A level set method for solving free boundary problems associated with obstacles ⋆

K. Majava

University of Jyväskylä, Department of Mathematical Information Technology, P.O. Box 35 (Agora), FIN-40014 University of Jyväskylä, Finland

X.-C. Tai

University of Bergen, Department of Mathematics, Johannes Brunsgate 12, N-5009 Bergen, Norway

Abstract

A method related to the level set method is proposed for solving free boundary problems coming from contact with obstacles. Two different approaches are described and applied for solving an unilateral obstacle problem. The cost functionals coming from the approach are nonsmooth. For solving the nonsmooth minimization problems, two methods are applied: firstly, a proximal bundle method, which a method for solving general nonsmooth optimization problems. Secondly, a gradient method is proposed for solving the regularized problems. Numerical experiments are included to verify the convergence of the methods and the quality of the results.

Key words: Level set methods, free boundary problems, obstacle problem

1 Introduction

The level set method initiated by Osher and Sethian [1] has proven to be an efficient numerical device for capturing moving fronts, see [2–4]. There are many industrial problems where interfaces need to be identified, which can be formulated as moving front problems. We mention, for example, image segmentation

⋆ This research was partially supported by the Academy of Finland, NorFA (Nordisk Forskerutdanningsakademi), and the research council of Norway.

Email addresses: majkir@mit.jyu.fi (K. Majava), tai@mi.uib.no (X.-C. Tai).

URLs: http://www.mit.jyu.fi/majkir (K. Majava), http://www.mi.uib.no/%7Etai (X.-C. Tai).

problems [5], inverse problems [6], and optimal shape design problem [7,8]. In free boundary problems, it is often needed to find the boundary of some domains. It is natural to use level set method for this kind of applications. In [3,9], the level set method was used for Stefan type of free boundary problems. In this work, we shall propose an alternative approach for the level set idea of [1,3,9] and use it for tracing the free boundaries from obstacle contact type of problems.

The contents of the paper are as follows. In Section 2, a model free boundary problem is described and the proposed level set approaches are introduced for solving the problem. In Section 3, the solution algorithms are described that are applied for realizing the proposed level set approaches. In Section 4, numerical experiments are presented to verify the convergence of the methods and the quality of the results. Finally, in Section 5, the conclusions are stated.

2 A modified level set method

Consider a model free boundary problem which comes from the minimization problem:

$$\min_{v \in K} F(v), \tag{1}$$

with

$$F(v) = \int_\Omega \left(\frac{1}{2} |\nabla v|^2 - fv \right) dx, \qquad K = \{v |\ v \in H_0^1(\Omega), v \geq \psi\}. \tag{2}$$

In the above, $\Omega \subset R^p$, $p = 1, 2$, ψ is the obstacle function satisfying $\psi \leq 0$ on $\partial\Omega$, and f typically represents external force for physical problems. The solution u for (2) is unique and it can be formally written as the function satisfying

$$-\Delta u \geq f, \quad u \geq \psi, \quad (-\Delta u - f) \cdot (u - \psi) = 0.$$

To find the solution u, we need to find the contact region $\Omega^+ = \{x |\ \ u(x) = \psi(x), x \in \Omega\}$. Once we know Ω^+, the value of u in $\Omega \backslash \Omega^+$ can be obtained from solving

$$-\Delta u = f \text{ in } \Omega \backslash \Omega^+, \quad u = 0 \text{ on } \partial\Omega, \quad u = \psi \text{ on } \partial\Omega^+.$$

In order to find u, we essentially just need to find $\Gamma = \partial\Omega^+$. Inside Γ, $u = \psi$ and outside Γ, u is the solution of the Poisson equation.

Based on the above observation, we see that it is essentially enough to find the curve in order to solve the free boundary problem (1). Starting with an initial curve, we shall slowly evolve the curve to the true free boundary. We can use the level set method to represent the curve, i.e., we try to find a function $\varphi(t, x)$ such that

$$\Gamma(t) = \{x |\ \varphi(t, x) = 0\}.$$

In the above, $\Gamma(0)$ is the initial curve and $\Gamma(t)$ converges to the true free boundary, when $t \to \infty$. One of the essential ingredient of the level set method is to find the velocity field $\vec{V}(t,x)$, which is then used to move the level set function $\varphi(t,x)$ by solving

$$\varphi_t - \vec{V}|\nabla\varphi| = 0, \quad \varphi(0,x) = \varphi_0(x) = \pm\text{distance}(x,\Gamma(0)).$$

