question we investigate the number iterations required to obtain $\bar{\epsilon} = 10^{-4}$, 10^{-6} and 10^{-8} solutions for *T1* and *T2* using only the backtracking procedure on L and simply a fixed value $\mu_k \in [10^{-4}, 10^4]$ for all iterations k, see Fig. 6.

The choice of fixed μ_k has a large impact on the required number of iterations, and there is a distinct optimal choice between 1 and 10. Choosing a fixed μ_k away from the optimal one leads to more iterations and the number of additional iterations grows faster for more accurate solutions. For comparison the figure also shows the corresponding number of iterations required by *UPN* plotted as function of the final *UPN*-estimate for μ. For all three *T1* cases *UPN* comes very close to the optimal number of iterations, without demanding an accurate guess of μ by the user. For *T2* we observe similar trends, although *UPN* requires slightly more iterations than with the optimal choice of fixed μ_k.

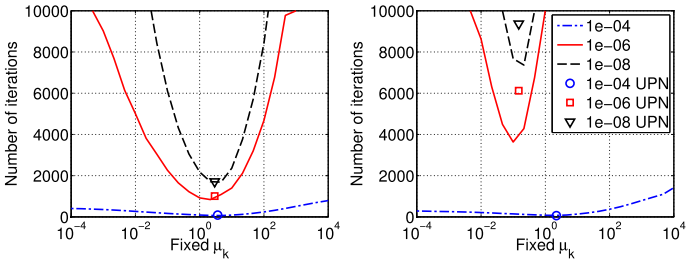

Fig. 6 Number of iterations needed to obtain TV-solutions ($\alpha = 0.01$) to tolerances $\bar{\epsilon} = 10^{-4}$, 10^{-6} and 10^{-8} using fixed μ_k, *left T1*, *right T2*. Also shown are the number iterations needed by *UPN* as function of the final estimate of μ. Choices of μ_k not equal to the unknown optimal value lead to many more iterations. *UPN* needs a near-optimal number of iterations without requiring the user to choose a value for μ

We conclude that there exists a choice of fixed μ_k that gives good performance; however, for an inaccurate guess of this value, the number of iterations will be much higher, in particular if an accurate solution is required. *UPN* avoids the need for such a guess and provides the solution using a near-optimal number of iterations. We emphasize that obtaining a *true* strong convexity parameter μ is not of particular interest here, nor is the final *UPN*-estimate for μ, as the goal is simply to obtain fast convergence.

8 Conclusion

We presented an implementation of an optimal first-order optimization algorithm for large-scale problems, suited for functions that are smooth and strongly convex. While the underlying algorithm by Nesterov depends on knowledge of two parameters that characterize the smoothness and strong convexity, we have implemented methods that estimate these parameters during the iterations, thus making the algorithm of practical use.

We tested the performance of the algorithm and compared it with two variants of the gradient projection algorithm and a variant of the FISTA algorithm. We applied the algorithms to total variation regularized tomographic reconstruction of a generic threedimensional test problem. The tests show that, with regards to the number of iterations, the proposed algorithm is competitive with other first-order algorithms, and superior for difficult problems, i.e., ill-conditioned problems solved to high accuracy. Simulations also show that even for problems that are not strongly convex, in practice we achieve the favorable convergence rate associated with strong convexity. The software is available as a C-implementation with an interface to MATLAB from www.imm.dtu.dk/~pch/TVReg/.

Acknowledgements We wish to thank both referees for their constructive comments which helped improve the presentation of the material.

Appendix A: The optimal convergence rate

Here we provide an analysis of an optimal method for smooth, strongly convex functions without the use of estimation functions as in [30]. This approach is similar to the analysis of optimal methods for smooth functions in [38, 39]. The motivation for the following derivations is to introduce the iteration dependent L_k and μ_k estimates of L and μ. This will support the analysis of how L_k and μ_k should be selected. We start with the following relations to the "hidden" supporting variables $z^{(k)}$ and γ_k [30, pp. 73–75, 89],

$$y^{(k)} - x^{(k)} = \frac{\theta_k \gamma_k}{\gamma_{k+1}}\big(z^{(k)} - y^{(k)}\big), \tag{A.1}$$

$$\begin{aligned} &\gamma_{k+1} = (1-\theta_k)\gamma_k + \theta_k \mu_k = \theta_k^2 L_k, \\ &\gamma_{k+1} z^{(k+1)} = (1-\theta_k)\gamma_k z^{(k)} + \theta_k \mu_k y^{(k)} - \theta_k G_{L_k}\big(y^{(k)}\big). \end{aligned} \tag{A.2}$$

