DE GRUYTER
OPEN

Communications in Applied
and Industrial Mathematics
ISSN 2038-0909

Research article

Commun. Appl. Ind. Math. 9 (1), 2018, 68–86 DOI: 10.1515/caim-2018-0005

Comparison of minimization methods for nonsmooth image segmentation

L. Antonelli[1]*, V. De Simone[2]

[1]Institute for High-Performance Computing and Networking (ICAR), CNR, via P. Castellino 111, 80131 Naples, Italy

[2]Department of Mathematics and Physics, Università della Campania "Luigi Vanvitelli", viale A. Lincoln 5, 81100 Caserta, Italy

*Email address for correspondence: laura.antonelli@cnr.it

Communicated by Roberto Natalini

Received on 08 01, 2017. Accepted on 02 02, 2018.

Abstract

Segmentation is a typical task in image processing having as main goal the partitioning of the image into multiple segments in order to simplify its interpretation and analysis. One of the more popular segmentation model, formulated by Chan-Vese, is the piecewise constant Mumford-Shah model restricted to the case of two-phase segmentation. We consider a convex relaxation formulation of the segmentation model, that can be regarded as a nonsmooth optimization problem, because the presence of the l1-term. Two basic approaches in optimization can be distinguished to deal with its non differentiability: the *smoothing methods* and the *nonsmoothing methods*. In this work, a numerical comparison of some first order methods belongs of both approaches are presented. The relationships among the different methods are shown, and accuracy and efficiency tests are also performed on several images.

Keywords: `nonsmooth optimization, first order methods, image segmentation`

AMS subject classification: `65K10 - 68U10 - 90C30`

1. Introduction

Many problems in image processing can be formulated as a nonsmooth optimization problem, where the properties of the objective function and/or the constraints reflects some specific requests on the solution. The cause of nonsmoothness can be of various nature: intrinsic to the phenomenon, methodological, i.e. the nonsmoothness is introduced by methodologies for solving the problems (decomposition, dual formulation, exact penalty functions, ...), or numerical, i.e. the so called stiff problems are analytically smooth but numerically nonsmooth. Since classical techniques, usually gradient-based methods, require certain differentiability and strong regularity assumptions upon the functions to be optimized, they can not be directly applied. Therefore a main challenge today is to develop "ad hoc" techniques taking into account the specific problem in order to compensate for the absence of differentiability. It is possible to identify two basic approaches to solution: *smoothing methods*, which formulate the problem as a suitable smooth one and solve the problem with smooth convex solvers (splitting, Huber or pseudo-Huber regularization, ...), and *nonsmoothing methods*, which usually do not know the whole subdifferential of the function but only one arbitrary subgradient at each point (subgradient, proximal gradient, bundle, ...).

In this work, we compare different first order methods in both approaches for solving a nonsmooth convex problem describing the process of image segmentation. In this problem, as in many other imaging tasks, the non-differentiability of the objective function is related to the presence of l1-penalty term. The paper is organized as follows. The Section 2 gives an introduction to segmentation problem and its description by means of a popular convex nonsmooth model. The Section 3 presents some popular first order methods for solving the selected segmentation problem. Section 4 is devoted to numerical tests and comparisons.

2. A nonsmooth model for image segmentation

Segmentation is a typical task in image processing having as main goal the partitioning of the image into multiple segments (sets or groups of pixels) in order to simplify its interpretation. Since the partitioning is realized according to some features, such as intensity, texture or colour, the output of the process represents the image like a map: each segment represent an object of the image, with different intensity, texture or colour. In this context the two-phase segmentation is considered, were the image is partitioned only in two regions: background and object regions. One of the more popular segmentation model is the non-convex Chan-Vese (CV) one [1], that can be view as the Mumford-Shah model [2] restricted to the case of piecewise constant two-phase segmentation; its level-set formulation reads as follows:

$$\min_{c_1,c_2,\phi} E_{CV}(c_1,c_2,\phi)\,, \tag{1}$$

where

$$\begin{aligned} E_{CV}(c_1,c_2,\phi) = {} & \textstyle\int_\Omega |\nabla H(\phi(x))|\,dx \\ & + \lambda\left(\textstyle\int_\Omega H(\phi(x))\,(c_1-\bar{u}(x))^2\,dx\right. \\ & \left. + \textstyle\int_\Omega (1-H(\phi(x)))\,(c_2-\bar{u}(x))^2\,dx\right)\,. \end{aligned} \tag{2}$$

Here $\Omega \subset \mathbb{R}^2$ is an open bounded set with Lipschitz boundary (generally a rectangle), $\bar{u}(x) : \Omega \to \mathbb{R}$ represents the image to be segmented, H is the Heaviside function, $\phi : \Omega \to \mathbb{R}$ is a Lipschitz-continuous function whose zero level set represents the boundary $\partial\Sigma$ of a set $\Sigma \subset \Omega$, $c_1, c_2 \in \mathbb{R}$, and $\lambda > 0$ is a suitable regularization parameter. Solving (1) means finding the best approximation to $\bar{u}(x)$ among all the functions that take only two values; c_1 and c_2 represent these values, while Σ and $\Omega \backslash \Sigma$ are the sets where they are taken, which provide a two-phase partition of Ω. The first term in the right-hand side of (2) is a regularization term, which penalizes the size of $\partial\Sigma$. For any fixed ϕ, the values of c_1 and c_2 that minimize E_{CV} are given by

$$c_1 = \frac{\int_\Omega \bar{u}(x) H(\phi(x)) dx}{\int_\Omega H(\phi(x)) dx}\,, \; c_2 = \frac{\int_\Omega \bar{u}(x)(1 - H(\phi(x))) dx}{\int_\Omega (1 - H(\phi(x))) dx}\,, \tag{3}$$

i.e., by the mean values of $\bar{u}(x)$ in the regions Σ and $\Omega \backslash \Sigma$. Therefore, a natural approach to solve problem (1) is to alternate between the computation of c_1 and c_2 through (3) and the minimization of $E_{CV}(c_1, c_2, \phi)$ with respect to ϕ.

The solution of problem (1) involves several difficulties than drastically restrict the methods that can be used. Indeed, the classical techniques are inappropriate because these may get stuck into local minima, thus providing poor segmentations.