In this work, we propose an alternative approach, which seems to be simpler than the approach outlined above. Define the Heaviside function $H(\varphi)$ as

$$H(\varphi) = \begin{cases} 1, & \varphi > 0, \\ 0, & \varphi \le 0. \end{cases}$$

For any $v \in K$, there is exists $\varphi \in H^1(\Omega)$ such that

$$v = \psi + \varphi H(\varphi). \tag{3}$$

It is easy to see that

$$v = \begin{cases} \psi, & if \quad \varphi \le 0 \quad \text{(i.e., in the contact region)} \\ \psi + \varphi, & if \quad \varphi > 0 \quad \text{(i.e., outside the contact region).} \end{cases} \tag{4}$$

Thus, the sign of the function φ tells the information of the contact region. The curve, which separates the regions where φ is positive or negative, gives the free boundary. In the traditional level set method, the function φ is only used to represent the curve Γ. In our approach, φ is not only used to represent the curve, but also to carry information about the solution u outside the contact region, i.e., φ is used to indicate that $u = \psi$ inside the contact region and its value outside the contact region shall be $\varphi = u - \psi$. We use iterative type of methods to find the correct values of φ both inside and outside the contact region. Note that the value of φ inside the contact region is not unique.

Representation (3) is not the only formulation that can be used to express the value of a function $v \in K$ as a function of $\varphi \in H^1(\Omega)$. In fact, there exists a function $\varphi \in H^1(\Omega)$ for every $v \in K$ such that

$$v = \psi + \frac{1}{2}(\varphi + |\varphi|). \tag{5}$$

For the above representation for v, we see that (4) is also correct. From now on, we use a shortened notation

$$\mathcal{J}_i(\varphi) := F(v_i(\varphi)), \quad \text{for} \quad i = 1, 2, \tag{6}$$

where

$$v_1(\varphi) = \psi + \varphi H(\varphi) \quad \text{and} \quad v_2(\varphi) = \psi + \frac{1}{2}(\varphi + |\varphi|). \tag{7}$$

On the boundary $\partial\Omega$, we have $\varphi = -\psi$ for both of the approaches (i.e., for $i = 1, 2$). Thus, the function φ assumes a Dirichlet boundary condition. For simplicity, we assume from now on that $\psi = 0$ on $\partial\Omega$ so that we have $\varphi \in H_0^1(\Omega)$ for both of the approaches.

Consider now the following unconstrained minimization problem

$$\min_{\varphi \in H_0^1(\Omega)} \mathcal{J}_i(\varphi), i = 1, 2. \tag{8}$$

Due to the use of the Heaviside function and absolute value of functions in approaches 1 and 2, the cost functional $\mathcal{J}_i$ is not differentiable. However, we can prove that the problem has a solution:

Theorem 1 *The non-smooth minimization problem (8) has a minimizer for $i = 1, 2$. The minimizer may not be unique.*

PROOF. For minimization problem (1), the cost functional is strictly convex and continuous in $H_0^1(\Omega)$ and K is closed in $H_0^1(\Omega)$, see [10, p.29]. Thus, there exists a unique $u \in K$ such that

$$F(u) \leq F(v) \quad \forall v \in K. \tag{9}$$

Associated with this unique solution u, let us define

$$\varphi^*(x) = \begin{cases} u(x) - \psi(x), & if \quad u(x) > \psi(x) \quad \text{(i.e., outside the contact region)} \\ 0, & if \quad u(x) = \psi(x) \quad \text{(i.e., inside the contact region).} \end{cases} \tag{10}$$

With φ^* given above, we have $u = v_i(\varphi^*), i = 1, 2$. For any $\varphi \in H_0^1(\Omega)$, we have $v_i(\varphi) \in K$. Thus, we get from (9) that

$$\mathcal{J}_i(\varphi^*) \leq \mathcal{J}_i(\varphi) \quad \forall \varphi \in H_0^1(\Omega), \ i = 1, 2. \tag{11}$$

This means that φ^* is a minimizer for (8). The value of φ^* inside the contact region can be any negative value, which means that the minimizer is non-unique. $\square$

3 Solution algorithms

Minimization problem (8) has a minimizer. However, the minimizer is non-unique and the cost functional $\mathcal{J}_i$ is non-convex. In order to find a minimizer for it, we shall use two algorithms. The first one is the proximal bundle method, which is a popular method for finding local minimizers for non-smooth and

non-convex minimization problems. For the second algorithm, we turn the non-smooth minimization problem into a differentiable problem by regularizing the non-smooth functions used in the cost functionals and then use a gradient method to find a minimizer for the smoothed problem.

