HHS Public Access

Author manuscript

Numer Algorithms. Author manuscript; available in PMC 2019 May 06.

Published in final edited form as:

Numer Algorithms. 2019 April ; 80(4): 1219–1240. doi:10.1007/s11075-018-0524-0.

Derivative-free superiorization with component-wise perturbations

Yair Censor[1], **Howard Heaton**[2], and **Reinhard Schulte**[3]

[1]Department of Mathematics, University of Haifa, Mt. Carmel, 3498838, Haifa, Israel

[2]Department of Mathematics, University of California Los Angeles, Los Angeles, CA, 90095, USA

[3]Division of Biomedical Engineering Sciences, Department of Basic Sciences, School of Medicine, Loma Linda University, Loma Linda, CA, 92350, USA

Abstract

Superiorization reduces, not necessarily minimizes, the value of a target function while seeking constraints compatibility. This is done by taking a solely feasibility-seeking algorithm, analyzing its perturbation resilience, and proactively perturbing its iterates accordingly to steer them toward a feasible point with reduced value of the target function. When the perturbation steps are computationally efficient, this enables generation of a superior result with essentially the same computational cost as that of the original feasibility-seeking algorithm. In this work, we refine previous formulations of the superiorization method to create a more general framework, enabling target function reduction steps that do not require partial derivatives of the target function. In perturbations that use partial derivatives, the step-sizes in the perturbation phase of the superiorization method are chosen independently from the choice of the nonascent directions. This is no longer true when component-wise perturbations are employed. In that case, the step-sizes must be linked to the choice of the nonascent direction in every step. Besides presenting and validating these notions, we give a computational demonstration of superiorization with component-wise perturbations for a problem of computerized tomography image reconstruction.

Keywords

Superiorization; Derivative-free; Component-wise perturbations; Image reconstruction; Feasibility-seeking; Perturbation resilience

1 Introduction

In this introduction, we first describe briefly the superiorization methodology and mention some of the previous work on it. Then we describe in general terms the proposed approach of derivative-free component-wise perturbations and the computational demonstration that we report about here.

Correspondence to: Yair Censor.

The superiorization methodology

The superiorization methodology (SM) is an algorithmic scheme that can be considered to reside between feasibility-seeking and constrained minimization. Rather than attempting to solve a full-fledged minimization problem, the SM takes a feasibility-seeking algorithm and proactively steers its iterates to find a feasible point that is superior, though not necessarily optimal, with respect to the value of a target function, to the output obtained by the feasibility-seeking algorithm. This approach originates from the discovery that many feasibility-seeking algorithms are perturbation resilient in the sense that, even if certain kinds of changes are made at the end of each iterative step, the algorithms still produce constraints-compatible solutions [3, 7, 14, 28].

When the steps to compute perturbations of the iterates of a feasibility-seeking algorithm to reduce the target function value are computationally efficient, a superior result is obtained with essentially the same computational cost as that of the original feasibility-seeking algorithm. Thus, the SM is useful for constrained minimization problems where either an exact algorithm has not been discovered or existing exact algorithms are exceedingly time-consuming or require too much computer space for realistically large problems to be solved on commonplace computers. In these cases, the SM enables efficient feasibility-seeking algorithms, which provide constraints-compatible solutions, to be turned into efficient algorithms that will be practically useful from the point of view of reducing the value of the underlying target function.

Previous work on superiorization

In the SM, the superiorized version of an iterative feasibility-seeking algorithm consists of two parts. The first part performs perturbations that aim to reduce the value of the target function. The other is a part where the operator for the feasibility-seeking algorithm is applied. As noted in [8], several works have made use of this idea with proposed algorithms for exact constrained minimization (e.g., see [2, 12, 13, 15, 21, 22, 29, 33, 34]). However, these approaches are unable to do what is accomplished by the superiorization approach, which is to automatically generate a heuristic constrained optimization algorithm from an iterative feasibility-seeking algorithm. The underlying idea of the SM is quite general and provides application in many areas. The mathematical principles of the SM over general consistent “problem structures” with the notion of bounded perturbation resilience were formulated in [7]. The framework of the SM was extended to the inconsistent case by using the notion of strong perturbation resilience in [5, 8]. In [8], the efficacy of the SM was also demonstrated by comparing it with the performance of the projected subgradient method for constrained minimization problems.

A comprehensive overview of the state of the art and current research on superiorization appears in our continuously updated bibliography Internet page which currently contains 76 items [4]. Research works in this bibliography include a variety of reports ranging from new applications to new mathematical results of the foundation of superiorization. A special issue entitled “Superiorization: Theory and Applications” of the journal *Inverse Problems* has recently appeared [9].

Derivative-free component-wise perturbations

In the SM, the perturbation part interlaces target function reduction steps into the feasibility-seeking algorithm. Until now, generation of nonascent directions, used for target function reduction steps, was mostly based on theorems such as [24, Theorem 1] and its variants such as [16, Theorem 1] and [17, unnumbered Theorem on page 7]. All these theorems make the assumption on the constructed nonascent direction g whose existence is guaranteed by the theorem that "Let $g \in \mathbb{R}^J$ satisfy the property: For $1 \leq j \leq J$, if the jth component g_j of g is not zero, then the partial derivative $\frac{\partial \phi}{\partial x_j}(x)$ of ϕ at x exists and its value is g_j". Thus, ϕ must have at least one partial derivative (which is nonzero) at points in the domain of ϕ. Otherwise, these theorems would apply only to the zero vector, which is a useless nonascending vector because it renders the SM ineffective.

To summarize this point, the definition of nonascending vectors (see Definition 3 below) does not require differentiability but in almost all existing works ϕ must obey the condition to have at least one partial derivative (which is nonzero) at points in its domain. The paper [18] is a possible exception since no derivatives are used there, but it refers only to the specific ℓ_1-norm and still does not answer the general question of how to implement the SM in cases when the abovementioned theorems do not apply due to total lack of partial derivatives. This question makes sense for cases in which only target function values can be calculated but nothing else about the function, such as, functions that are defined by tables of values. Many such derivative-free objective functions are available in the field of derivative-free optimization, see, e.g., [30].

In this paper, we offer an approach how to handle target functions ϕ which do not obey the above condition of having at least one nonzero partial derivative or for which one is unable to verify it. Our main contribution is to propose the use of *component-wise perturbations* within the SM. When component-wise perturbations are employed, the classical notion of nonascent does not necessarily apply because if a point at a certain distance from a given point, along some coordinate, has a lower target function value, it does not guarantee that any other point in the neighborhood of the given point does so. In such a case, the step-sizes of the perturbations must be properly linked to the choice of the nonascent direction in every step, giving rise to a new notion of "local nonascent" of the target function (see Definition 4 below). These notions of local nonascent and component-wise perturbations have not been used in superiorization until now and they have both theoretical and practical significance. Such formulation of superiorization is logically more generally applicable than previously studied superiorization methods since it allows a wider selection of target function reduction steps and enables the SM to be applied with target functions for which not even one partial derivative is available.

Computational demonstration

By considering component-wise perturbations, we generalize previous superiorization schemes to enable use of a wider selection of methods for step-wise reduction of the target function. As a first step in showing that component-wise perturbations in the SM work, we present a new superiorization scheme for reducing total variation (TV) during image

reconstruction, i.e., total variation superiorization (TVS). We decided to do component-wise perturbations on iterates to reduce the TV function although it has calculable partial derivatives. This way, we have something to compare our results with. Surprisingly, we found that, even for this sub-differentiable target function, component-wise perturbations can outperform negative gradient perturbations within the SM. This is not to say or claim that component-wise perturbations always outperform perturbations based on derivative information. On the contrary, it is expected that gradient-based perturbations will, in general, be more efficient in the SM. The true merit of component-wise perturbations is that it opens the door for derivative-free perturbations in the SM, e.g., by applying it to superiorization of biological merit functions in intensity-modulated radiation therapy (IMRT). Another computational demonstration of derivative-free perturbations in the SM, based on the ideas presented here that we communicated to the authors, appears in [19, Section 4.3].

What is in this paper

The remainder of this work is outlined as follows. In Section 2, we present the mathematical framework of the SM in the context of solving a convex feasibility problem, which is followed by our proposed scheme for TVS with component-wise perturbations in Section 3. Then Section 4 provides an example of the specific proposed scheme for TVS applied to image reconstruction, juxtaposing our approach with a negative gradient-based approach based on previous works (e.g., [8, 24]). Discussion and conclusions are provided, respectively, in Section 5 and Section 6.