In this work we consider a convex relaxation approach based on the idea of removing the constraint that $u(x)$ is approximated by functions taking only two values. Specifically, Chan-Esedoglu-Nikolova proved in [3] that for any given (c_1, c_2), a global minimizer for the model (1) can be obtained by solving the following convex problem, that we named CEN model:

$$\begin{aligned} &\min\nolimits_u \, E_{CEN}(c_1, c_2, u)\,, \\ &\text{ s.t. } \;\; 0 \le u \le 1\,, \end{aligned} \tag{4}$$

where

$$\begin{aligned} E_{CEN}(c_1, c_2, u) = &\int_\Omega |\nabla u| dx \\ &+\lambda \left(\int_\Omega u(x) \left(c_1 - \bar{u}(x)\right)^2 dx \right. \\ &\left. + \int_\Omega (1 - u(x)) \left(c_2 - \bar{u}(x)\right)^2 dx \right)\,, \end{aligned} \tag{5}$$

and by taking

$$\Sigma = \{x \in \Omega : u(x) \ge \mu\}\,, \tag{6}$$

for almost any $\mu \in (0, 1)$.

3. Numerical methods for CEN model

We focus on a discretize-than-optimize approach, i.e. we first introduce a dicrete formulation of the CEN model, then we solve the resulting optimization problem. To this aim, let Γ be a discretization of the image domain Ω into $m \times n$ pixels

$$\Gamma = \{(i,j) : 0 \le i \le m-1, 0 \le j \le n-1\},$$

we denote by $u_{i,j}$ the value of intensity function in the centre of pixel (i,j). We shall use the following norm and inner product notations:

$$|u|_1 = \sum_{i,j} |u|_{i,j} \,, \; \|u\| = \left(\sum_{i,j} u_{i,j}^2\right)^{\frac{1}{2}} \,, \; < u, v >= \sum_{i,j} u_{i,j} \cdot v_{i,j} \,.$$

To discretize the total variation term in (5) we set

$$|\nabla u|_{i,j} = \sqrt{\left(\delta_+^x u_{i,j}\right)^2 + \left(\delta_+^y u_{i,j}\right)^2},$$

where δ_+ is the forward finite-difference operator with the values of u related to the indexes outside Γ defined by replication. Then, the discrete formulation of problem of (5) has the following form:

$$\begin{aligned} &\min\nolimits_u \; E(c_1, c_2, u) = |\nabla u|_1 + \lambda < u, r > , \\ &\text{s.t.} \;\; 0 \le u \le 1 \,, \; u_{min} \le c_1, c_2 \le u_{max} \,, \end{aligned} \tag{7}$$

with

$$r_{i,j} = (c_1 - \bar{u}_{i,j})^2 - (c_2 - \bar{u}_{i,j})^2 \,. \tag{8}$$

The discrete version of (3) is:

$$c_1 = \frac{\sum_{i,j} \bar{u}_{i,j} u_{i,j}}{\sum_{i,j} u_{i,j}} \,, \; c_2 = \frac{\sum_{i,j} \bar{u}_{i,j}(1 - u_{i,j})}{\sum_{i,j}(1 - u_{i,j})} \,. \tag{9}$$

Different methods have been proposed to solve the problem (7) [3–9], among which we compare, in this work, some first order optimization methods. The first one is included in the class of smoothing methods, where ε-regularized version of $|\nabla u|$ is used to deal with its non-differentiability; the other methods, that we considered, rely on the use of proximal operators[a] [10], and

[a] The proximal operator associated with a function h is defined by:

$$prox_t(h)(z) := \underset{u}{argmin} \left\{ h(u) + \frac{1}{2t} ||u - z||^2 \right\}, \quad t > 0.$$

can also be directly applied to non-differentiable functionals

$$F(x) := f(Kx) + g(x)\,, \tag{10}$$

where $f : Y \to \mathbb{R}$ and $g : X \to \mathbb{R}$ are closed proper convex functions and $K : X \to Y$ is a linear operator.

3.1. Smoothing approach

SPG-based Alternating Segmentation (SPGASeg) algorithm [9] is based on the application of nonmonotone spectral projected gradient (SPG) method [11], within an alternating minimization procedure, to (7). The interest for SPG methods is due to their faster convergence with respect to classical gradient projection methods, which results from the combination of the spectral properties of the Barzilai and Borwein (BB) steplength [12,13] with the nonmonotone line-search technique by Grippo, Lampariello and Lucidi (GLL) [14]. Since $|\nabla u|_{i,j}$ is nondifferentiable in zero, we used a classical regularization as follows:

$$|\nabla u|_{i,j} = \sqrt{(\delta_+^x u_{i,j})^2 + (\delta_+^y u_{i,j})^2 + \epsilon}\,,$$

where ϵ is a small positive parameter. We denote with G the vector ∇E, with G_c the vector with components $\partial E/\partial c_1$ and $\partial E/\partial c_2$, G_u the vector with components $\partial E/\partial u_{i,j}$. The k-th iteration of the minimization procedure apply an SPG step to $E(c_1^k, c_2^k, u)$ according to

$$u^{k+1} = u^k + \theta^k d_u^k\,, \tag{11}$$

where the search direction d_u^k is defined as

$$d_u^k = P_U\left(u^k - \alpha^k G_u^k\right) - u^k\,. \tag{12}$$

In (12) P_U denotes the orthogonal projection on $U = [0,1]^{mn}$ and α^k is the BB step length, more precisely given a "small" parameter $\alpha_{min} > 0$ and a "large" one $\alpha_{max} > 0$, α^k is defined as:

$$\alpha^k = \begin{cases} \min\left\{\alpha_{max}, \max\left\{\alpha_{min}, \alpha_{BB}^k\right\}\right\}, & \text{if } \alpha_{BB}^k > 0 \\ \alpha_{max}\,, & \text{otherwise} \end{cases} \tag{13}$$

where $\alpha_{BB}^k = \frac{\langle s^{k-1}, s^{k-1}\rangle}{\langle s^{k-1}, y^{k-1}\rangle}$, $s^k = u^{k+1} - u^k$ and $y^k = G_u^{k+1} - G_u^k$.
The steplength θ^k in (11) is such that GLL condition holds, i.e.:

$$E\left(c_1^k, c_2^k, u^k + \theta^k d^k\right) \le \max_{0 \le j \le \min\{k, \nu-1\}} E\left(c_1^{k-j}, c_2^{k-j}, u^{k-j}\right) + \gamma\, \theta_k \left\langle G_u^k, d_u^k \right\rangle, \tag{14}$$

with $\nu \in \mathbb{N}$ and $\gamma \in (0,1)$. A steplength θ^k satisfying (11) is computed by a line search procedure, starting from $\theta_0^k = 1$ as described in [9].
In the second step c_1^{k+1} and c_2^{k+1} are computed by exact minimization of E using u^{k+1} in (9). *SPGASeg* algorithm is summarized in Algorithm 3.1.