3.1 Proximal bundle method (PB)

In what follows, we introduce shortly the proximal bundle method, which is a method for finding a local minimum of a general unconstrained nonlinear optimization problem

$$\min_{x \in R^n} \quad \mathcal{J}(x), \tag{12}$$

where $\mathcal{J} : R^n \to R$. We assume that $\mathcal{J}$ in (12) is a locally Lipschitz continuous function. Thus, $\mathcal{J}$ may be non-differentiable but it has a subdifferential at each point x. The subdifferential $\partial\mathcal{J}$ (of Clarke [11]) of $\mathcal{J}$ at $x \in R^n$ is defined by

$$\partial\mathcal{J}(x) = \text{conv}\{\xi \in R^n : x^i \to x, \exists \nabla\mathcal{J}(x^i) \text{ and } \nabla\mathcal{J}(x^i) \to \xi\}.$$

Element $\xi \in \partial\mathcal{J}(x)$ is called a subgradient. If $\mathcal{J}$ is differentiable at x, then $\partial\mathcal{J}(x) = \{\nabla\mathcal{J}(x)\}$.

The basic idea in bundle methods is to approximate the whole subdifferential by collecting subgradients calculated in a neighbourhood of the considered point x. At each iterate, this approximation of the subdifferential is used to build up a local model of the original problem. The origin of the methods is the classical cutting plane method [12], where a piecewise linear approximation of the objective function is formed. In bundle methods, a stabilizing quadratic term is added to the polyhedral approximation in order to accumulate some second order information about the curvature of the objective function.

PB method assumes that at each point $x \in R^n$, we can evaluate the function value $\mathcal{J}(x)$ and an arbitrary subgradient $g(x)$ from the subdifferential $\partial\mathcal{J}(x)$. For the convergence of the method, in the nonconvex case, $\mathcal{J}$ is further assumed to be upper semismooth [13,14]. Some details of the PB method are given in Appendix of this paper. For more details, see [13,15,16].

The tested proximal bundle algorithm is from the software package NSOLIB. The code utilizes the subgradient aggregation strategy of [15] to keep the storage requirements bounded, and the safeguarded quadratic interpolation algorithm of [13] to control the size of the search region. For solving the quadratic subproblem, the code employs the quadratic solver QPDF4, which is based on the dual active-set method described in [17].

3.2 Smoothed approximations and the gradient method

Because the Heaviside function is nondifferentiable at $\varphi = 0$, we replace it by a smooth approximation

$$H_\varepsilon(\varphi) = \frac{1}{\pi} tan^{-1} \frac{\varphi}{\varepsilon} + \frac{1}{2},$$

for small positive ε. The gradient of $H_\varepsilon(\varphi)$ is denoted by $\delta_\varepsilon(\varphi)$ (the smoothed Dirac delta function):

$$\delta_\varepsilon(\varphi) = \frac{\varepsilon}{\pi(\varphi^2 + \varepsilon^2)}.$$

For approach 1, i.e., for $i = 1$ in (6)–(7), we then have

$$\frac{\partial F}{\partial v_1} = -\Delta v_1 - f \quad \text{and} \quad \frac{\partial v_1}{\partial \varphi} = H_\varepsilon(\varphi) + \varphi \delta_\varepsilon(\varphi).$$

Correspondingly, we have

$$\frac{\partial \mathcal{J}_1}{\partial \varphi} = \frac{\partial F}{\partial v_1} \frac{\partial v_1}{\partial \varphi} = (-\Delta v_1 - f)(H_\varepsilon(\varphi) + \varphi \delta_\varepsilon(\varphi)). \tag{13}$$