2 Superiorization with local nonascent

2.1 The superiorization framework

In order to make the paper to some extent self-contained, we briefly review the SM framework as developed in earlier publications, see, e.g., [5, 7, 8, 10, 24]. Given a collection of closed convex subsets $C_i \subset \mathbb{R}^L$ for $i = 1, 2, \ldots, m$, in the L-dimensional Euclidean space, the convex feasibility problem (CFP) is to find a point $x^* \in \cap_{i=1}^{m} C_i$. In the superiorization method, one seeks a solution to the CFP that is superior, although not necessarily optimal, with respect to some target function ϕ. A superior solution is here considered to be a better solution, with respect to the target function value, than that which would have been found by the given feasibility-seeking algorithm without superiorization steps. Suppose that we have a feasibility-seeking algorithmic operator $\mathcal{A} : \mathbb{R}^L \to \mathbb{R}^L$ with which we define an iterative process for the solution of a CFP

$$x^{k+1} = \mathcal{A}(x^k) \quad \text{for all } k \geq 0 \text{ with arbitrary } x^0 \in \mathbb{R}^L. \quad (1)$$

This process is called “the basic algorithm” and the sequence of iterates it produces can be evaluated using a notion of proximity to the sets of the CFP. Let $\{C_i\}_{i=1}^{m}$ be a finite family of closed convex sets and suppose the existence of a nonempty subset $\Lambda \subset \mathbb{R}^L$ such that $C_i \subset \Lambda$ for all $i = 1, 2, \ldots, m$,. We denote this CFP by T and associate with it a *proximity*

function $\mathrm{Prox}_T: \Lambda \to \mathbb{R}_+$ that indicates how compatible an $x \in \Lambda$ is with the constraints. Given any positive ε, any point $x \in \Lambda$ for which $\mathrm{Prox}_T(x) \leq \varepsilon$ is called an *ε-compatible* solution of the CFP. Thus, the basic algorithm can be terminated when the proximity function gives a value less than some positive ε. We define this as the *ε-output of a sequence* of points generated by an iterative algorithmic operator, see [7, page 5].

Definition 1—Given a family of constraints sets $\{C_i\}_{i=1}^m$ of a CFP T, a proximity function $\mathrm{Prox}_T: \Lambda \to \mathbb{R}_+$, a sequence $\{x^k\}_{k=0}^\infty \subset \Lambda$, and an $\varepsilon > 0$, an element x^K of the sequence which has the properties:

- **i.** $\mathrm{Prox}_T(x^K) \leq \varepsilon$, and
- **ii.** $\mathrm{Prox}_T(x^k) > \varepsilon$ for all $0 \leq k < K$,

is called the *ε-output of the sequence* $\{x^k\}_{k=0}^\infty$ with respect to the pair (T, Prox_T).

The ε-output x^K of a sequence is denoted by $\mathcal{O}\left(T, \varepsilon, \{x^k\}_{k=0}^\infty\right)$. Such an output may not exist; however, when it does, it is unique. Furthermore, when the sequence $\{x^k\}_{k=0}^\infty$ is generated by a basic algorithm for solving a CFP, the point $\mathcal{O}\left(T, \varepsilon, \{x^k\}_{k=0}^\infty\right)$ gives the output of the basic algorithm when the stopping criterion is ε-compatibility.

The following version of the SM, presented in [24], is known as strong superiorization. (See [5] for a review of strong and weak superiorization.) Here the solution set C of the CFP T may be empty and solving the CFP is then understood to mean finding a point that is within a given proximity of the constraints. The "superiorized version of a basic algorithm" is created by taking advantage of the fact that successive iterates of the basic algorithm can, in some instances, be systematically perturbed without losing overall convergence of the iterates. Our problem at hand is stated as follows.

Problem 1—Let $\{C_i\}_{i=1}^m$ be a family of closed convex sets of a CFP T, $C_i \subseteq \Lambda \subseteq \mathbb{R}^L$ for all i, let $\phi: \mathbb{R}^L \to \mathbb{R}$ be a given target function and let $\mathcal{A}: \Lambda \to \mathbb{R}^L$ be an iterative algorithmic operator defining a basic algorithm for solving the associated CFP. The *function reduction problem* is to use a superiorized version of the basic algorithm to find a point x^* that is ε-compatible with C and has a lesser value of the function ϕ than that of another ε-compatible point that would have been obtained by applying the basic algorithm alone.

Strong perturbation resilience is a property that describes the ability of a basic algorithm to be perturbed and not lose its ability to yield an *ε-compatible* solution of the CFP. This notion was termed "bounded perturbation resilience" in [24, Subsection II.C] and is defined as follows.

Definition 2—Assume we are given family of constraints $\{C_i\}_{i=1}^m$ of a CFP T, a proximity function Prox_T, an algorithmic operator $\mathcal{A}$, and an $x^0 \in \Lambda$. We use $\{x^k\}_{k=0}^\infty$ to denote the

sequence generated by the basic algorithm when it is initialized at x^0. The basic algorithm is said to be *strongly perturbation resilient* if the following hold:

i. there exist an $\varepsilon > 0$ such that the ε-output $\mathcal{O}\left(T, \varepsilon, \{x^k\}_{k=0}^{\infty}\right)$ exists for every $x^0 \in \Lambda$; and

ii. for every $\varepsilon > 0$, for which the ε-output $\mathcal{O}\left(T, \varepsilon, \{x^k\}_{k=0}^{\infty}\right)$ exists for every $x^0 \in \Lambda$, the ε'-output $\mathcal{O}\left(T, \varepsilon', \{y^k\}_{k=0}^{\infty}\right)$ also exists for every $\varepsilon' > \varepsilon$ and for every sequence $\{y^k\}_{k=0}^{\infty}$ generated by

$$y^{k+1} := \mathcal{A}\left(y^k + \beta_k v^k\right), \quad \text{for all } k \geq 0, \qquad (2)$$

where the vector sequence $\{v^k\}_{k=0}^{\infty}$ is bounded and the scalars $\{\beta_k\}_{k=0}^{\infty}$ are such that $\beta_k \geq 0$ for all $k \geq 0$ and the β_k are summable, i.e.,

$$\sum_{k=0}^{\infty} \beta_k < \infty . \qquad (3)$$

Sufficient conditions for strong perturbation resilience of a basic algorithm were proven in [24, Theorem 1].

2.2 Locally nonascending directions

The chief motivation to perturb iterates of a basic algorithm by sequences $\{\beta_k\}_{k=0}^{\infty}$ and $\{v^k\}_{k=0}^{\infty}$ is to reduce the values of the target function ϕ by employing directions of nonascent. Below we present the definition of nonascent that is in use in all works on the SM, see [8, Subsection II.D].

Definition 3—Given a function $\phi: \mathbb{R}^L \to \mathbb{R}$ and a point $y \in \mathbb{R}^L$, we say that a vector $d \in \mathbb{R}^L$ is *nonascending for ϕ at y* iff $\|d\| \leq 1$ ($\|\cdot\|$ denotes the Euclidean norm) and there is a $\delta > 0$ such that

$$\text{for all } \mu \in [0, \delta] \text{ we have } \phi(y + \mu d) \leq \phi(y) . \qquad (4)$$

This definition asserts that the nonascent inequality in (4) holds throughout the interval $[0, \delta]$. Under such circumstances, one can dictate the step-sizes in the perturbation phase of the SM independently of the choice of the nonascent vector. However, in order to employ component-wise perturbations or other perturbations which do not assume the availability of *any* partial derivative of the target function ϕ at y, we need to use a different definition of nonascent directions. We wish to allow the user to look in a neighborhood of the current

point y for a point where the target function value is lower without assuming that it is lower in an interval around the current point (this could be the case, e.g., with nonconvex target functions). To do this, the choice of nonascent direction and the perturbation step-size must be linked together to guarantee the reduced target function value. Therefore, we relax the above definition of nonascending vectors so that we may use a wider class of perturbations such as, in particular, component-wise perturbations.