Algorithm 3.1 *SPGASeg*

```
given tol, $\alpha_{max} > \alpha_{min} > 0,\ \alpha^0 \in [\alpha_{min}, \alpha_{max}],\ \nu \in \mathbb{N},\ \gamma \in (0,1)$
choose $u^0$ and compute $c_1^0$ and $c_2^0$ as in (9), using $u^0$
set $k = 0$
while stopping criteria does not satisfy tol do
    compute $d_u^k = P_U\left(u^k - \alpha^k G_u^k\right) - u^k$
    compute $\theta^k$ that satisfies (14) by a line search along $u^k + \theta d_u^k$
    set $u^{k+1} = u^k + \theta^k d_u^k$
    compute $c_1^{k+1}$ and $c_2^{k+1}$ as in (9), using $u^{k+1}$
    compute $\alpha_{k+1}$ as in (13)
    set $k = k + 1$
end while
```

3.2. Nonsmoothing approach

Split Bregman (SB) [15] is one of the nonsmoothing method for solving problem (7) without any regularization technique. Introducing the auxiliary variable, $d = \nabla u$, the segmentation problem (7) can be formulated as

$$\begin{aligned} &\min\nolimits_{0\le u\le 1,d} E(u,d)\,, \\ &\text{s.t.} \qquad d = \nabla u\,, \end{aligned} \tag{15}$$

where $E(u,d) = |d|_1 + \lambda < u, r >$.
One way to solve (15) is to convert it into an unconstrained problem, for example by using a penalty function/continuation method, which approximates (15) by a sequence

$$\min\nolimits_{0\le u\le 1,d} E(u,d) + \tfrac{\eta_k}{2}\|d - \nabla u\|^2, \quad \eta_k \in \mathbb{R}^+, \tag{16}$$

where large values of η_k make (16) extremely difficult to solve numerically. Alternatively, SB reduces (15) in a short sequence of unconstrained problems by using the *Bregman distance* [16] associated with (15) [15], where conversely the value of η_k remains constant. The application of SB method to problem (7) produces the framework described in Algorithm 3.2, that we named *SBSeg*.

Algorithm 3.2 *SBSeg*

```
given tol,   $\lambda > 0$,   $\eta > 0$,
choose $u^0$ and compute $c_1^0$ and $c_2^0$ as in (9), using $u^0$
set $k = 0$
while stopping criteria does not satisfy tol do
    compute $(u^{k+1}, d^{k+1}) = argmin_{0\leq u\leq 1,d} E(u,d) + \frac{\eta}{2}\|d - \nabla u - b^k\|^2$
    set $b^{k+1} = b^k + \nabla u^{k+1} - d^{k+1}$
    compute $c_1^{k+1}$ and $c_2^{k+1}$ as in (9), using $u^{k+1}$
    set $k = k + 1$
end while
```

At k-th iteration of the Algorithm 3.2, the minimization of the sub-problem is not trivial, because of the presence of a nonsmooth term. Anyway, there is an extensive literature in optimization about methods for solving convex programming problems with an objective function of the form (10), among which the splitting and proximal gradient methods.
The algorithm presented by Goldstein, Bresson and Osher in [15], that we named *SBSeg-GBO*, proposes a *split* of the l1 and l2 terms in (15) and then an alternating minimization scheme is applied, with respect to u and d, performing

$$(17) \qquad \begin{cases} \text{Step } 1: u^{k+1} = \min_{0\leq u\leq 1} \lambda < u, r^k > + \frac{\eta}{2}\|d^k - \nabla u - b^k\|^2 \\ \text{Step } 2: d^{k+1} = \min_d |d|_1 + \frac{\eta}{2}\|d - \nabla u^{k+1} - b^k\|^2 \quad . \end{cases}$$

About the minimization with respect to u, the authors observe that the objective function is quadratic in $u_{i,j}$, then the minimum is found by solving:

$$(18) \qquad (\Delta u)_{i,j} = \frac{\lambda}{\eta} r_{i,j} + (div(d-b))_{i,j} \;\; i = 1, ..., m, \;\; j = 1, ..., n$$

where the discrete Laplace and divergence operators are setting as

$$(\Delta u)_{i,j} = \frac{1}{4}(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}),$$

and

$$(div(u))_{i,j} = \delta_-^x p_{i,j} + \delta_-^y p_{i,j}$$

with δ_- is the backward finite-difference operator. The system of equations (18) can be solved by either Gauss Seidel (GS) method or Fourier transform based methods. The minimization with respect to d is carried out by using the proximal operator of l1-norm, that is the shrink operator defined as:

$$shrink(x,\gamma) = \frac{x}{|x|} \cdot max(|x| - \gamma, 0)\,. \tag{19}$$

In [17] is showed the connection between the SBSeg-GBO algorithm and the Douglas-Rachford splitting (DRS) one solving the dual formulation of problem (16). DRS algorithm is equivalent to *Primal-Dual (PD)* algorithms [6], that usually minimize functionals (10), when $K = I$.

Using duality, PD rewrites the minimization of problem (10) when $K = I$, as the saddle point one and than applies the alternating proximal minimization algorithm for dual and primal steps generating this scheme:

$$\begin{cases} x^{k+1} = prox_\tau(g)(x^k - \tau y^k) \\ y^{k+1} = prox_\sigma(f^*)(y^k + \sigma(2x^{k+1} - x^k)) \end{cases} \quad , \tag{20}$$

with $\sigma\,, \tau \leq 1$. Choosing $\sigma = 1/\tau$ and $\tilde{x}^k = x^k - \tau y^k$, and using the Moreau's identity the iteration in (20) can be equivalently written as the following DRS iterations:

$$\begin{cases} x^{k+1} = prox_\tau(g)(\tilde{x}^k) \\ \tilde{x}^{k+1} = \tilde{x}^k - x^{k+1} + prox_\tau(f)(2x^{k+1} - \tilde{x}^k) \end{cases} \quad , \tag{21}$$

Setting

$$f(u,d) = \lambda < u, r > + |d|_1 + \chi_U(u)\,, \qquad g(u,d) = \chi_C\,, \tag{22}$$

where χ is the indicator function and $C = \{(u,d) \,:\, d = \nabla u\}$, the application of DRS method to segmentation problem (7) can be summarized in Algorithm 3.3.