Since also $|\varphi|$ is nondifferentiable at 0, we use again a smooth approximation $|\varphi| \approx \sqrt{\varphi^2 + \hat{\varepsilon}}$, for small positive $\hat{\varepsilon}$ for approach 2, i.e., for $i = 2$ in (6)–(7). Then,

$$\frac{\partial v_2}{\partial \varphi} = \frac{1}{2} \left(1 + \frac{\varphi}{\sqrt{\varphi^2 + \hat{\varepsilon}}} \right),$$

and, altogether, we have

$$\frac{\partial \mathcal{J}_2}{\partial \varphi} = \frac{\partial F}{\partial v_2} \frac{\partial v_2}{\partial \varphi} = \frac{1}{2}(-\Delta v_2 - f) \left(1 + \frac{\varphi}{\sqrt{\varphi^2 + \hat{\varepsilon}}} \right). \tag{14}$$

Let $\mathcal{J}_i^\varepsilon$ denote the corresponding cost functionals with the smoothed functions. We shall use the following gradient method (GR) to find a function φ, which approximates the minimizers of (8):

$$\varphi^{n+1} = \varphi^n - \alpha \frac{\partial \mathcal{J}_i^\varepsilon}{\partial \varphi}(\varphi^n), \quad i = 1, 2.$$

The step size α is fixed and is obtained by trial and error approach.

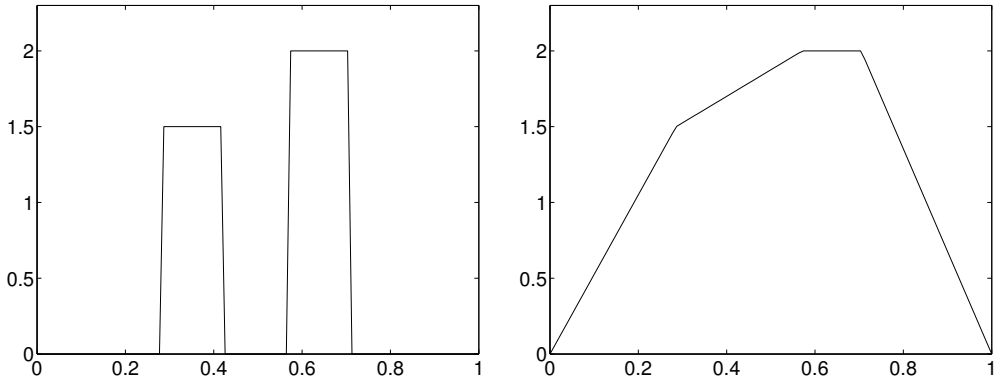

Fig. 1. The obstacle (left) and the analytical solution (right) for $n = 100$.

4 Numerical experiments

In this section, we present the results of the numerical experiments that were computed using the proposed algorithms. All experiments are performed on an HP9000/J5600 workstation (2×552 MHz PA8600 CPU) and the algorithms are implemented with Fortran 77.

4.1 General issues about the experiments

The following two example problems were considered. Both problems have $f = 0$.

Example 2 *In this one-dimensional example, we chose* $\Omega = [0, 1]$. *The obstacle and the analytical solution are shown in Figure 1.*

Example 3 *In this example, we chose* $\Omega = [-2, 2] \times [-2, 2]$ *and*

$$\psi(x, y) = \begin{cases} \sqrt{1 - x^2 - y^2}, & \text{for } \ x^2 + y^2 \leq 1, \\ -1, & \text{elsewhere.} \end{cases} \tag{15}$$

With the consistent Dirichlet boundary condition, problem (1) *with* ψ *introduced in* (15) *has an analytical solution of the form*

$$u(x, y) = \begin{cases} \sqrt{1 - x^2 - y^2}, & \text{for } \ r \leq r^*, \\ -(r^*)^2 ln(r/R)/\sqrt{1 - (r^*)^2}, & \text{for } \ r \geq r^*, \end{cases}$$

where $r = \sqrt{x^2 + y^2}, R = 2,$ *and* $R^* = 0.6979651482\ldots,$ *which satisfies*