Definition 4—Given a target function $\phi : \Delta \to \mathbb{R}$ where $\Delta \subset \mathbb{R}^L$, a point $y \in \Delta$, and a positive $\delta \in \mathbb{R}$, we say that $d \in \mathbb{R}^L$ is a *nonascending δ-bound direction for* ϕ at y if $\|d\| \leq \delta$ and $\phi(y + d) \leq \phi(y)$. The collection of all such vectors is called a *nonascending δ-ball* and is denoted by $\mathcal{B}_{\delta,\phi}(y)$, i.e.,

$$\mathcal{B}_{\delta,\phi}(y) := \{d \in \mathbb{R}^L \mid \|d\| \leq \delta,\ \phi(y + d) \leq \phi(y)\}. \quad (5)$$

The zero vector is contained in each nonascending δ-ball, i.e., $0 \in \mathcal{B}_{\delta,\phi}(y)$ for each $\delta > 0$ and $y \in \Delta$. This definition will allow us to use as a nonascent direction any vector $d \in \mathbb{R}^L$ at which $\phi(y + d) \leq \phi(y)$ holds, which might be detected by only function value calculations. This will be useful even when ϕ is not convex, or if we do component-wise search for a point with reduced target function value. Even functions defined by tabular representations are valid candidates for this nonascending δ-bound directions. We refer to this kind of nonascent as “local nonascent”.

2.3 Superiorized version of a basic algorithm with locally nonascending directions

The superiorized version of the basic algorithm presented here in Algorithm 1 assumes that we have a summable sequence $\{\eta_\ell\}_{\ell=0}^{\infty}$ of positive real numbers generated by $\eta_\ell := a^\ell$ where $a \in (0, 1)$, called kernel in [6], is user-chosen. This summable sequence is used to perturb iterates with the goal to reduce the value of the target function ϕ while maintaining convergence of the iterates to a solution of the original CFP. Each η_ℓ is used to generate a nonascending η_ℓ-ball for ϕ about iterates produced by applying the basic algorithmic operator $\mathcal{A}$. Points chosen from each of these η_ℓ-balls generate sequences $\{v^k\}_{k=0}^{\infty}$ and $\{\beta_k\}_{k=0}^{\infty}$, corresponding to the sequences in Definition 2. These sequences aim to steer the sequence to a lesser value of ϕ. This superiorized version of the basic algorithm also depends on a chosen initial point $\bar{y}$ and a sequence $\{N_k\}_{k=0}^{\infty}$ of positive integers bounded by some positive integer N. With this, the superiorized version of the basic algorithm is presented in Algorithm 1 by its pseudo-code.

Algorithm 1

Superiorized version using local nonascent of a strongly perturbation resilient basic algorithm

$k \leftarrow 0$
$y^k \leftarrow \bar{y}$
$\ell \leftarrow 0$
while stopping criterion not met
  $y^{k,0} \leftarrow y^k$
  for $n = 0$ to $(N_k - 1)$ **do**
    Let $v^{k,n} \in \mathcal{B}_{\eta_\ell \phi}(y^{k,n})$ (see Definition 4)
    $y^{k,n+1} \leftarrow y^{k,n} + v^{k,n}$
    $\ell \leftarrow \ell + 1$
  end for
  $y^{k+1} \leftarrow \mathcal{A}(y^{k,N_k})$
  $k \leftarrow k + 1$
end while

The behavior of this superiorized version of a basic algorithm is analyzed here according to how well it achieves feasibility and according to how well it reduces the target function values. For the feasibility question, we have the following lemma, which resembles the arguments in [24, Subection II.E] but differs in the use of local nonascending directions.

Lemma 1—*Assume that a basic algorithm represented by the algorithmic operator $\mathcal{A}$ is strongly perturbation resilient and produces an ε-compatible output for some $\varepsilon > 0$. If $\{\eta_\ell\}_{\ell=0}^{\infty}$ is a summable sequence of positive real numbers, then the superiorized version of the basic algorithm using local nonascent, given by* Algorithm 1, *produces an ε'-compatible output for each $\varepsilon' > \varepsilon$.*

Proof: We have to show that if $\mathcal{O}\left(T, \varepsilon, \{y^k\}_{k=0}^{\infty}\right)$ is defined for each $y^0 \in \mathbb{R}^L$, then for any $\varepsilon' > \varepsilon$, Algorithm 1 produces an ε'-compatible output. The strong perturbation resilience of $\mathcal{A}$ guarantees this if there exist a summable sequence $\{\beta_k\}_{k=0}^{\infty}$ of nonnegative real numbers and a bounded sequence $\{v^k\}_{k=0}^{\infty}$ of vectors in $\mathbb{R}^L$ such that

$$y^{k+1} = \mathcal{A}(y^k + \beta_k v^k), \text{ for all } k \geq 0. \quad (6)$$

Indeed, define

$$\beta_k := \max \{\|v^{k,n}\| \mid 0 \le n \le N_k - 1\} \quad (7)$$

and

$$v^k := \begin{cases} \sum_{n=0}^{N_k-1} \frac{1}{\beta_k} v^{k,n}, & \text{if } \beta_k > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (8)$$

Since $y^{k,0} = y^k$, it follows from steps 5–10 that these definitions result in $y^{k,N_k} = y^k + \beta_k v^k$. From step 7, it follows that $\{\beta_k\}_{k=0}^{\infty}$ is a subsequence of $\{\eta_\ell\}_{\ell=0}^{\infty}$ and, hence, it is a summable sequence of nonnegative real numbers. Because the sequence $\{\eta_\ell\}_{\ell=0}^{\infty}$ is summable and each $\|v^{k,n}\| \le \eta_\ell$ for appropriate ℓ, it follows that $\{v^k\}_{k=0}^{\infty}$ is bounded. Hence, the superiorized version using local nonascent, given by Algorithm 1, produces an ε'-compatible output for each $\varepsilon' > \varepsilon$.

Algorithm 1. works as follows. Initially, the iteration number k is set to 0 and y^0 is set to its initial value $\bar{y}$. The index ℓ for picking the next term of the sequence $\{\eta_\ell\}_{\ell=0}^{\infty}$ is initialized to $\ell = 0$ and is repeatedly incremented by step 9. Steps 4–13 do a full iterative step, from y^k to y^{k+1}, and repetitions of these steps generate the sequence $\{y^k\}_{k=0}^{\infty}$. During one iterative step, there is one application of the operator $\mathcal{A}$, in step 11, but there are N_k steering steps aimed at reducing the value of ϕ; the latter are done by steps 6–10. These steps produce a sequence of inner loop points $y^{k,n}$, where $0 \le n \le N_k$ with $y^{k,0} = y^k$ and $y^{k,n} \in \mathbb{R}^L$.

To our knowledge, except for [10], no proof has been published to date asserting the precise behavior of a superiorized version of an algorithm regarding the target function values. However, here we claim that Algorithm 1 systematically reduces target function values within the inner loops of perturbations, similarly to the analysis in [24, Subection II.E].

Theorem 1—*Under the conditions of Lemma 1, sequences of inner loop points $y^{k,n}$, generated by* Algorithm 1, *where $0 \le n \le N_k$ with $y^{k,0} = y^k$ and $y^{k,n} \in \mathbb{R}^L$, have the property that for all $k = 0, 1, 2, \ldots$, and all $0 \le n \le N_k$,*

$$\phi(y^{k,n}) \le \phi(y^k). \quad (9)$$

Proof: The proof is by induction. Fix an integer $k \ge 0$. For $n = 0$, we have $y^{k,0} = y^k$ and so $\phi(y^{k,0}) = \phi(y^k)$. Now assume, for any $0 \le n < N_k$, that $\phi(y^{k,n}) \le \phi(y^k)$. Next, we show that steps 6–10 lead from $y^{k,n}$ to $y^{k,n+1}$ that gives $\phi(y^{k,n+1}) \le \phi(y^k)$. The vector $v^{n,k}$ in step 7 is

chosen, by Definition 4, such that $\phi(y^{k,n} + v^{k,n}) \leq \phi(y^{k,n})$. But, in step 8, $y^{k,n+1} = y^{k,n} + v^{k,n}$ and, by the induction hypothesis, $\phi(y^{k,n}) \leq \phi(y^{k})$. Thus,

$$\phi(y^{k,n+1}) = \phi(y^{k,n} + v^{k,n}) \leq \phi(y^{k,n}) \leq \phi(y^{k}). \quad (10)$$

Therefore, we conclude that $\phi(y^{k,n}) \leq \phi(y^{k})$ for all $0 \leq n \leq N_k$.

After going through the inner loop N_k times, step 11 is executed to produce y^{k+1}. Then, increasing the value of k allows us to move to the next iterative step. Infinitely many repetitions of such steps produce the sequence of points $\{y^k\}_{k=0}^{\infty}$. Due to the repeated steering, by steps 6–10, toward reducing the value of the target function ϕ, we can expect that the output of the superiorized version using local nonascent will be superior, from the point of view of ϕ, to the output that would have been obtained, with everything else being equal by the basic algorithm. This "expected" outcome has been observed in all published experimental reports to date, see, e.g., the many papers mentioned in [4], but has not been yet mathematically proven. On this theoretical side, there is, as far as we know, only the result of [10].