Note that, the proximal operator of g is the orthogonal projection on the convex set C; it can be computed by solving a linear system of equations:

$$\tilde{y} = \nabla\tilde{x} = \nabla\left((I - \Delta)^{-1}(\tilde{x} - div(\tilde{y}))\right)\,. \tag{23}$$

Furthermore, f in (22) is a separable sum of two functions, one depends only on u and the other only on d; thus the proximal operator can be decomposed as two simpler operators which can be applied to u and d respectively, giving a projection in the set U for u and a shrink operator for d, respectively.

Finally, we consider a further algorithm based on SB, that we named *SBSeg-FPG*, where the Fast Proximal Gradient method (FPG) [18,19] is applied to solve the minimization of the sub-problem at each outer iteration of the

Algorithm 3.3 *DRSSeg*

```
$\lambda > 0$, tol $\tau > 0$,
choose $u^0$ and compute $c_1^0$ and $c_2^0$ as in (9), using $u^0$
set $d^0 = \nabla u^0$
set $k = 0$
while stopping criteria does not satisfy tol do
    set $i = 0$, $x^0 = u^k$, $y^0 = d^k$
    while inner stopping criteria does not satisfy do
        compute $(x^{i+1}, y^{i+1}) = prox_\tau(g)(\tilde{x}^i, \tilde{y}^i)$
        compute $(\tilde{x}^{i+1}, \tilde{y}^{i+1}) = (\tilde{x}^i, \tilde{y}^i) - (x^{i+1}, y^{i+1}) + prox_\tau(f)(2(x^{i+1}, y^{i+1}) - (\tilde{x}^i, \tilde{y}^i))$
        set $i = i + 1$
    end while
    set $(u^{k+1}, d^{k+1}) = (x^{i-1}, y^{i-1})$
    compute $c_1^{k+1}$ and $c_2^{k+1}$ as in (9), using $u^{k+1}$
    set $k = k + 1$
end while
```

Algorithm 3.2, without use of any alternating scheme as made in *SBSeg-GBO* and *DRSSeg* algorithms.
FPG method with backtracking stepsize rule is an accelerated variant of Proximal Gradient (PG) algorithm, for solving a minimization of (10), where K is the identity operator and f is continuously differentiable.
Setting:

$$(24) \quad f(u,d) = \lambda < u, r > +\frac{\eta}{2}\|d - \nabla u - b^k\|^2 \quad \text{and} \quad g(u,d) = |d|_1 + \chi_U(u)\,,$$

the PG method produces a new approximation according to:

$$(25) \qquad (u^{k+1}, d^{k+1}) = prox_{\beta_k} g((u^k, d^k) - \beta_k \nabla(f(u^k, d^k)))\,.$$

where β_k satisfies:

$$(26) \qquad f(u^{k+1}, d^{k+1}) + g(u^{k+1}, d^{k+1}) \leq Q(u^k, d^k, u^{k+1}, d^{k+1})\,.$$

and Q is the approximation model of $F(u,d)$ defined as:

$$\begin{aligned} Q(u,d,u',d') := f(u,d) + &< (u',d') - (u,d), \nabla f(u,d) > + \\ &+ \tfrac{1}{2\beta_k}||(u',d') - (u,d)||^2 + g(u',d')\,. \end{aligned}$$

We can note that g is a separable sum of two functions that depends only on u and the other only on d; thus the proximal operator can be decomposed as two simpler operators, giving a projection in the set U for u and a shrink operator for d respectively.

4. Numerical results and comparisons

In this section we analyzed the behaviour of the Algorithms 3.1–3.3 described in the previous section. To this aim, we briefly describe implementation details and the description of the setting parameters used in the tests.

- The Algorithm 3.1, *SPGASeg*, is written in C with a MEX interface to MATLAB (for further details, see [9]), its native stopping criterion requires:

$$\left\| P_U \left(u^k - G_u \left(c_1^k, c_2^k, u^k \right) \right) - u^k \right\| \leq \texttt{tol} \text{ and } k < \texttt{maxit} \, .$$

- An implementation of *SBSeg-GBO* algorithm is available from http://www.xavier-bresson.tk. It is written in C with a MEX interface to MATLAB, and it implements a stopping criterion based on the Sum of Differences Squared defined as $\texttt{sde}(z^k) = \sum_{i=1}^{N} (z_i^k - z_i^{k-1})^2$, that requires:

$$|\texttt{diff}^{(k+1)} - \texttt{diff}^{(k)}| < \texttt{tol} \text{ and } k < \texttt{maxit} \, , \tag{27}$$

where

$$\texttt{diff}^{(k)} = \texttt{sde}(u^k) / \left(\texttt{sde}(u^k) \cdot \texttt{sde}(u^{k-1}) \right) \, .$$

GS method is used for solving the system (18).
- The implementation of *SBSeg-FPG* algorithm, was realized modifying the previous *SBSeg-GBO* code, according to (25) and (26).
- The implementation of Algorithm 3.3, *DRSSeg*, is written in MATLAB and the *DRS* solver is based on the MATLAB code available from http://www.numerical-tours.com/matlab/ [20] (see section: `Convex Region-Based Image Segmentation`) where the native stopping criterion based only on the maximum number of iteration, was modified introducing the tolerance, $\texttt{tol}_{\texttt{DRS}}$. In this implementation, the system (23) is solved by means of the IFFT/FFT algorithms, using periodic boundary conditions for the gradient operator.

In order to make a fair comparison, we used for all the algorithms the stopping criterion (27).
For all the tests, the common parameters of the Algorithms 3.1–3.3 was chosen as follows:

- μ, the parameter in (6) was set to 0.5;

- `tol` and `maxit`, the tolerance and the maximum number of iterations in (27), were set in $[10^{-12}, 10^{-6}]$ and to 1000 respectively;
- λ, the parameter in (7) was changed according to test image in order to control the level of segmentation details;
- u^0, the initial function was set to the image to be segmented with the intensity values scaled between 0 and 1.