In addition we will make use of the relations

$$\begin{aligned} &\frac{\gamma_{k+1}}{2}\|z^{(k+1)} - y^{(k)}\|_2^2 \\ &\quad = \frac{1}{2\gamma_{k+1}}\big((1-\theta_k)^2\gamma_k^2\|z^{(k)} - y^{(k)}\|_2^2 \\ &\qquad - 2\theta_k(1-\theta_k)\gamma_k G_{L_k}\big(y^{(k)}\big)^T\big(z^{(k)} - y^{(k)}\big) + \theta_k^2\big\|G_{L_k}\big(y^{(k)}\big)\big\|_2^2\big), \end{aligned} \tag{A.3}$$

$$(1-\theta_k)\frac{\gamma_k}{2} - \frac{1}{2\gamma_{k+1}}(1-\theta_k)^2\gamma_k^2 = \frac{(1-\theta_k)\gamma_k\theta_k\mu_k}{2\gamma_{k+1}}. \tag{A.4}$$

which originate from (A.2). We will also later need the relation

$$\begin{aligned} &(1-\theta_k)\frac{\gamma_k}{2}\|z^{(k)} - y^{(k)}\|_2^2 - \frac{\gamma_{k+1}}{2}\|z^{(k+1)} - y^{(k)}\|_2^2 + \theta_k G_{L_k}\big(y^{(k)}\big)^T\big(y^{(k)} - x^\star\big) \\ &\quad = (1-\theta_k)\frac{\gamma_k}{2}\|z^{(k)} - y^{(k)}\|_2^2 - \frac{\gamma_{k+1}}{2}\|z^{(k+1)} - y^{(k)}\|_2^2 \\ &\qquad + \big(-\gamma_{k+1}z^{(k+1)} + (1-\theta_k)\gamma_k z^{(k)} + \theta_k\mu_k y^{(k)}\big)^T\big(y^{(k)} - x^\star\big) \\ &\quad = \left((1-\theta_k)\frac{\gamma_k}{2} - \frac{\gamma_{k+1}}{2} + \theta_k\mu_k\right)\big(y^{(k)}\big)^T y^{(k)} + (1-\theta_k)\frac{\gamma_k}{2}\big(z^{(k)}\big)^T z^{(k)} \\ &\qquad - \frac{\gamma_{k+1}}{2}\big(z^{(k+1)}\big)^T z^{(k+1)} + \gamma_{k+1}\big(z^{(k+1)}\big)^T x^\star \\ &\qquad - (1-\theta_k)\gamma_k\big(z^{(k)}\big)^T x^\star - \theta_k\mu_k\big(y^{(k)}\big)^T x^\star \\ &\quad = (1-\theta_k)\frac{\gamma_k}{2}\big(\|z^{(k)} - x^\star\|_2^2 - \big(x^\star\big)^T x^\star\big) - \frac{\gamma_{k+1}}{2}\big(\|z^{(k+1)} - x^\star\|_2^2 - \big(x^\star\big)^T x^\star\big) \\ &\qquad + \frac{\theta_k\mu_k}{2}\big(\|y^{(k)} - x^\star\|_2^2 - \big(x^\star\big)^T x^\star\big) \\ &\qquad + \left((1-\theta_k)\frac{\gamma_k}{2} - \frac{\gamma_{k+1}}{2} + \frac{\theta_k\mu_k}{2}\right)\big(y^{(k)}\big)^T y^{(k)} \\ &\quad = (1+\theta_k)\frac{\gamma_k}{2}\|z^{(k)} - x^\star\|_2^2 - \frac{\gamma_{k+1}}{2}\|z^{(k)} - x^\star\|_2^2 + \theta_k\frac{\mu_k}{2}\|y^{(k)} - x^\star\|_2^2, \end{aligned} \tag{A.5}$$

where we again used (A.2). We can now start the analysis of the algorithm by considering the inequality in Lemma 5.1,

$$(1-\theta_k) f\left(x^{(k+1)}\right) \leq (1-\theta_k) f\left(x^{(k)}\right) + (1-\theta_k) G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^{(k)}\right) - (1-\theta_k)\frac{1}{2L_k}\left\| G_{L_k}\left(y^{(k)}\right)\right\|_2^2, \tag{A.6}$$

where we have omitted the strong convexity part, and the inequality

$$\theta_k f\left(x^{(k+1)}\right) \leq \theta_k f\left(x^\star\right) + \theta_k G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^\star\right) - \theta_k \frac{1}{2L_k}\left\| G_{L_k}\left(y^{(k)}\right)\right\|_2^2 - \theta_k \frac{\mu_k^\star}{2}\|y^{(k)} - x^\star\|_2^2. \tag{A.7}$$