3 Total variation superiorization with component-wise perturbations

3.1 The application, the approach, and the numerical demonstration

Total variation (TV) superiorization (TVS) has been used before in image reconstruction from projections with very good experimental performance, as can be seen in several of the papers posted on [4]. Since TV has everywhere a subgradient, all previous work on TVS used negative subgradients of TV as nonascent directions for the perturbations in the superiorized version of the basic feasibility-seeking algorithm.

In situations of superiorization in the SM with respect to other target functions for which there is no guarantee to have at least one nonzero partial derivative at points in the domain of the function, the notion of δ-bound nonascending perturbations, developed above, plays an important role. As mentioned before, such situations will arise when attempting to apply the SM to target functions ϕ which are not convex, or to functions defined by tabular representations.

The purpose of the numerical demonstration presented in the sequel is to show that superiorization with component-wise perturbations works at all. We do not present a full-fledged methodological numerical investigation and, therefore, the findings do not allow to draw general conclusions yet. It would be interesting to see future results when using a larger sample of datasets (e.g., randomized variations of the phantom) and get more statistical information about how superiorization with component-wise perturbations fares in comparison with gradient-based perturbations in the SM.

To explore the numerical behavior of the SM with component-wise perturbations, we wish to have something to compare it with. Therefore, we apply it to TVS without resorting to

calculations of its subgradients and compare the results with those obtained from TVS with negative subgradients as directions of nonasecent.

Our computational work surprisingly shows that even in this case in which the target function lends itself to gradient or subgradient calculations, such as TV, component-wise perturbations may be advantageous. Obviously, we do not make any general claim to this effect since more work is needed to investigate the numerical behavior of component-wise perturbations in the SM.

3.2 Image representation

Series expansion methods in image reconstruction from projections, see, e.g., [23], assume that a two-dimensional (2D) image can be represented using a linear combination of a set of fixed basis functions. Let $f: \mathbb{R}^2 \to \mathbb{R}$ be a 2D image. Then a digital approximation of f is defined at each point $r \in \mathbb{R}^2$ by

$$f(r) \approx \sum_{\ell=1}^{L} u_\ell \cdot b_\ell(r), \quad (11)$$

where b_ℓ denotes the ℓth basis function of some finite set $\{b_\ell\}_{\ell=1}^{L}$ of appropriately chosen basis functions and each component u_ℓ of the vector $u \in \mathbb{R}^L$ gives a weighting factor for the contribution of b_ℓ. For a given set of basis functions, the image estimate in (11) is uniquely determined by u, which is called the *image vector*.

Pixels form the set of basis functions used in this work. These are picture elements that cover the entire image. Each pixel has the support of a square and is defined by

$$b_\ell(r) := \begin{cases} 1, & \text{if } r \text{ is inside the } \ell\text{th pixel,} \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

When using pixel basis functions, each u_ℓ in (11) gives the average value of the image f inside the ℓth pixel. Hereafter, we denote the image approximation in (11) simply by u and use double-indexing $u_{i,j}$ for $i, j = 1, 2, \ldots, J$, to denote the value of the digital approximation in (11) at the pixel location (i, j) where the support of u is composed of $L = J^2$ pixels.

3.3 Total variation in imaging

The introduction of noise in reconstructed images is inevitable in practice. However, as introduced in [31], image restoration based on total variation has proven quite effective for a wide range of applications, including inpainting [32], super-resolution [25], image restoration [1, 11, 35], and medical imaging [27, 34, 36]. TV is formally defined as follows.

Definition 5—Let $u: \mathbb{R}^2 \to \mathbb{R}$ be a smooth image. Then the *total variation* of u is defined by

$$\mathrm{TV}(u) := \int \|\nabla u\|, \quad (13)$$

where ∇u denotes the gradient[1] of u, so that $\|\nabla u\| := \sqrt{(D_x u)^2 + (D_y u)^2}$ where D_x and D_y denote the horizontal and vertical partial derivative operators.

In the discrete case, the integral in (13) is replaced by a summation over the extent of the pixels of the digital approximation u, so that

$$\mathrm{TV}(u) = \sum_{i,j} \sqrt{\left(D_x u_{i,j}\right)^2 + \left(D_y u_{i,j}\right)^2}, \quad (14)$$

where the discrete differential operators are given by

$$D_x u_{i,j} := \begin{cases} u_{i+1,j} - u_{i,j}, & \text{if } 1 \le i < J, \\ 0, & \text{otherwise,} \end{cases} \quad (15)$$

and

$$D_y u_{i,j} := \begin{cases} u_{i,j+1} - u_{i,j}, & \text{if } 1 \le j < J, \\ 0, & \text{otherwise.} \end{cases} \quad (16)$$

3.4 TVS with component-wise perturbations

For TVS, we propose a new algorithm inspired by the framework presented in the previous sections. Our algorithm computes each nonascent vector $v^{k,n}$ for the target function $\phi = \mathrm{TV}$, in step 7 of Algorithm 1, in a specific manner applicable to TVS. Our approach proposes reducing TV by smoothing out local extrema, i.e., reducing their relative magnitude, through a type of averaging. This is accomplished using a first-order approximation of nearby points in an image u. Recall that for points $r, h \in \mathbb{R}^2$

$$u(r) \approx u(r+h) - \langle \nabla u(r+h), h \rangle \quad (17)$$

[1]This is not to be confused with the notion of gradient of a function. The meaning of ∇ will always be understood according to what it operates on.

gives a first-order approximation where $\langle \cdot, \cdot \rangle$ denotes the scalar product. Smoothing can be accomplished by using averaged first-order approximations in opposite directions from each point, i.e., by the approximation of the average given by

$$\frac{1}{2}[u(r+h)+u(r-h)] \approx u(r)+\frac{1}{2}[\langle \nabla u(r+h), h\rangle - \langle \nabla u(r-h), h\rangle]. \quad (18)$$

The corresponding perturbation of the image u, denoted by w, is defined, for each $r \in \mathbb{R}^2$, so that

$$w(r)=\frac{1}{2}(\langle \nabla u(r+h), h\rangle - \langle \nabla u(r-h), h\rangle). \quad (19)$$

By construction, this perturbation $w(r)$ yields an image $u(r) + w(r)$ with lower TV than $u(r)$ for sufficiently small h since, for each $r \in \mathbb{R}^2$, $u(r)+w(r)$ will have a value between the values of local extrema of u in proximity of r. In the actual computations, we compute step 7 of Algorithm 1 in a way that assigns a zero perturbation if the vector obtained was ascending, see (26) below. In the discrete case, when h is taken to be a unit vector along the x axis, the value $w_{i,j}$ of the perturbation w at the pixel location (i, j) is defined so that

$$w_{i,j}=\frac{1}{2}\left(D_x u_{i,j} - D_x u_{i-1,j}\right), \quad (20)$$

and when h is taken to be a unit vector along the y axis

$$w_{i,j}=\frac{1}{2}\left(D_y u_{i,j} - D_y u_{i,j-1}\right). \quad (21)$$

Note that, due to the indexing of the discrete differential operators in (15) and (16), the derivatives $D_x u_{i,j}$ and $D_y u_{i,j}$ are used above instead of $D_x u_{i+1,j}$ and $D_y u_{i,j+1}$, respectively.

In order to use w to perturb an image u within the SM, we must bound the magnitude of w. This can be done by bounding the contribution of each derivative used in the definition of w to compute a perturbation w^*, which is thereby bounded component-wise. For $\theta > 0$, define the operator $\lfloor \cdot \rfloor_\theta : \mathbb{R} \to \mathbb{R}$ by

$$\lfloor \alpha \rfloor_\theta := \min\{\theta, |\alpha|\} \cdot \operatorname{sgn}(\alpha), \quad (22)$$

where $|\cdot|$ denotes absolute value and sgn denotes the signum function. When h is a unit vector along the x axis and $\theta > 0$ is given, the value $w^*_{i,j}$ of the perturbation vector w^* at the pixel location (i, j) is defined to be

$$w^*_{i,j} := \frac{1}{2}\left(\lfloor D_x u_{i,j} \rfloor_\theta - \lfloor D_x u_{i-1,j} \rfloor_\theta\right). \quad (23)$$

Otherwise, when h gives a unit vector along the y axis, we let

$$w^*_{i,j} := \frac{1}{2}\left(\lfloor D_x u_{i,j} \rfloor_\theta - \lfloor D_x u_{i,j-1} \rfloor_\theta\right) \quad (24)$$

This formulation of w^* enables smoothing an image with control of the magnitude of the perturbation by bounding it component-wise. Each $v^{k,n}$ in step 7 of Algorithm 1 can use w^* with $\theta = \eta_\ell/\sqrt{L}$. We formalize this with the following proposition.