After, the specific parameters for each algorithm were set as described below:

SPGASeg	$\epsilon = 10^{-6},\ \alpha_{min} = 10^{-30},\ \alpha_{max} = 10^{30}$ $\gamma = 10^{-4},\ \nu = 10$
SBSeg-GBO	$\eta = 1,\ \mathtt{tol}_{\mathtt{GS}} = 10^{-2},\ \mathtt{maxit}_{\mathtt{GS}} = 30$
SBSeg-FPG	$\eta = 1,\ \mathtt{tol}_{\mathtt{FPG}} = 10^{-2},\ \mathtt{maxit}_{\mathtt{FPG}} = 30$
DRSSeg	$\tau = 1,\ \mathtt{tol}_{\mathtt{DRS}} = 10^{-2},\ \mathtt{maxit}_{\mathtt{DRS}} = 30$

The four algorithms were run using MATLAB (R2013a, v. 8.1) with Image Processing Toolbox for read and display images, on Intel Core i5 processor with clock frequency of 2.7 GHz, 8 GB of RAM, and a 64 Bit Linux system. We analyzed the behaviour of the four algorithms on two sets of images with intensity values in $[0, 255]$: the first set includes three black and white (black-white) images, named *on-off*, *hoffman* and *geo* respectively, (see Figure 1), for which the ground truth[b] is available; moreover a gaussian noise with mean 0 and standard deviation 0.01 is added to the image *geo*; the second one includes three grey scale images, named *cameraman*, *squirrel* and *ninetyeight* respectively (see Figure 2).
Since the implementations of the four algorithms have different design, their efficiency is compared evaluating the computational cost (by means of the number of floating point operations (flops)) and not measuring the execution time. Let $m \times n$ the size of the image, we give an estimate of the cost of one outer iteration (the iteration on k) of each algorithm counting the operations that require $O(mn)$ flops, and in particular for *DRSSeg* we take in account the FFT and IFFT algorithms that perform $O(mn \cdot log(mn))$ flops. Note that the update of c_1^k and c_2^k is performed in each one of the four algorithms, so we do not take in account its computational cost.

[b] Generally, for one image there are several “true” segmentations that are typically made by one or more human experts. The ground truth of a black and white image is the image itself.

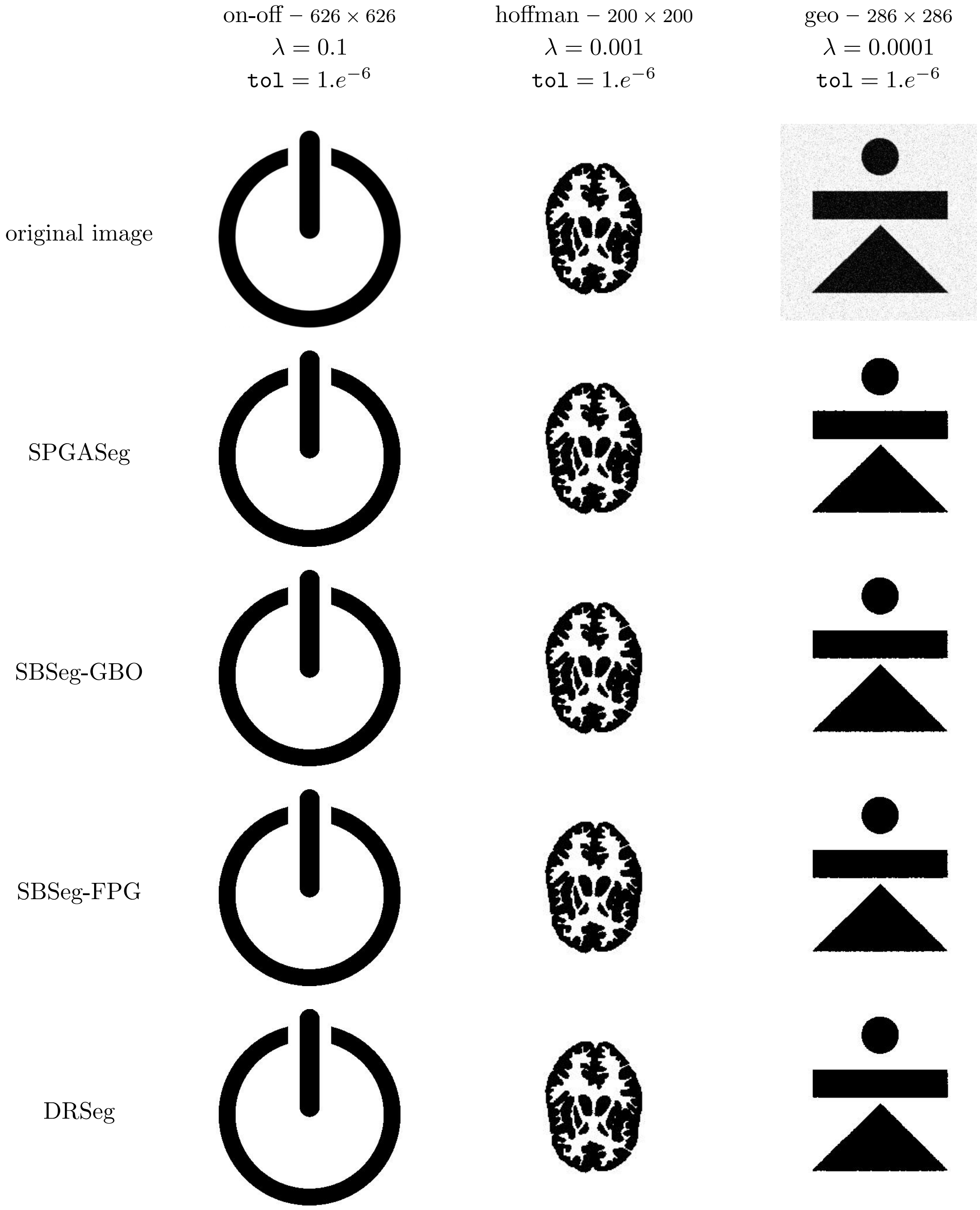

Figure 1. Synthetic pictures and corresponding segmentations by *SPGASeg*, *SBSeg-GBO*, *SBSeg-FPG* and *DRSSeg*. The number of pixels of each image is specified after the name. The images have been resized using different scalings to fit the figure layout.