$$(r^*)^2(1 - ln(r^*/R)) = 1.$$

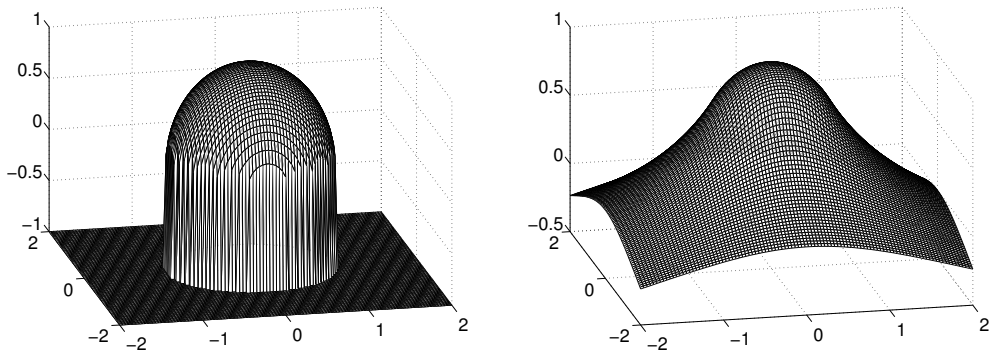

Fig. 2. The obstacle (left) and the analytical solution (right) for $\bar{n} = 100$.

The obstacle and the analytical solution are illustrated in Figure 2. The same example has been considered, e.g., in [18].

Discretization of the problems. In the one-dimensional case, $\Omega = [0, 1]$ is divided into n subintervals, and we denote $h = 1/(n + 1)$. In the two-dimensional case, $\Omega = [-2, 2] \times [-2, 2]$ is divided into $\bar{n}$ subintervals in x-direction and y-direction. We denote $n = \bar{n}^2$ and $h = (4/(\bar{n}+1)$. Then, u, f, φ, and ψ denote vectors in R^n, whose components correspond to the values of the unknown functions in the equidistant discretization points in Ω. A finite difference approximation is used for $-\Delta$ with Dirichlet boundary conditions.

How to apply PB for solving (8)**.** Even if it is possible to apply PB for solving problem (8) with $\varepsilon, \hat{\varepsilon} = 0$, our main focus in this paper is to solve the problems with $\varepsilon, \hat{\varepsilon} > 0$. In numerical simulations, the level set function φ is seldom exactly zero at a nodal point. Hence, special care needs to be taken to deal with the case, where φ at a given node has different sign than at one of the neighbouring nodes. Our numerical tests also show that the cost of PB for large problem sizes is much more expensive than the cost of GR (gradient) method. Hence, PB is primarily used only as a tool to justify that the results obtained using GR are correct. However, we have applied PB also with $\hat{\varepsilon} = 0$ for approach 2.

In order to apply PB for realizing the level set approaches of Section 2, i.e., for solving problem (8), we need to be able to compute the cost function value $\mathcal{J}_i(\varphi)$ and a subgradient $g(\varphi) \in \partial\mathcal{J}_i(\varphi)$ at each point $\varphi \in R^n$. In the nonsmooth case (i.e., without the regularization using $\hat{\varepsilon}$), we only need to supply one of the subgradients at each φ. For the smoothed cases, i.e., for $\varepsilon, \hat{\varepsilon} > 0$, the cost functional $\mathcal{J}_i(\varphi)$ is differentiable, so that instead of calculating a subgradient, we set $g(\varphi) = \frac{\partial}{\partial\varphi}\mathcal{J}_i(\varphi)$ due to (13) or (14).

Initial values and stopping criteria. In the experiments, we chose $\varphi^0 = 1$ for the initial value and the boundary condition needs to be properly supplied. PB with $\hat{\varepsilon} = 0$ converged to a different solution when the initial value $\varphi^0 = 0$ was used. However, during the experiments, a surprising observation was made. Despite the nonconvexity of the problems, when $\varepsilon, \hat{\varepsilon} > 0$ were chosen properly, both PB and GR converged to the same result, regardless of the value $\varphi_0 \geq 0$.

We have used the stopping criterion $||\frac{\partial}{\partial \varphi^n} \mathcal{J}_i(\varphi^n)|| < 10^{-2}$ for GR. For PB, the stopping criteria are more complicated and they are explained in the Appendix.

4.2 Results of the experiments

GR vs. PB. In the first experiments, we tested if GR obtains the same results as PB for $\varepsilon, \hat{\varepsilon} > 0$. For this purpose, we considered three different problem sizes for both examples. In Example 2, $n = 30, 50, 100$ and in Example 3, $\bar{n} = 10, 30, 50$. PB is not able to solve larger problems than $n = 50 \times 50$ due to memory problems. The values of the smoothing parameters $\varepsilon, \hat{\varepsilon}$ were chosen to be the smallest ones (as powers of h) for which both algorithms converged to the same solution from both $\varphi_0 = 0$ and $\varphi_0 = 1$. In the one-dimensional case, we chose $\varepsilon = h$, $\hat{\varepsilon} = h^2$, and in the two-dimensional case, $\varepsilon = h^2$, $\hat{\varepsilon} = h^4$.