Proposition 1—*Let $\delta > 0$ be given. If we define $\theta := \delta/\sqrt{L}$, then the vector w^* obtained by* (23) *and/or* (24) *is such that $\|w^*\| \le \delta$.*

Proof—Let $\delta > 0$ be given. Whether h is a unit vector along either the x or y axis, the above definitions allow each $w^*_{i,j}$ to equal $w_{i,j}$ while $|w_{i,j}| \le \theta$. Otherwise, the signs of the terms composing $w^*_{i,j}$ will match those of $w_{i,j}$, but with a reduced magnitude so that the relation $|w^*_{i,j}| \le \theta$ always holds. Arranging the pixel values lexicographically to write w^* as an image vector, it then follows that

$$\|w^*\| \le \underbrace{\sqrt{\theta^2 + \cdots + \theta^2}}_{L \text{ terms}} = \theta \cdot \sqrt{L}. \quad (25)$$

Thus, by choice of θ, $\|w^*\| \le \delta$.

The new concept of the perturbation vector for TVS defined above allows us to use w^* as a δ-bound nonascending vector for ϕ at u provided that $\phi(u + w^*) \le \phi(u)$ holds. To ensure this, we compute $v^{k,n}$ in step 7 of Algorithm 1 by choosing

$$v^{k,n} := \begin{cases} w^*, & \text{if } \phi(y^{k,n} + w^*) \le \phi(y^{k,n}), \\ 0, & \text{otherwise.} \end{cases} \quad (26)$$

In (26), we compute w^* using either (23) or (24).

4 TVS with component-wise perturbations applied to image reconstruction

4.1 Image reconstruction problem

The discretized model in the series expansion approach to the image reconstruction problem of computerized tomography (CT) is described as follows. Some physical entities (e.g., x-rays) are systematically passed through an object to be scanned. Measurements are made of some physical property of these entities (e.g., attenuation). The goal of image reconstructions is to use measurements to reconstruct an image that represents the object scanned as faithfully as possible. Discretizing the object into pixels or voxels and the outer x-rays field into rays, the modeling of CT yields a matrix A called the *system matrix* and a corresponding *measurement vector* y. For a complete description, see, e.g., [23]. Each measurement y_m, which is the mth component of the vector y, can be approximated by

$$y_m \approx \sum_{\ell=1}^{L} u_\ell \cdot a_\ell^m, \quad (27)$$

where a_ℓ^m denotes entry of A in the mth row and ℓth column and each u_ℓ represents the contribution of the ℓth pixel basis function b_ℓ. One commonly used approach to solving the system $Au = y$ is to use a feasibility-seeking projection method, such as an algebraic reconstruction technique (ART) described in the next subsection.

4.2 Algebraic reconstruction techniques

The basic algorithmic operator $\mathscr{A}$ that we used to solve the image reconstruction problem is the algebraic reconstruction technique (ART) (see [23, Chapter 11]). For each row m of the system matrix, denoted by a^m, we define the operator $T_m : \mathbb{R}^L \to \mathbb{R}^L$ by

$$T_m(u) := u + \lambda \frac{y_m - \langle a^m, u \rangle}{\|a^m\|^2} a^m, \quad (28)$$

where $\lambda \in (0, 2)$ is a relaxation parameter. The basic algorithmic operator $\mathscr{A} : \mathbb{R}^L \to \mathbb{R}^L$ is then given by

$$\mathscr{A}(u) := T_M \cdots T_2 T_1(u), \quad (29)$$

where M denotes the number of rows in the system matrix. From previous works (e.g., [8]), it is known that the basic algorithmic operator $\mathscr{A}$ for ART, defined as above, is strongly perturbation resilient.

4.3 Target function reducing steps

Two methods were compared in this work: the new component-wise perturbation method for TVS (CW-TVS) and a method using negative gradients for TVS (NG-TVS) based on [8, pp. 737–738]. The target function used in this example was total variation $\phi = \mathrm{TV}$. During each perturbation step of the first method, an iterate $y^{k,n}$ was perturbed component-wise using (23) and then (24) using the perturbation size $\frac{1}{2}\eta_\ell$ for each. This process was repeated for each perturbation vector $v^{k,n}$ to reduce the value of the target function. The second method that we used was the TVS algorithm called "superiorized version of the basic algorithm" on pages 737–738 in [8]. That is, we set

$$v^{k,n} := -\eta_\ell \cdot \frac{\nabla\phi\left(y^{k,n}\right)}{\left\|\nabla\phi\left(y^{k,n}\right)\right\|} \text{ if } \mathrm{TV}\left(y^{k,n} - \eta_\ell \cdot \frac{\nabla\phi\left(y^{k,n}\right)}{\left\|\nabla\phi\left(y^{k,n}\right)\right\|}\right) \leq \mathrm{TV}\left(y^{k,n}\right), \quad (30)$$

and if this statement did not hold, then ℓ was incremented until the above statement did hold. The computation of the gradient $\nabla\phi(y)$ of the target function ϕ has up to three fraction terms. In the denominator of each fraction is a term of the form

$$\sqrt{\left(D_x(y)_{i,j}\right)^2 + \left(D_y(y)_{i,j}\right)^2} \quad (31)$$

at pixel location (i, j). To maintain numerical stability when the expression (31) becomes small, we replace the denominator term (31) with

$$\gamma_{tol} + \sqrt{\left(D_x(y)_{i,j}\right)^2 + \left(D_y(y)_{i,j}\right)^2}, \quad (32)$$

where $\gamma_{tol} := 10^{-12}$.

4.4 Computational details

The computations reported here were done with Matlab [26] on a single machine using a single CPU, a quad core Intel i5-3317U at 1.70 GHz with 4.00 GB RAM. The AIR tools package [20] was used to generate the simulated data. All reconstructions were done in the Matlab environment. Differences in reported reconstruction times are, thus, not due to different algorithms implemented in different environments.

Figure 1 shows the phantom used in our study, which is a 256 × 256 digitized version of the Shepp-Logan phantom whose calculated TV is 1461. We used this phantom as created by the AIR tools package [20]. It is represented by an image vector with 65,536 components. The values of the components in the Shepp-Logan phantom range from 0 to 1. For our displays, we use the range [0,1]. Any value below 0 is shown as black and any value above 1

is shown as white and a linear mapping is used in between. This display window was used for all images presented here.

Two sets of experiments were conducted. One had 2% Gaussian noise added to the measurement data and the other was noise-free. Projection data were collected by approximating line integrals through the digitized phantom in Fig. 1 using a fan beam, which consists of lines diverging from a single source point.The fan beam was rotated in 15 degree increments about the phantom (24 positions in total) for the noise-free data and in 9 degree increments for the noisy data (40 positions in total). Each line integral gives rise to a linear equation. The phantom itself lies in the intersection of all the solutions of the linear equations associated with these lines. The total number of linear equations generated was 12,288 for the noise-free data and 20,480 for the noisy data, thereby creating an underdetermined problem since there were $256^2 = 65,536$ unknowns. The stopping criterion used for each image reconstruction was when the proximity function

$$\text{Prox}_T(u) := \|Au - y\| \quad (33)$$

yielded a value less than or equal to $\varepsilon = 1$ for the noise-free data and $\varepsilon = 70$ for the noisy data. The initial iterate for each reconstruction was the zero vector, for which $\text{Prox}_T(0) = 3,497$ in the noise-free case. The specific choice made when running the superiorized version of the basic algorithm for our comparative study were $\eta_\ell = 0.2 \times 0.995^\ell$ and $N_k = 10$ for each k. The initial size $\eta_0 = 0.2$ appeared to give the best results for the NG-TVS method when using a kernel $a = 0.995$. The choice of relaxation parameter λ when applying ART was $\lambda = 1.0$ for the noise-free data and $\lambda = 0.2$ for the noisy data.