- In *SPGASeg*, at k-th iteration, is performed one gradient evaluation, the computation of α_{BB} in (13), the computation of $\langle G_u^k, d_u^k \rangle$ in (14), and a number, $\mathtt{nf}_{(k)}$, of objective function evaluations.
- In *SBSeg-GBO*, at k-th iteration, is carried out the computation of r in (8), the updating of both d^{k+1} and b^{k+1} and a number, $\mathtt{n}_{\mathtt{GS}(k)}$ of GS iterations.
- In *SBSeg-FPG*, at k-th iteration, is also performed the computation of r in (8) and the updating of b^{k+1}, but conversely from *SBSeg-GBO*, are performed a number, $\mathtt{n}_{\mathtt{FPG(k)}}$, of gradient evaluation of the function f defined in (24) and a number, $\mathtt{nfq}_{(k)}$, of function evaluations of the f and Q in (26).
- In *DRSSeg* the cost of k-th iteration is mainly associated with the computation of u^k and d^k which requires $\mathtt{n}_{\mathtt{DRS(k)}}$ iterations, each iteration spend $2{\cdot}O(mn)$ flops for the update of u and $2{\cdot}O(mn(log(mn))$ flops for the update of d.

In Table 1, we summarize the cost of the k-th iteration of the Algorithms 3.1–3.3.

Table 1. Computational cost of Algorithms 3.1–3.3 of the $k-th$ iteration.

Algorithm	Computational cost for iteration
SPGASeg	$(\mathtt{nf}_{(k)} + 3) \cdot O\,(mn)$
SBSeg-GBO	$(\mathtt{n}_{\mathtt{GS}(k)} + 3) \cdot O\,(mn)$
SBSeg-FPG	$(\mathtt{n}_{\mathtt{FPG}(k)} + \mathtt{nfq}_{(k)} + 2) \cdot O\,(mn)$
DRSSeg	$2\mathtt{n}_{\mathtt{DRS}(k)} \cdot [O\,((mn \cdot log(mn)) + O\,(mn)]$

In order to complete the comparison on efficiency, in Table 2, we show the number, $\mathtt{nit}$, of the outer iterations on k, performed by the four algorithms to satisfy the stopping criterion (27) for the selected images. According with Table 1, we also show the total number, $\mathtt{nf}$, of the objective function evaluations in *SPGASeg*, and the total number, $\mathtt{nfq}$, of the evaluations of f and Q, in *SBSeg-FPG*. Furthermore, we listed the total numbers, $\mathtt{n}_{\mathtt{GS}}$, $\mathtt{n}_{\mathtt{FPG}}$ and $\mathtt{n}_{\mathtt{DRS}}$, of the inner solver iterations for *SBSeg-GBO*, *SBSeg-FPG* and *DRSSeg* algorithms respectively.

We can observe that on black-white images, the four algorithms performed the same number $\mathtt{nit}$ of outer iterations. In details, *SPGASeg* and *SBSeg-GBO* shown a comparable efficiency, because performed about the same value of $\mathtt{nf}$ and $\mathtt{n}_{\mathtt{GS}}$ respectively, which have the same computational cost as shown in Table 1. Differently, although *SBSeg-FPG* performed values of $\mathtt{n}_{\mathtt{FPG}}$ equal to $\mathtt{nf}$ and $\mathtt{n}_{\mathtt{GS}}$, the efficiency is lower than the previous algorithms, because of the charge of $\mathtt{nfq}$ evaluations. Similar result was achieved

Table 2. Details about the execution of *SPGASeg*, *SBSeg-GBO*, *SBSeg-FPG* and *DRSSeg* on test images shown in Fig. 1 and Fig. 2.

Image	*SPGASeg*		*SBSeg-GBO*		*SBSeg-FPG*			*DRSSeg*	
	`nit`	`nf`	`nit`	$\mathtt{n}_{\mathtt{GS}}$	`nit`	$\mathtt{n}_{\mathtt{FPG}}$	`nfq`	`nit`	$\mathtt{n}_{\mathtt{DRS}}$
on-off	2	5	2	4	2	4	8	2	4
hoffman	2	5	2	4	2	4	8	2	26
geo	2	5	2	5	2	6	13	2	12
cameraman	5	11	4	13	4	15	30	4	31
squirrel	147	248	41	206	535	2297	4112	17	438
ninetyeight	8	17	8	63	5	21	40	5	94

executing *DRSSeg* on image *on-off*, indeed the value of $\mathtt{n}_{\mathtt{DRS}}$ is about the same of `nf` and $\mathtt{n}_{\mathtt{GS}}$ but as shown in Table 1 the computational cost of inner solver is greater than the other two algorithms. For more reason, since on the other two black-white images *DRSSeg* performed the greatest number of inner iterations, it is not competitive with *SPGASeg* and *SBSeg-GBO*. Similar results of the behaviour of the four algorithms, can be deduced on grey scale image, *cameraman*. Conversely, on *squirrel* and *ninetyeight* that have a non uniform background, the more efficient algorithm was *SB-GBO* and *SPGASeg* respectively.

Regarding the accuracy of Algorithms 3.1–3.3, we listed in Table 3 the values of the objective function E in (7), evaluated both at u^0, and at the approximation, $u^{\mathtt{nit}}$, carried out at `nit` iteration of Table 2. For the sake of simplicity, we will indicate in this section, with $E^{(0)} := E(c_1^0, c_2^0, u^0)$ and with $E^{(\mathtt{nit})} := E(c_1^{\mathtt{nit}}, c_2^{\mathtt{nit}}, u^{\mathtt{nit}})$.

We can note that on two black-white images, *on-off* and *hoffman*, *SPGASeg*, *SBSeg-GBO* and *SBSeg-FPG* reached the same value of $E^{(\mathtt{nit})}$, instead of the value of $E^{(\mathtt{nit})}$ carried out by *DRSSeg* algorithm is higher than the other ones. Moreover, on noised image, *geo*, *SPGASeg* shown the lowest value of $E^{(\mathtt{nit})}$. Conversely, on grey scale images the lowest value of $E^{(\mathtt{nit})}$ was reached by *DRSSeg*.

Table 3. Evaluations of the objective function (7) at u^0, $E^{(0)}$, and at computed solution, $E^{(\mathtt{nit})}$, respectively.