Table 1
Results for Example 2.

n	Appr	*Alg*	*it*	*CPU*	$e(u^*, \bar{u})$	α	$F(u^*)$
30	1	GR	225729	16.11	$2.01 \cdot 10^{-2}$	10^{-4}	10.389331
		PB	3534	1.79	$2.01 \cdot 10^{-2}$		10.389239
30	2	GR	1859277	29.63	$1.30 \cdot 10^{-2}$	10^{-4}	10.507347
		PB	34523	16.52	$1.30 \cdot 10^{-2}$		10.506380
50	1	GR	163496	19.10	$2.54 \cdot 10^{-2}$	10^{-4}	10.583929
		PB	2968	2.44	$2.54 \cdot 10^{-2}$		10.583918
50	2	GR	14608221	376.77	$2.20 \cdot 10^{-2}$	10^{-5}	10.656211
		PB	34320	26.14	$2.20 \cdot 10^{-2}$		10.655964
100	1	GR	1054158	242.09	$6.37 \cdot 10^{-3}$	10^{-5}	11.049151
		PB	2125	3.30	$6.38 \cdot 10^{-2}$		11.049157
100	2	RR	10583641	529.16	$4.21 \cdot 10^{-3}$	10^{-5}	11.087094
		PB	47293	75.11	$4.21 \cdot 10^{-3}$		11.087025

Table 2
Results for Example 3.

n	Appr	Alg	it	CPU	$e(u^*, \bar{u})$	α	$F(u^*)$	$e(u^*, U)$
10*10	1	GR	7124	1.03	$2.73 \cdot 10^{-2}$	10^{-2}	3.476744	
		PB	2222	3.68	$2.78 \cdot 10^{-2}$		3.474095	
10*10	2	GR	344365	16.96	$5.15 \cdot 10^{-3}$	10^{-3}	3.842950	
		PB	48173	56.41	$6.01 \cdot 10^{-3}$		3.826969	
30*30	1	GR	34991	39.84	$3.54 \cdot 10^{-3}$	10^{-3}	3.872834	
		PB	4742	38.00	$3.55 \cdot 10^{-3}$		3.872785	
30*30	2	GR	200238	66.85	$8.93 \cdot 10^{-4}$	10^{-3}	3.919601	
		PB	48509	422.62	$8.83 \cdot 10^{-4}$		3.919754	
50*50	1	GR	24599	76.00	$1.27 \cdot 10^{-3}$	10^{-3}	3.916609	
		PB	18135	285.30	$1.27 \cdot 10^{-3}$		3.916593	
50*50	2	GR	1535794	1340.53	$3.40 \cdot 10^{-4}$	10^{-4}	3.933813	
		PB	204773	2924.01	$3.40 \cdot 10^{-4}$		3.933813	
63*63	1	GR	212296	1038.48	$7.27 \cdot 10^{-4}$	10^{-4}	3.926376	$5.95 \cdot 10^{-4}$
63*63	2	GR	1374101	1862.80	$1.32 \cdot 10^{-4}$	10^{-4}	3.937281	$2.09 \cdot 10^{-5}$
127*127	1	GR	132984	2615.49	$1.90 \cdot 10^{-4}$	10^{-4}	3.940715	$1.50 \cdot 10^{-4}$
127*127	2	GR	9633471	51064.97	$4.47 \cdot 10^{-5}$	10^{-5}	3.943423	$3.65 \cdot 10^{-6}$

The results are given in Tables 1–2, where in the first column, the size of the problem is given. In the next column, we state which of the level set approaches is considered. Then, the solution algorithm used is given. In the next two columns, it denotes the number of iterations needed and CPU the elapsed CPU time in seconds. In the sixth column, $e(u^*, \bar{u})$ denotes the average error between the analytical solution $\bar{u}$ and the obtained result u^* :

$$e(u^*, \bar{u}) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (u^* - \bar{u})_i^2}.$$

In the seventh column, the value of the constant step size α is given for GR. We always chose the largest value of α for which the algorithms converged. In the next column, $F(u^*)$ denotes the value of the cost functional for the obtained result.