4.5 Computational results

The image reconstruction results are shown in Table 1 and samples are visualized in Fig. 2. Filtered back projection (FBP) images were also generated with AIR tools and are provided in Fig. 2 for reference to this traditional method using the noise-free and noisy data. Plots of TVS versus time and $\log(\|Ax^k - b\|)$ versus time are shown in Figs. 3 and 4, respectively, for the noise-free data and Figs. 5 and 6, respectively, for the noisy data. Our computational example indicates a speedup with the CW-TVS method over the NG-TVS method in the noise-free case. As shown in Table 1, for the noise-free experiment, the TV output of the component-wise approach (1500) was noticeably superior to the negative gradient approach (1833), which required over four times more computation time. As seen in Fig. 2a, the component-wise method yielded a faithful reconstruction with negligible artifacts. The NG-TVS reconstruction with noise-free data had more blurred features (specifically around the white phantom border) and several artifacts outside the phantom. Note also that although the CW-TVS method was notably faster for the noise-free data, it required more iterations of ART than the NG-TVS method (124 versus 108). In Fig. 3, we see the CW-TVS method TV function values appear to converge to the optimal TV value and at a much quicker rate than the NG-TVS method.

In the experiment with 2% Gaussian noise added to the measurements, the component-wise approach still had superior TV output (2032 versus 2941). In the noisy data case, the NG-TVS method was faster on average. For the CW-TVS reconstruction, artifacts may be seen in the corners of the image along with some blurring of edges, especially for the three small ellipses inside the phantom. The NG-TVS method displays more notable artifacts outside the phantom, faintly reminiscent of the streaks in the filtered back projection reconstructions. There are also blurring artifacts in the NG-TVS reconstruction. Lastly, in Fig. 5, we see the TV values increase over time.

Remark 1—The SM parameters such as N_k and the number a with which the parameters η_ℓ were generated, as well as the parameters associated with the feasibility-seeking ART were chosen as well as we could based on earlier published experiences and on some preliminary runs that we did with various values. The main point to observe in this regard is that they were identical in the runs with component-wise perturbations and the runs with negative subgradients as directions of nonascent. Therefore, it is reasonable to assume that, since they were equal, they did not affect the comparative outcomes of the runs. Future methodological numerical investigations should address the choice of parameters systematically.

5 Discussion

Relative computational costs

Analysis of the difference in the computational cost of each TVS method is as follows. The negative gradient approach given in Section 4.3 requires the computation of up to three fractions for every component of the perturbation vector. In the denominator of each of these fractions is a square root term along with two multiplications and multiple additions/subtractions. On the other hand, the component-wise perturbation method given in Section 3.4 requires only additions/subtractions, direct comparisons of floating point numbers with a minimum function, multiplication by the signum function, and division by 2. Hence, we may expect the component-wise TVS method to be less computationally expensive per iteration. This is consistent with the results in Table 1.

Differences in output TV values

Inference of the difference in the output TV values from each method is as follows. With the CW-TVS method, the value of the perturbation to each pixel is bounded individually. This causes each component in a perturbation vector to be relatively equally weighted, thereby enabling a Gaussian distribution of entries. On the other hand, with NG-TVS, the partial derivative is computed with respect to each pixel and then the vector is scaled as a whole. And, the partial derivative of TV at pixels along any sharp edge in the image has far greater magnitude than at the majority of pixels. This is the case with the piecewise-constant Shepp-Logan phantom. This implies that the distribution of values in the perturbation vector for NG-TVS in our reconstructions should consist mostly of near-zero values and a few large nonzero values. Indeed, this is consistent with our observations. Also, due to these few pixels having large perturbations, it was observed in our reconstructions that the parameter ℓ often incremented several times between steps before perturbations had sufficiently small magnitude to reduce the TV. After several loops through the superiorization algorithm, this

caused the η_l-bound to be so small that the perturbations to reduce TV became negligible. Thus, we assume that the chief advantage of the new TVS method with respect to TV value output is due to the component-wise bounding of perturbations.

Connection to previous formulations of the SM

Previous works (e.g., [24]) made note that convexity of the target function does not need to be assumed for superiorization. However, when the target function ϕ is not convex, there may exist a nonascent point η_d for which d is not a nonascending vector. Additionally, if such a nonascent point is found, we do not need to be concerned whether there exists $\delta > 0$ such that, for all $\lambda \in [0, \delta]$, the quantity λd gives a nonascent valued point. Hence, the notion of a nonascending δ-ball may be understood to be less restrictive and allow for a wider class of target function value reducing steps.

6 Conclusions

The superiorization methodology (SM) allows the conversion of a feasibility-seeking algorithm into a superiorized version of the feasibility-seeking algorithm that, in addition to finding an ε-compatible solution of the constraints, steers iterates toward a reduced target function value. The superiorized version of the basic algorithm accomplishes this by interlacing target function nonascent steps into the original algorithm in an automatic fashion. This work has extended the scope of the SM by introducing the notion of nonascending δ-balls for the nonascent steps. Using this notion, perturbation steps of the superiorized version of a basic algorithm can now be chosen from a wider class of target function value reducing steps, namely, functions that do not have any partial derivatives or whose partial derivatives cannot be calculated. Future investigations may also apply this formulation of the SM to problems where the target functions may not be convex (e.g., as often occurs in the field of intensity-modulated radiation therapy (IMRT) treatment planning) and to functions that are given only by tabular presentations.

We have presented an example that shows that our CW-TVS (component-wise total variation superiorization) method works well. As a byproduct, we discovered that it finds a better solution than an NG-TVS (negative-gradient TVS) approach, and in less computation time in the noise-free case. Due to the limited scope of our numerical work, we do not make any general claims. However, this finding is understandable in view of the simplicity of the component-wise perturbations with their averaging nature and the fact that the components of these perturbations are weighted relatively equally, which allows for a larger portion of an image to be smoothed with each perturbation than with the negative gradient approach. While the negative gradient approach directly attempts to reduce the target function of total variation, it is limited in its ability to remove artifacts from the image. We demonstrated this experimentally on a large-sized image reconstruction application that was modeled and set up as a constrained superiorization problem.

Acknowledgments

This project was supported by Research Grant No. 2013003 of the United States-Israel Binational Science Foundation (BSF) and by Award No. 1P20183640-01A1 of the National Cancer Institute (NCI) of the National Institutes of Health (NIH).

We greatly appreciate the constructive comments of two anonymous reviewers which helped us improve the paper.

References

1. 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; [PubMed: 19635705]
2. Bian J, Siewerdsen J, Han X, Sidky E, Prince J, Pelizzari C, Pan X. Evaluation of sparse-view reconstruction from flat-panel-detector cone-beam CT. Phys Med Biol. 55:6575–6599.2010; [PubMed: 20962368]
3. Butnariu D, Davidi R, Herman GT, Kazantsev IG. Stable convergence behavior under summable perturbations of a class of projection methods for convex feasibility and optimization problems. IEEE J Sel Top Sign Proces. 1:540–547.2007;
4. Censor, Y. Superiorization and perturbation resilience of algorithms: a bibliography compiled and continuously updated. http://math.haifa.ac.il/yair/bib-superiorization-censor.html see also: arXiv: 1506.04219
5. Censor Y. Weak and strong superiorization: Between feasibility-seeking and minimization. An Stiint ale Univ Ovidius Constanta-Ser Mat. 23:41–54.2015;
6. Censor Y. Can linear superiorization be useful for linear optimization problems? Inverse Prob. 33:044006.2017;
7. Censor Y, Davidi R, Herman GT. Perturbation resilience and superiorization of iterative algorithms. Inverse Prob. 26:065008.2010;
8. Censor Y, Davidi R, Herman GT, Schulte RW, Tetruashvili L. Projected subgradient minimization versus superiorization. J Optim Theory Appl. 160:730–747.2014;
9. Censor Y, Herman GT, Jiang M. Superiorization: theory and applications. Inverse Problems. 33(4)2017;
10. Censor Y, Zaslavski Á. Strict Fejer monotonicity by superiorization of feasibility-seeking projection methods. J Optim Theory Appl. 165:172–187.2015;
11. Chan, T, Esedoglu, S, Park, F, Yip, A. Handbook of Mathematical Models in Computer Vision. Springer Science+Business Media, Inc; 2006. Total variation image restoration: overview and recent developments; 17–31.
12. Combettes P, Luo J. An adaptive level set method for nondifferentiable constrained image recovery. IEEE Trans Image Process. 11:1295–1304.2002; [PubMed: 18249699]
13. Combettes P, Pesquet JC. Image restoration subject to a total variation constraint. IEEE Trans Image Process. 13:1213–1222.2004; [PubMed: 15449583]
14. Davidi R, Herman GT, Censor Y. Perturbation-resilient block-iterative projection methods with application to image reconstruction from projections. Int Trans Oper Res. 16:505–524.2009; [PubMed: 23271857]
15. Defrise M, Vanhove C, Liu X. An algorithm for total variation regularization in high-dimensional linear problems. Inverse Prob. 27:065002.2011;
16. Garduño E, Herman GT. Superiorization of the ML-EM algorithm. IEEE Trans Nucl Sci. 61:162–172.2014;
17. Garduño E, Herman GT. Computerized tomography with total variation and with shearlets. Inverse Prob. 33:044011.2017;
18. Garduño E, Herman GT, Davidi R. Reconstruction from a few projections by ℓ_1-minimization of the Haar transform. Inverse Prob. 27:055006.2011;
19. Gibali A, Petra S. DC-programming versus ℓ_0-superiorization for discrete tomography. Analele Stiintifice ale Universitatii Ovidius Constanta-Seria Matematica. 2017
20. Hansen PC, Saxild-Hansen M. AIR Tools–A MATLAB package of algebraic iterative reconstruction methods. J Comput Appl Math. 236:2167–2178.2012;
21. Helou Neto E, De Pierro Á. Incremental subgradients for constrained convex optimization: a unified framework and new methods. SIAM J Optim. 20:1547–1572.2009;
22. Helou Neto E, De Pierro Á. On perturbed steepest descent methods with inexact line search for bilevel convex optimization. Optimization. 60:991–1008.2011;