Adding these bounds and continuing, we obtain

$$\begin{aligned}
& f\left(x^{(k+1)}\right) \\
& \leq (1-\theta_k) f\left(x^{(k)}\right) + (1-\theta_k) G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^{(k)}\right) \\
& \quad + \theta_k f^\star + \theta_k G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^\star\right) - \theta_k \frac{\mu_k^\star}{2}\left\|x^\star - y^{(k)}\right\|_2^2 - \frac{1}{2L_k}\left\| G_{L_k}\left(y^{(k)}\right)\right\|_2^2 \\
& = (1-\theta_k) f\left(x^{(k)}\right) + (1-\theta_k)\frac{\theta_k\gamma_k}{\gamma_{k+1}} G_{L_k}\left(y^{(k)}\right)^T \left(z^{(k)} - y^{(k)}\right) \\
& \quad + \theta_k f^\star + \theta_k G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^\star\right) - \theta_k \frac{\mu_k^\star}{2}\left\|x^\star - y^{(k)}\right\|_2^2 - \frac{1}{2L_k}\left\| G_{L_k}\left(y^{(k)}\right)\right\|_2^2 \\
& \leq (1-\theta_k) f\left(x^{(k)}\right) + (1-\theta_k)\frac{\theta_k\gamma_k}{\gamma_{k+1}} G_{L_k}\left(y^{(k)}\right)^T \left(z^{(k)} - y^{(k)}\right) \\
& \quad + \theta_k f^\star + \theta_k G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^\star\right) - \theta_k \frac{\mu_k^\star}{2}\left\|x^\star - y^{(k)}\right\|_2^2 - \frac{1}{2L_k}\left\| G_{L_k}\left(y^{(k)}\right)\right\|_2^2 \\
& \quad + \frac{(1-\theta_k)\theta_k\gamma_k\mu_k}{2\gamma_{k+1}}\left\|z^{(k)} - y^{(k)}\right\|_2^2 \\
& = (1-\theta_k) f\left(x^{(k)}\right) + (1-\theta_k)\frac{\theta_k\gamma_k}{\gamma_{k+1}} G_{L_k}\left(y^{(k)}\right)^T \left(z^{(k)} - y^{(k)}\right) \\
& \quad + \theta_k f^\star + \theta_k G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^\star\right) - \theta_k \frac{\mu_k^\star}{2}\left\|x^\star - y^{(k)}\right\|_2^2 - \frac{1}{2L_k}\left\| G_{L_k}\left(y^{(k)}\right)\right\|_2^2 \\
& \quad + \left((1-\theta_k)\frac{\gamma_k}{2} - \frac{1}{2\gamma_{k+1}}(1-\theta_k)^2\gamma_k^2\right)\left\|z^{(k)} - y^{(k)}\right\|_2^2 \\
& = (1-\theta_k) f\left(x^{(k)}\right) + (1-\theta_k)\frac{\gamma_k}{2}\left\|z^{(k)} - y^{(k)}\right\|_2^2 - \frac{\gamma_{k+1}}{2}\left\|z^{(k+1)} - y^{(k)}\right\|_2^2 \\
& \quad + \theta_k f^\star + \theta_k G_{L_k}\left(y^{(k)}\right)^T \left(y^{(k)} - x^\star\right) - \theta_k \frac{\mu_k^\star}{2}\left\|x^\star - y^{(k)}\right\|_2^2
\end{aligned}$$

$$= (1-\theta_k) f\left(x^{(k)}\right) + \theta_k f^\star - \theta_k \frac{\mu_k^\star}{2} \left\|x^\star - y^{(k)}\right\|_2^2$$
$$+ (1-\theta_k)\frac{\gamma_k}{2}\|z^{(k)} - x^\star\|_2^2 - \frac{\gamma_{k+1}}{2}\left\|z^{(k+1)} - x^\star\right\|_2^2 + \theta_k \frac{\mu_k}{2}\left\|y^{(k)} - x^\star\right\|_2^2,$$

where we have used (A.1), a trivial inequality, (A.4) , (A.3), (A.2), and (A.5). If $\mu_k \leq \mu_k^\star$ then

$$f\left(x^{(k+1)}\right) - f^\star + \frac{\gamma_{k+1}}{2}\left\|z^{(k+1)} - x^\star\right\|_2^2 \leq (1-\theta_k)\left(f\left(x^{(k)}\right) - f^\star + \frac{\gamma_k}{2}\left\|z^{(k)} - x^\star\right\|_2^2\right) \tag{A.8}$$