Image	$E^{(0)}$	*SPGASeg* $E^{(\mathtt{nit})}$	*SBSeg-GBO* $E^{(\mathtt{nit})}$	*SBSeg-FPG* $E^{(\mathtt{nit})}$	*DRSSeg* $E^{(\mathtt{nit})}$
on-off	4.47707e+06	1.45885e+06	1.45885e+06	1.45885e+06	4.10918e+06
hoffman	1.42093e+04	2.35818e+03	2.35818e+03	2.35818e+03	2.44027e+03
geo	2.70916e+04	3.86373e+03	4.04925e+03	4.05665e+03	3.98271e+03
cameraman	6.33818e+04	1.53337e+04	1.51511e+04	1.61450e+04	1.50890e+04
squirrel	1.57972e+04	5.52358e+03	5.06419e+03	5.64449e+03	5.06408e+03
ninetyeight	1.47830e+05	2.71111e+04	2.62707e+04	2.88795e+04	2.62372e+04

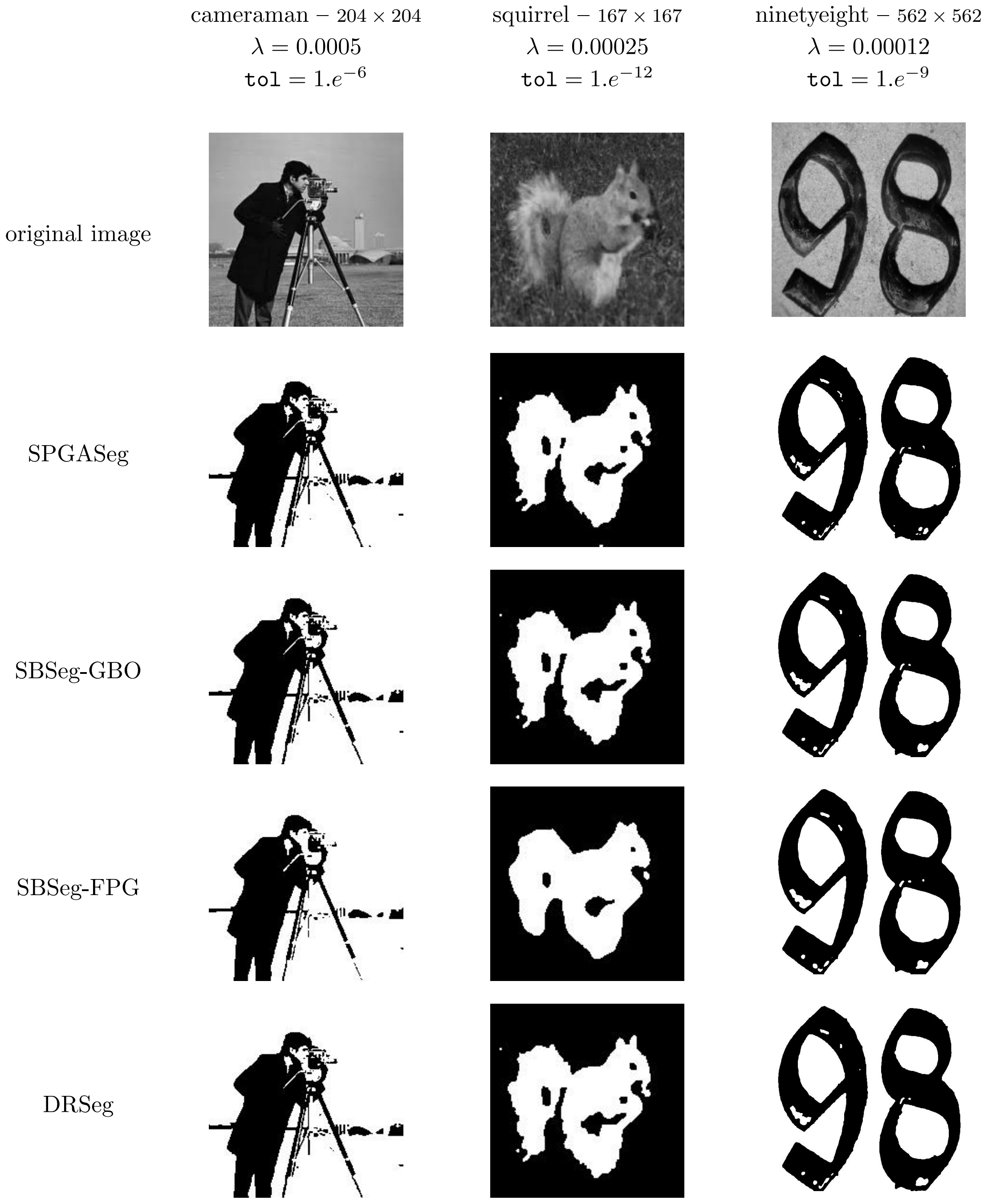

Figure 2. Real images and corresponding segmentations by SPGASeg, SBSeg-GBO, SBSeg-FPG and DRSSeg. The number of pixels of each image is specified after the name. The images have been resized using different scalings to fit the figure layout.

In order to evaluate the accuracy of the produced segmentations by the four algorithms, on black-white images we applied two standard metrics of the segmentation error measures, namely: the Global Consistency Error (GCE) [21] and the Rand Index (RI) [22][c]. These metrics provides a measure of similarity between the produced segmentation and the ground truth, assigning a real valued output in the range $[0, 1]$. When the segmentation is consistent with respect to the ground truth, the values of RI should be higher and on the other hand the value of GCE should be low. The results of GCE and RI applied to black-white images are shown in 4. According with the results of Table 3, we can see that on the image *hoffman*, GCE and RI assigned the same value to produced segmentations, instead of on image *on-off*, the value assigned to segmentation produced by *DRSSeg*, was slight highest. Moreover, confirming the results of Table 3, the segmentation of *geo* produced by *SPGASeg* seem more consistent with the corresponding ground-truth. Anyway, the two metrics proved that the quality of the segmentations produced by the four algorithms on the synthetic images is comparable as we can see in Figure 1.

Table 4. Segmentation error measures produced by GCE and RI for *SPGASeg*, *SBSeg-GBO*, *SBSeg-FPG* and *DRSSeg* on the black-white images of Figure 1.

algorithm	*on-off*		*hoffman*		*geo*	
	GCE	RI	GCE	RI	GCE	RI
SPGASeg	0.111164	0.914595	0.2058	0.889554	0.084106	0.474784
SBSeg-GBO	0.111164	0.914595	0.2058	0.889554	0.092775	0.475307
SBSeg-FPG	0.111164	0.914595	0.2058	0.889554	0.092770	0.475812
DRSSeg	0.111231	0.914868	0.2058	0.889554	0.086130	0.474411

Finally, in Figure 2 we show the segmentations produced by the four algorithms on the grey scale images. We can see that except for *SBSeg_FPG*, the quality of the segmentations produced by the other three algorithms is comparable and a good two-phase partition of each image is realized separating object regions from the background. On the other hand, we note that *SBSeg-FPG* produces generally an under segmentation of the test images especially for *cameraman* and *squirrel.* This result confirms that the behaviour of the Bregman method is largely depending on how the subproblem in Algorithm 3.2 is solved. Indeed, different from the two variants *SBSeg_GBO* and *DRSSeeg*, *SBSeg_FPG* provides an inexact solution not only for u but also for d. Therefore, the tests on this set of images indicate that the use of an exact minimization (oppure a closed form solution for the minimization problem) is most appropriate for producing accurate

[c] A software implementing GCE and RI is available to follow link: https://people.eecs.berkeley.edu/~yang/software/lossy_segmentation/.

segmentations of the grey scale images.