23. Herman, GT. Fundamentals of Computerized Tomography. 2. Springer-Verlag; London: 2009.
24. Herman GT, Garduño E, Davidi R, Censor Y. Superiorization: an optimization heuristic for medical physics. Med Phys. 39:5532–5546.2012; [PubMed: 22957620]
25. Marquina A, Osher S. Image super-resolution by TV-regularization and Bregman iteration. J Sci Comput. 37:367–382.2008;
26. MATLAB: A high-level language and interactive environment system by Mathworks. http://www.mathworks.com/products/matlab
27. Needell D, Ward R. Stable image reconstruction using total variation minimization. SIAM J Imag Sci. 6:1035–1058.2013;
28. Nikazad T, Davidi R, Herman GT. Accelerated perturbation-resilient block-iterative projection methods with application to image reconstruction. Inverse Prob. 28:035005.2012;
29. Nurminski E. Envelope stepsize control for iterative algorithms based on Fejer processes with attractants. Optim Methods Soft. 25:97–108.2010;
30. Rios L, Sahinidis N. Derivative-free optimization: a review of algorithms and comparison of software implementations. J Glob Optim. 56:1247–1293.2013;
31. Rudin L, Osher S, Fatemi E. Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenomena. 60:259–268.1992;
32. Shen J, Chan T. Mathematical models for local nontexture inpaintings. SIAM J Appl Math. 62:1019–1043.2002;
33. Sidky E, Duchin Y, Pan X, Ullberg C. A constrained, total-variation minimization algorithm for low-intensity x-ray CT. Med Phys. 38:S117–S125.2011; [PubMed: 21978112]
34. Sidky E, Pan X. Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. Phys Med Biol. 53:4777–4807.2008; [PubMed: 18701771]
35. Wang Y, Yang J, Yin W, Zhang Y. A new alternating minimization algorithm for total variation image reconstruction. SIAM J Imag Sci. 1:248–272.2008;
36. Zhang HM, Wang LY, Yan B, Li L, Xi XQ, Lu LZ. Image reconstruction based on total-variation minimization and alternating direction method in linear scan computed tomography. Chin Phys B. 22:078701.2013;

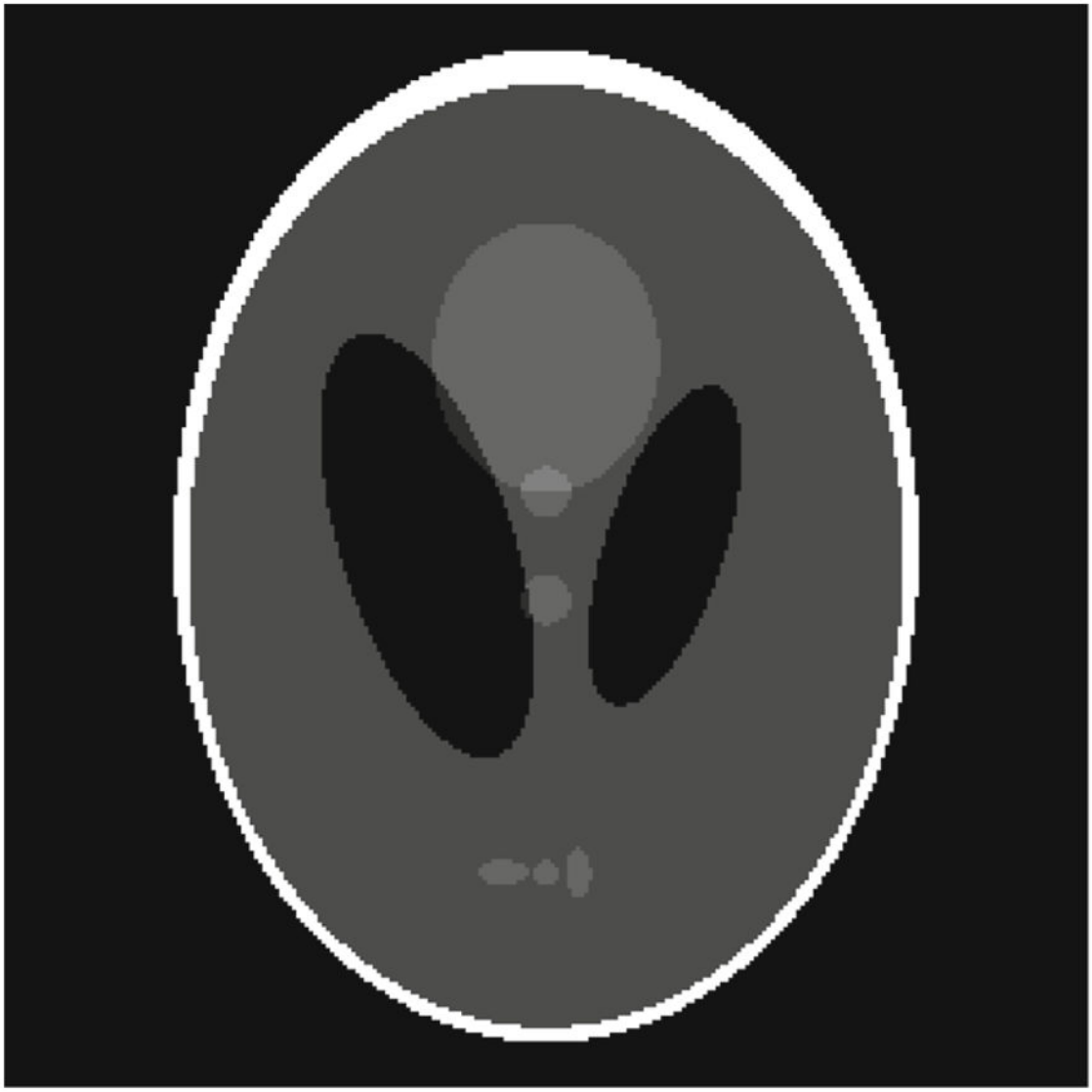

Fig. 1.
Original 256 × 256 pixel Shepp-Logan phantom with TV = 1461

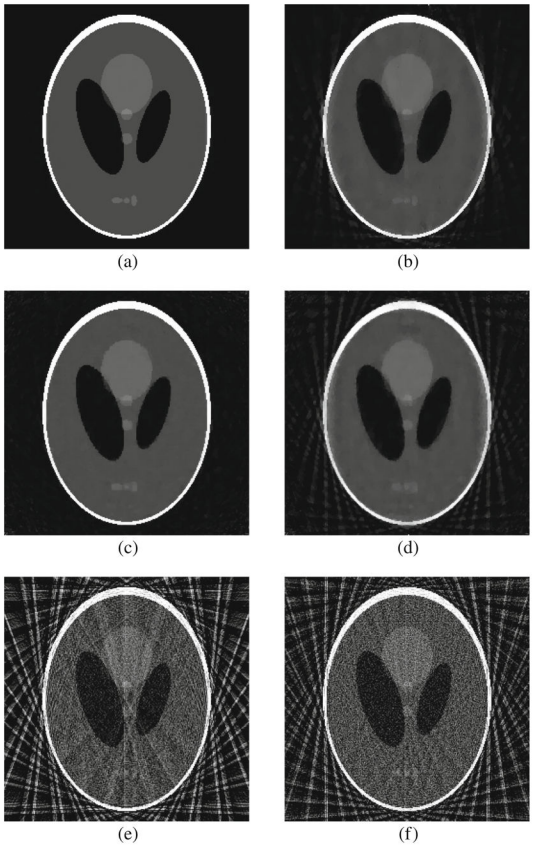

Fig. 2.