in which case we can combine the bounds to obtain

$$f\left(x^{(k)}\right) - f^\star + \frac{\gamma_k}{2}\left\|z^{(k)} - x^\star\right\|_2^2 \leq \left(\prod_{i=0}^{k-1}(1-\theta_i)\right)\left(f\left(x^{(0)}\right) - f^\star + \frac{\gamma_k}{2}\left\|z^{(0)} - x^\star\right\|_2^2\right), \tag{A.9}$$

where we have also used $x^{(0)} = y^{(0)}$ and (A.1) to obtain $x^{(0)} = z^{(0)}$. For completeness, we will show why this is an optimal first-order method. Let $\mu_k = \mu_k^\star = \mu$ and $L_k = L$. If $\gamma_0 \geq \mu$ then using (A.2) we obtain $\gamma_{k+1} \geq \mu$ and $\theta_k \geq \sqrt{\mu/L} = \sqrt{Q^{-1}}$. Simultaneously, we also have $\prod_{i=0}^{k-1}(1-\theta_k) \leq \frac{4L}{(2\sqrt{L}+k\sqrt{\gamma_0})^2}$ [30, Lemma 2.2.4], and the bound is then

$$f\left(x^{(k)}\right) - f^\star$$
$$\leq \min\left(\left(1-\sqrt{Q^{-1}}\right)^k, \frac{4L}{(2\sqrt{L}+k\sqrt{\gamma_0})^2}\right)\left(f\left(x^{(0)}\right) - f^\star + \frac{\gamma_0}{2}\left\|x^{(0)} - x^\star\right\|_2^2\right). \tag{A.10}$$

This is the optimal convergence rate for the class $\mathcal{F}_{0,L}$ and $\mathcal{F}_{\mu,L}$ simultaneously [28, 30].

Appendix B: Complexity analysis

In this Appendix we prove Theorem 6.1, i.e., we derive the complexity for reaching an ϵ-suboptimal solution for the algorithm *UPN*. The total worst-case complexity is given by (a) the complexity for the worst case number of restarts and (b) the worst-case complexity for a successful termination.

With a slight abuse of notation in this Appendix, $\mu_{k,r}$ denotes the kth iterate in the rth restart stage, and similarly for $L_{k,r}$, $\tilde{L}_{k,r}$, $x^{(k,r)}$, etc. The value $\mu_{0,0}$ is the initial estimate of the strong convexity parameter when no restart has occurred. In the worst case, the heuristic choice in (6.4) never reduces μ_k, such that we have $\mu_{k,r} = \mu_{0,r}$. Then a total of R restarts are required, where

$$\rho_\mu^R \mu_{0,0} = \mu_{0,R} \leq \mu \quad \Longleftrightarrow \quad R \geq \log(\mu_{0,0}/\mu)/\log(1/\rho_\mu).$$

In the following analysis we shall make use of the relation

$$\exp\left(-\frac{n}{\delta^{-1}-1}\right) \leq (1-\delta)^n \leq \exp\left(-\frac{n}{\delta^{-1}}\right), \quad 0<\delta<1,\ n\geq 0.$$

B.1 Termination complexity

After sufficiently many restarts (at most R), $\mu_{0,r}$ will be sufficiently small in which case (6.9) holds and we obtain

$$\begin{aligned}
&\left\|G_{\tilde{L}_{k+1,r}}\left(x^{(k+1,r)}\right)\right\|_2^2 \\
&\quad\leq \prod_{i=1}^{k}\left(1-\sqrt{\frac{\mu_{i,r}}{L_{i,r}}}\right)\left(\frac{4\tilde{L}_{k+1,r}}{\mu_{k,r}}-\frac{2\tilde{L}_{k+1,r}}{2L_{0,r}}+\frac{2\tilde{L}_{k+1,r}\gamma_{1,r}}{\mu_{k,r}^2}\right)\left\|G_{L_0}\left(x^{(0,r)}\right)\right\|_2^2 \\
&\quad\leq \left(1-\sqrt{\frac{\mu_{k,r}}{L_{k,r}}}\right)^k\left(\frac{4\tilde{L}_{k+1,r}}{\mu_{k,r}}-\frac{2\tilde{L}_{k+1,r}}{2L_{0,r}}+\frac{2\tilde{L}_{k+1,r}\gamma_{1,r}}{\mu_{k,r}^2}\right)\left\|G_{L_{0,r}}\left(x^{(0,r)}\right)\right\|_2^2 \\
&\quad\leq \exp\left(-\frac{k}{\sqrt{L_{k,r}/\mu_{k,r}}}\right)\left(\frac{4\tilde{L}_{k+1,r}}{\mu_{k,r}}-\frac{\tilde{L}_{k+1,r}}{L_{0,r}}+\frac{2\tilde{L}_{k+1,r}\gamma_{1,r}}{\mu_{k,r}^2}\right)\left\|G_{L_{0,r}}\left(x^{(0,r)}\right)\right\|_2^2,
\end{aligned}$$