Conclusion

In this work, a comparison of some first order optimization methods for solving the nonsmooth convex relaxation of Chan-Vese model was presented. Two typical approaches to deal with non differentiability of the model under investigation were considered: the *smoothing methods* and the *nonsmoothing methods*. In particular, a smoothing method, *SPGASeg*, based on nonmonotone projected gradient method was compared with nonsmoothing ones based on proximal operators and splitting schemes. The results shown that the accuracy of the two approaches is comparable, so the smoothness introduced by *SPGASeg* has no effect on the produced segmentations. Also, the efficiency of *SPGASeg* was comparable with *SBSeg-GBO*, despite that SB methods usually show the best performances. Regarding the nonsmoothing approach, numerical tests evidenced that generally *SBSeg-GBO* was more efficient and accurate than the version *SBSeg-FPG*, instead of *DRSeg* sometimes was more accurate than *SBSeg-GBO* because the use of a different inner solver. Anyway, confirming results of the literature, in general *SBSeg-GBO* yields very accurate solutions even if subproblems are not solved accurately.

Acknowledgements

The authors thank Daniela di Serafino for her valuable comments and suggestions on the manuscript.
This work is partially supported by INdAM-GNCS, under the 2017 Projects *Metodi numerici per problemi di ottimizzazione vincolata di grandi dimensioni e applicazioni* and by MIUR, under FIRB 2010 Project n. RBFR106S1Z002.

REFERENCES

1. T. Chan and L. Vese, Active contours without edges, *IEEE Transactions on Image Processing*, vol. 10, no. 2, pp. 266–277, 2001.
2. D. Mumford and J. Shah, Optimal approximations by piecewise smooth functions and associated variational problems, *Communications on Pure and Applied Mathematics*, vol. 45, no. 2, pp. 577–685, 1989.
3. T. Chan, S. Esedoḡlu, and M. Nikolova, Algorithms for finding global minimizers of image segmentation and denoising models, *SIAM Journal on Applied Mathematics*, vol. 66, no. 5, pp. 1632—1648, 2006.
4. X. Bresson, S. Esedoḡlu, P. Vandergheynst, J.-P. Thiran, and S. Osher,

Fast global minimization of the active contour/snake model, *Journal of Mathematical Imaging and Vision*, vol. 28, no. 2, pp. 151–167, 2007.

5. R. Yildizoğlu, J.-F. Aujol, and N. Papadakis, Active contours without level sets, in *ICIP 2012 – IEEE International Conference on Image Processing (Orlando, FL, Sept. 30 – Oct. 3, 2012)*, pp. 2549–2552, IEEE, 2012.
6. A. Chambolle and T. Pock, A first-order primal-dual algorithm for convex problems with applications to imaging, *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2011.
7. J. Yuan, E. Bae, and X. Tai, A study on continous max-flow and min-cut approaches, in *Computer Vision and Pattern Recognition*, pp. 2217–2224, IEEE, 2010.
8. G. Paul, J. Cardinale, and I. F. Sbalzarini, Coupling image restoration and segmentation: A generalized linear model/bregman perspective, *International Journal of Computer Vision*, vol. 104, pp. 69–93, 2013.
9. L. Antonelli, V. De Simone, and D. di Serafino, On the application of the spectral projected gradient method in image segmentation, *J. Math. Imaging Vis.*, vol. 54, pp. 106–116, Jan. 2016.
10. A. Beck and M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, *SIAM J. Img. Sci.*, vol. 2, pp. 183–202, Mar. 2009.
11. E. Birgin, J. Martínez, and M. Raydan, Spectral projected gradient methods: review and perspectives, *Journal of Statistical Software*, vol. 60, no. 3, 2014.
12. J. Barzilai and J. Borwein, Two-point step size gradient methods, *IMA Journal of Numerical Analysis*, vol. 8, pp. 141–148, 1988.
13. D. di Serafino, V. Ruggiero, G. Toraldo, and L. Zanni, On the steplength selection in gradient methods for unconstrained optimization, *Applied Mathematics and Computation*, 2017.
14. L. Grippo, F. Lampariello, and S. Lucidi, A nonmonotone line search technique for Newton's method, *SIAM Journal on Numerical Analysis*, vol. 23, no. 4, pp. 707–716, 1986.
15. T. Goldstein, X. Bresson, and S. Osher, Geometric applications of the split Bregman method: segmentation and surface reconstruction, *Journal of Scientific Computing*, vol. 45, no. 1–3, pp. 272–293, 2010.
16. L. Bregman, The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex programming, *USSR Computational Mathematics and Mathematical Physics*, vol. 7, pp. 69–93, 1967.
17. S. Setzer, Split bregman algorithm, douglas-rachford splitting and frame shrinkage, in *Proc. of the Second International Conference on*

Scale Space Methods and Variational Methods in Computer, vol. 5567, pp. 464–476, 2009.

18. O. Güler, New proximal point algorithms for convex minimization, *SIAM Journal on Optimization*, vol. 2, no. 4, pp. 649–664, 1992.
19. Y. Nesterov, A method of solving a convex programming problem with convergence rate o (1/k2), *Soviet Mathematics Doklady*, vol. 27, pp. 372–376, 1983.
20. G. Peyré, The numerical tours of signal processing, *Advanced Computational Signal and Image Processing IEEE Computing in Science and Engineering*, vol. 13, no. 4, pp. 94–97, 2011.
21. D. Martin, C. Fowlkes, D. Tal, and J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in *Proc. 8th Int'l Conf. Computer Vision*, vol. 2, pp. 416–423, July 2001.
22. W. M. Rand, Objective criteria for the evaluation of clustering methods, *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.