A noise-free reconstruction with component-wise TVS method in **a** and the negative gradient TVS method in **b**. A reconstruction with 2% Gaussian noise with component-wise TVS method in **c** and the negative gradient TVS method in **d**. FBP reconstructions are provided for noise-free and noisy reconstructions in **e** and **f**, respectively

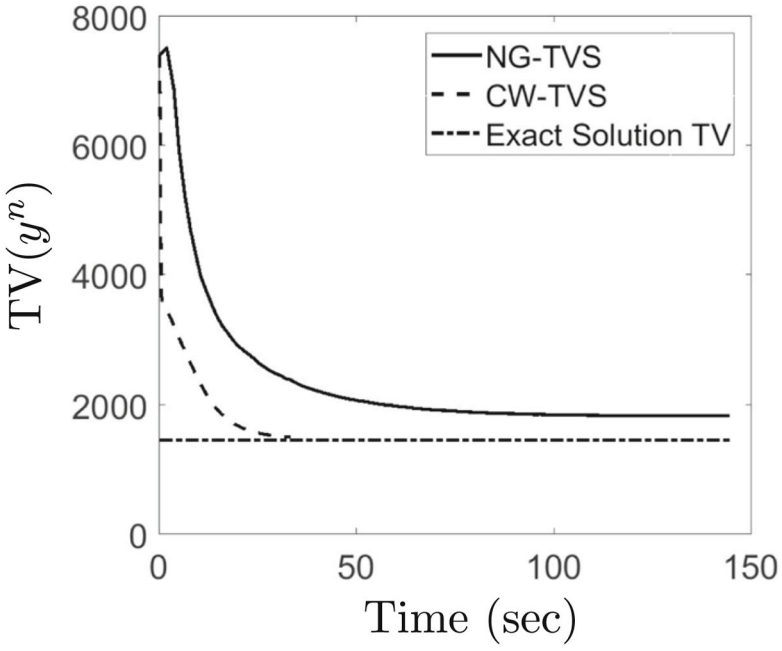

Fig. 3.
Plots of TV from a noise-free data trial used to create the images in Fig. 2a and b

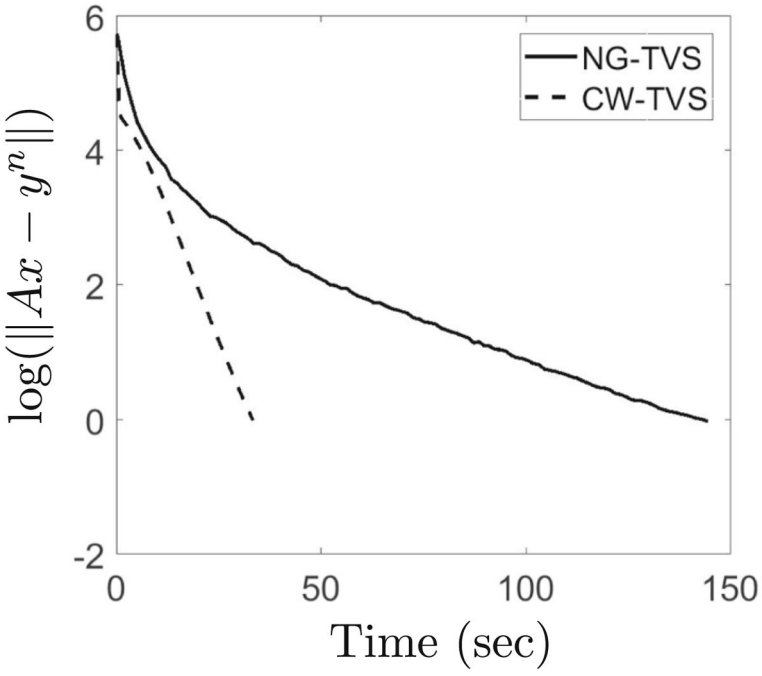

Fig. 4.
Plots of proximity from a noise-free data trial used to create the images in Fig. 2a and b

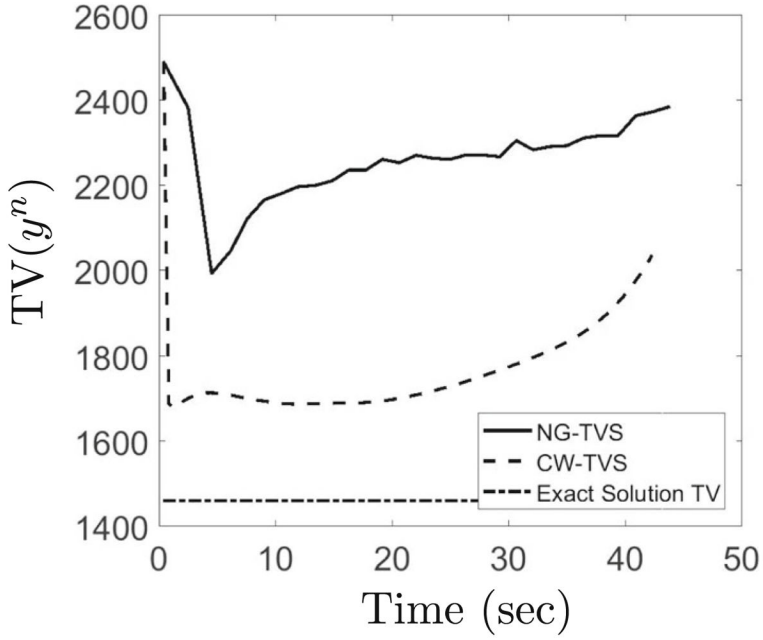

Fig. 5.
Plots of TV from a noisy data trial used to create the images in Fig. 2c and d

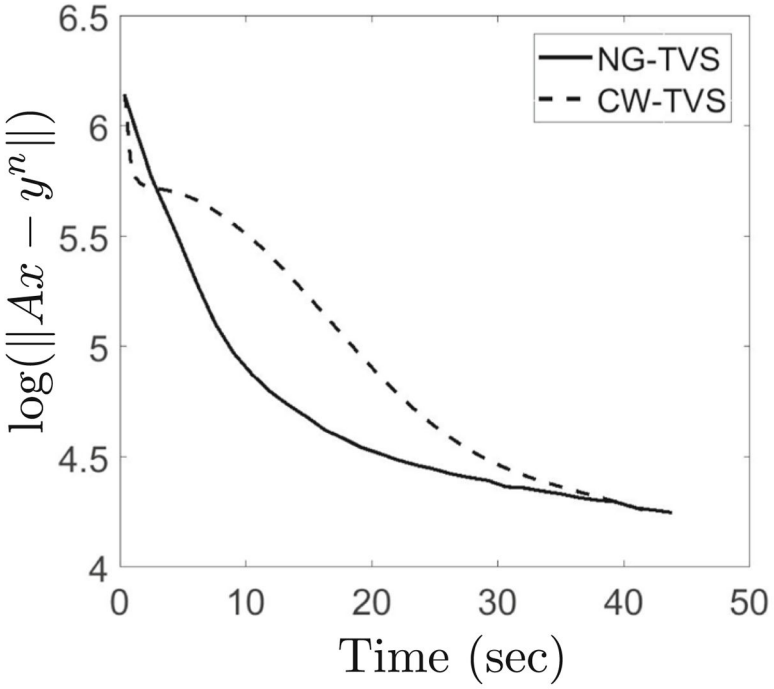

Fig. 6.
Plots of proximity from a noisy data trial used to create the images in Fig. 2c and d

Table 1

Simulated image reconstruction results

Method	2% Gaussian noise			Noise-free		
	TV	Time (s)	Iterations	TV	Time (s)	Iterations
CW-TVS	2032 ± 11	40.0 ± 0.1	106.9 ± 0.6	1500 ± 0	33.1 ± 0.5	124 ± 0
NG-TVS	2941 ± 897	38.3 ± 10.3	25.0 ± 7.5	1833 ± 0	143.5 ± 1.2	108 ± 0

The stopping criterion differed between the noise-free and noisy data reconstructions and so results should not be compared between the two cases. Displayed values are averages of 30 trials and range is one standard deviation