where we have used $L_{i,r}\leq L_{i+1,r}$ and $\mu_{i,r}\geq\mu_{i+1,r}$. To guarantee $\|G_{\tilde{L}_{k+1,r}}(x^{(k+1,r)})\|_2\leq\bar{\epsilon}$ we require the latter bound to be smaller than $\bar{\epsilon}^2$, i.e.,

$$\begin{aligned}
&\left\|G_{\tilde{L}_{k+1,r}}(x^{(k+1,r)})\right\|_2^2 \\
&\quad\leq \exp\left(-\frac{k}{\sqrt{L_{k,r}/\mu_{k,r}}}\right)\left(\frac{4\tilde{L}_{k+1,r}}{\mu_{k,r}}-\frac{\tilde{L}_{k+1}}{L_{0,r}}+\frac{2\tilde{L}_{k+1,r}\gamma_{1,r}}{\mu_{k,r}^2}\right)\left\|G_{L_{0,r}}(x^{(0,r)})\right\|_2^2\leq\bar{\epsilon}^2.
\end{aligned}$$

Solving for k, we obtain

$$k=\mathcal{O}\left(\sqrt{Q}\log Q\right)+\mathcal{O}\left(\sqrt{Q}\log\bar{\epsilon}^{-1}\right), \tag{B.1}$$

where we have used $\mathcal{O}(\sqrt{L_{k,r}/\mu_{k,r}})=\mathcal{O}(\sqrt{\tilde{L}_{k+1,r}/\mu_{k,r}})=\mathcal{O}(\sqrt{Q})$.

B.2 Restart complexity

How many iterations are needed before we can detect that a restart is needed? The restart detection rule (6.9) gives

$$\begin{aligned}
&\left\|G_{\tilde{L}_{k+1,r}}\left(x^{(k+1,r)}\right)\right\|_2^2 \\
&\quad> \prod_{i=1}^{k}\left(1-\sqrt{\frac{\mu_{i,r}}{L_{i,r}}}\right)\left(\frac{4\tilde{L}_{k+1,r}}{\mu_{k,r}}-\frac{2\tilde{L}_{k+1,r}}{2L_{0,r}}+\frac{2\tilde{L}_{k+1,r}\gamma_{1,r}}{\mu_{k,r}^2}\right)\left\|G_{L_{0,r}}\left(x^{(0,r)}\right)\right\|_2^2
\end{aligned}$$

$$\geq \left(1-\sqrt{\frac{\mu_{1,r}}{L_{1,r}}}\right)^{k}\left(\frac{4\tilde{L}_{1,r}}{\mu_{1,r}}-\frac{2\tilde{L}_{1,r}}{2L_{0,r}}+\frac{2\tilde{L}_{1,r}\gamma_{1,r}}{\mu_{1,r}^2}\right)\left\|G_{L_{0,r}}\left(x^{(0,r)}\right)\right\|_2^2$$

$$\geq \exp\left(-\frac{k}{\sqrt{L_{1,r}/\mu_{1,r}}-1}\right)\left(\frac{4L_{1,r}}{\mu_{1,r}}-\frac{2L_{1,r}}{2L_{0,r}}+\frac{2L_{1,r}\gamma_{1,r}}{\mu_{1,r}^2}\right)\left\|G_{L_{0,r}}\left(x^{(0,r)}\right)\right\|_2^2,$$

where we have used $L_{i,r}\leq L_{i+1,r}$, $L_{i,r}\leq \tilde{L}_{i+1,r}$ and $\mu_{i,r}\geq \mu_{i+1,r}$. Solving for k, we obtain

$$k>\left(\sqrt{\frac{L_{1,r}}{\mu_{1,r}}}-1\right)\left(\log\left(\frac{4L_{1,r}}{\mu_{1,r}}-\frac{L_{1,r}}{L_{0,r}}+\frac{4\gamma_{1,r}L_{1,r}}{\mu_{1,r}^2}\right)+\log\frac{\|G_{L_{0,r}}(x^{(0,r)})\|_2^2}{\|G_{\tilde{L}_{k+1,r}}(x^{(k+1,r)})\|_2^2}\right). \tag{B.2}$$

Since we do not terminate but restart, we have $\|G_{\tilde{L}_{k+1,r}}(x^{(k+1,r)})\|_2\geq\bar{\epsilon}$. After r restarts, in order to satisfy (B.2) we must have k of the order

$$\mathcal{O}\left(\sqrt{Q_r}\right)\mathcal{O}(\log Q_r)+\mathcal{O}\left(\sqrt{Q_r}\right)\mathcal{O}\left(\log\bar{\epsilon}^{-1}\right),$$

where

$$Q_r=\mathcal{O}\left(\frac{L_{1,r}}{\mu_{1,r}}\right)=\mathcal{O}\left(\rho_\mu^{R-r}Q\right).$$

The worst-case number of iterations for running R restarts is then given by

$$\begin{aligned}
&\sum_{r=0}^{R}\mathcal{O}\left(\sqrt{Q\rho_\mu^{R-r}}\right)\mathcal{O}\left(\log Q\rho_\mu^{R-r}\right)+\mathcal{O}\left(\sqrt{Q\rho_\mu^{R-r}}\right)\mathcal{O}\left(\log\bar{\epsilon}^{-1}\right)\\
&\quad=\sum_{i=0}^{R}\mathcal{O}\left(\sqrt{Q\rho_\mu^{i}}\right)\mathcal{O}\left(\log Q\rho_\mu^{i}\right)+\mathcal{O}\left(\sqrt{Q\rho_\mu^{i}}\right)\mathcal{O}\left(\log\bar{\epsilon}^{-1}\right)\\
&\quad=\mathcal{O}\left(\sqrt{Q}\right)\left\{\sum_{i=0}^{R}\mathcal{O}\left(\sqrt{\rho_\mu^{i}}\right)\left[\mathcal{O}\left(\log Q\rho_\mu^{i}\right)+\mathcal{O}\left(\log\bar{\epsilon}^{-1}\right)\right]\right\}\\
&\quad=\mathcal{O}\left(\sqrt{Q}\right)\left\{\sum_{i=0}^{R}\mathcal{O}\left(\sqrt{\rho_\mu^{i}}\right)\left[\mathcal{O}(\log Q)+\mathcal{O}\left(\log\bar{\epsilon}^{-1}\right)\right]\right\}\\
&\quad=\mathcal{O}\left(\sqrt{Q}\right)\left\{\mathcal{O}(1)\left[\mathcal{O}(\log Q)+\mathcal{O}\left(\log\bar{\epsilon}^{-1}\right)\right]\right\}\\
&\quad=\mathcal{O}\left(\sqrt{Q}\right)\mathcal{O}(\log Q)+\mathcal{O}\left(\sqrt{Q}\right)\mathcal{O}\left(\log\bar{\epsilon}^{-1}\right)\\
&\quad=\mathcal{O}\left(\sqrt{Q}\log Q\right)+\mathcal{O}\left(\sqrt{Q}\log\bar{\epsilon}^{-1}\right),
\end{aligned}\tag{B.3}$$

where we have used

$$\sum_{i=0}^{R}\mathcal{O}\left(\sqrt{\rho_\mu^{i}}\right)=\sum_{i=0}^{R}\mathcal{O}\left(\sqrt{\rho_\mu}^{\,i}\right)=\mathcal{O}\left(\frac{1-\sqrt{\rho_\mu^{R+1}}}{1-\sqrt{\rho_\mu}}\right)=\mathcal{O}(1).$$

B.3 Total complexity

The total iteration complexity of *UPN* is given by (B.3) plus (B.1):

$$\mathcal{O}\big(\sqrt{Q}\log Q\big) + \mathcal{O}\big(\sqrt{Q}\log \bar{\epsilon}^{-1}\big). \tag{B.4}$$

It is common to write the iteration complexity in terms of reaching an ϵ-suboptimal solution satisfying $f(x) - f^\star \leq \epsilon$. This is different from the stopping criteria $\|G_{\tilde{L}_{k+1,r}}(x^{(k+1,r)})\|_2 \leq \bar{\epsilon}$ or $\|G_{L_{k,r}}(y^{(k,r)})\|_2 \leq \bar{\epsilon}$ used in the *UPN* algorithm. Consequently, we will derive a relation between ϵ and $\bar{\epsilon}$. Using Lemmas 5.1 and 5.2, in case we stop using $\|G_{L_{k,r}}(y^{(k,r)})\|_2 \leq \bar{\epsilon}$ we obtain

$$f\big(x^{(k+1,r)}\big) - f^\star \leq \left(\frac{2}{\mu} - \frac{1}{2L_{k,r}}\right)\big\|G_{L_{k,r}}\big(y^{(k,r)}\big)\big\|_2^2 \leq \frac{2}{\mu}\big\|G_{L_{k,r}}\big(y^{(k,r)}\big)\big\|_2^2 \leq \frac{2}{\mu}\bar{\epsilon}^2,$$

and in case we stop using $\|G_{\tilde{L}_{k+1,r}}(x^{(k+1,r)})\|_2 \leq \bar{\epsilon}$, we obtain

$$\begin{aligned} f\big(\tilde{x}^{(k+1,r)}\big) - f^\star &\leq \left(\frac{2}{\mu} - \frac{1}{2\tilde{L}_{k+1,r}}\right)\big\|G_{\tilde{L}_{k+1,r}}\big(x^{(k+1,r)}\big)\big\|_2^2 \leq \frac{2}{\mu}\big\|G_{\tilde{L}_{k+1,r}}\big(x^{(k+1,r)}\big)\big\|_2^2 \\ &\leq \frac{2}{\mu}\bar{\epsilon}^2. \end{aligned}$$

To return with either $f(\tilde{x}^{(k+1,r)}) - f^\star \leq \epsilon$ or $f(x^{(k+1,r)}) - f^\star \leq \epsilon$ we require the latter bounds to hold and thus select $(2/\mu)\bar{\epsilon}^2 = \epsilon$. The iteration complexity of the algorithm in terms of ϵ is then

$$\begin{aligned} &\mathcal{O}\big(\sqrt{Q}\log Q\big) + \mathcal{O}\big(\sqrt{Q}\log\big((\mu\epsilon)^{-1}\big)\big) \\ &\quad = \mathcal{O}(\sqrt{Q}\log Q) + \mathcal{O}\big(\sqrt{Q}\log \mu^{-1}\big) + \mathcal{O}\big(\sqrt{Q}\log \epsilon^{-1}\big) \\ &\quad = \mathcal{O}(\sqrt{Q}\log Q) + \mathcal{O}\big(\sqrt{Q}\log \epsilon^{-1}\big), \end{aligned}$$

where we have used $\mathcal{O}(1/\mu) = \mathcal{O}(L/\mu) = \mathcal{O}(Q)$.

References

1. Alter, F., Durand, S., Froment, J.: Adapted total variation for artifact free decompression of JPEG images. J. Math. Imaging Vis. **23**, 199–211 (2005)
2. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal. **8**, 141–148 (1988)
3. Beck, A., Teboulle, M.: Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. IEEE Trans. Image Process. **18**, 2419–2434 (2009)
4. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**, 183–202 (2009)
5. Becker, S., Bobin, J., Candès, E.J.: NESTA: a fast and accurate first-order method for sparse recovery. SIAM J. Imaging Sci. **4**(1), 1–39 (2011)
6. Becker, S., Candès, E.J., Grant, M.: Templates for convex cone problems with applications to sparse signal recovery. Math. Program. Comput. **3**, 165–218 (2011)

7. Bioucas-Dias, J.M., Figueiredo, M.A.T.: A new TwIST: two-step iterative shrinkage/thresholding algorithms for image restoration. IEEE Trans. Image Process. **16**(12), 2992–3004 (2007)
8. Birgin, E.G., Martínez, J.M., Raydan, M.: Nonmonotone spectral projected gradient methods on convex sets. SIAM J. Optim. **10**, 1196–1211 (2000)
9. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
10. Chambolle, A.: An algorithm for total variation minimization and applications. J. Math. Imaging Vis. **20**, 89–97 (2004)
11. Chambolle, A.: Total variation minimization and a class of binary MRF models. In: Rangarajan, A., Vemuri, B., Yuille, A.L. (eds.) Energy Minimization Methods in Computer Vision and Pattern Recognition. Lecture Notes in Computer Science, vol. 3757, pp. 136–152. Springer, Berlin (2005)
12. Chambolle, A., Pock, T.: A first-order primal-dual algorithm for convex problems with applications to imaging. J. Math. Imaging Vis. **40**, 120–145 (2011)
13. Chan, T.F., Golub, G.H., Mulet, P.: A nonlinear primal-dual method for total variation-based image restoration. SIAM J. Sci. Comput. **20**, 1964–1977 (1998)
14. Chan, T.F., Shen, J.: Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods. SIAM, Philadelphia (2005)
15. Combettes, P.L., Luo, J.: An adaptive level set method for nondifferentiable constrained image recovery. IEEE Trans. Image Process. **11**, 1295–1304 (2002)
16. Dahl, J., Hansen, P.C., Jensen, S.H., Jensen, T.L.: Algorithms and software for total variation image reconstruction via first-order methods. Numer. Algorithms **53**, 67–92 (2010)
17. Dai, Y.H., Liao, L.Z.: R-linear convergence of the Barzilai and Borwein gradient method. IMA J. Numer. Anal. **22**, 1–10 (2002)
18. Darbon, J., Sigelle, M.: Image restoration with discrete constrained total variation—Part I: Fast and exact optimization. J. Math. Imaging Vis. **26**, 261–276 (2006)
19. Fletcher, R.: Low storage methods for unconstrained optimization. In: Allgower, E.L., Georg, K. (eds.) Computational Solution of Nonlinear Systems of Equations, pp. 165–179. Am. Math. Soc., Providence (1990)
20. Goldfarb, D., Yin, W.: Second-order cone programming methods for total variation-based image restoration. SIAM J. Sci. Comput. **27**, 622–645 (2005)
21. Grippo, L., Lampariello, F., Lucidi, S.: A nonmonotone line search technique for Newton's method. SIAM J. Numer. Anal. **23**, 707–716 (1986)
22. Hansen, P.C.: Discrete Inverse Problems: Insight and Algorithms. SIAM, Philadelphia (2010)
23. Herman, G.T.: Fundamentals of Computerized Tomography: Image Reconstruction from Projections, 2nd edn. Springer, New York (2009)
24. Hintermüller, M., Stadler, G.: An infeasible primal-dual algorithm for total bounded variation-based INF-convolution-type image restoration. SIAM J. Sci. Comput. **28**, 1–23 (2006)
25. Jørgensen, J.H.: Tomobox (2010). www.mathworks.com/matlabcentral/fileexchange/28496-tomobox
26. Kak, A.C., Slaney, M.: Principles of Computerized Tomographic Imaging. SIAM, Philadelphia (2001)
27. Lan, G., Lu, Z., Monteiro, R.D.C.: Primal-dual first-order methods with $O(1/\epsilon)$ iteration-complexity for cone programming. Math. Program., Ser. A **126**(1), 1–29 (2011)
28. Nemirovsky, A.S., Yudin, D.B.: Problem Complexity and Method Efficiency in Optimization. Wiley, New York (1983)
29. Nesterov, Y.: A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. Sov. Math. Dokl. **269**, 543–547 (1983)
30. Nesterov, Y.: Introductory Lectures on Convex Optimization. Kluwer Academic, Dordrecht (2004)
31. Nesterov, Y.: Smooth minimization of nonsmooth functions. Math. Program., Ser. A **103**, 127–152 (2005)
32. Nesterov, Y.: Gradient methods for minimizing composite objective function (2007). CORE Discussion Paper No 2007076, www.ecore.be/DPs/dp_1191313936.pdf
33. Nolet, G. (ed.): Seismic Tomography with Applications in Global Seismology and Exploration Geophysics. Reidel, Dordrecht (1987)
34. Parrish, R.: getLebedevSphere (2010). www.mathworks.com/matlabcentral/fileexchange/27097-getlebedevsphere
35. Raydan, M.: The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. SIAM J. Optim. **7**, 26–33 (1997)
36. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D **60**, 259–268 (1992)

37. Schabel, M.: 3D Shepp-Logan phantom (2006). www.mathworks.com/matlabcentral/fileexchange/9416-3d-shepp-logan-phantom
38. Tseng, P.: On accelerated proximal gradient methods for convex-concave optimization. Manuscript (2008). www.math.washington.edu/~tseng/papers/apgm.pdf
39. Vandenberghe, L.: Optimization methods for large-scale systems. Lecture Notes (2009). www.ee.ucla.edu/~vandenbe/ee236c.html
40. Vogel, C.R., Oman, M.E.: Iterative methods for total variation denoising. SIAM J. Sci. Comput. **17**, 227–238 (1996)
41. Weiss, P., Blanc-Féraud, L., Aubert, G.: Efficient schemes for total variation minimization under constraints in image processing. SIAM J. Sci. Comput. **31**, 2047–2080 (2009)
42. Zhu, M., Wright, S.J., Chan, T.F.: Duality-based algorithms for total-variation-regularized image restoration. Comput. Optim. Appl. (2008). doi:10.1007/s10589-008-9225-2