From the tables, we conclude that with the chosen values of $\varepsilon, \hat{\varepsilon}$, GR obtains the same results as PB, so that we can trust that GR is able to solve the problems. PB is faster than GR for small problems, but it gets slow when the

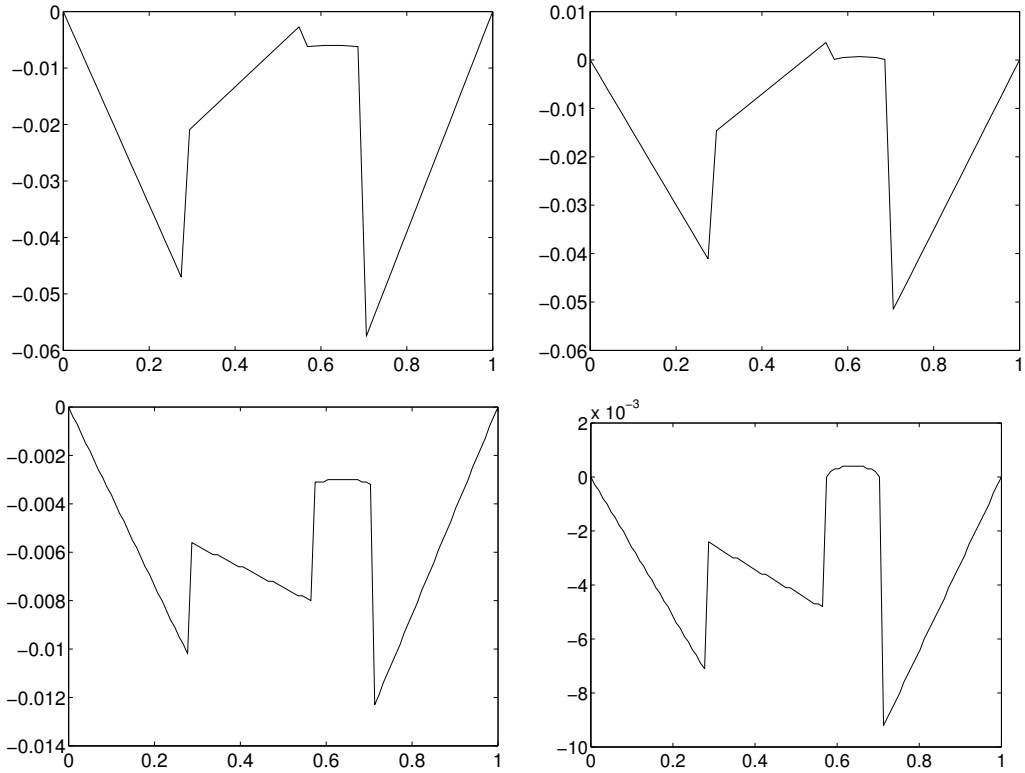

Fig. 3. Difference $(u^* - u)$ between the computed solution u^* and the analytical solution u. Top: $n = 50$, bottom $n = 100$. Left: approach 1, right: approach 2.

size of the problem gets large.

Quality of the results with $\varepsilon, \hat{\varepsilon} > 0$. PB is not able to solve larger problems than $n = 50 \times 50$. However, from the above experiments we know that GR is able to solve the considered problems for $\varepsilon, \hat{\varepsilon} > 0$. Hence, we solved Example 3 with GR also for $\bar{n} = 63, 127$, in order to be able to compare the obtained results with the finite element results of [18]. The results are presented Table 2, where $e(u^*, U)$ in the last column denotes the average error between the obtained result u^* and the finite element result U of [18].

In Figure 3, the differences between the computed solution u^* and the analytical solution u are presented for Example 2 with $n = 50$ and $n = 100$. In Figure 4, the same differences are presented for Example 3 with $\bar{n} = 63$ and $\bar{n} = 127$. Finally, in Figure 5, the differences between the obtained result u^* and the finite element result U are presented for Example 3.

Both the tables and the difference plots clearly indicate that the results of approach 2 are more accurate than results of approach 1. This is partly because for approach 2, a smaller value of the smoothing parameter $\hat{\varepsilon}$ can be used. In

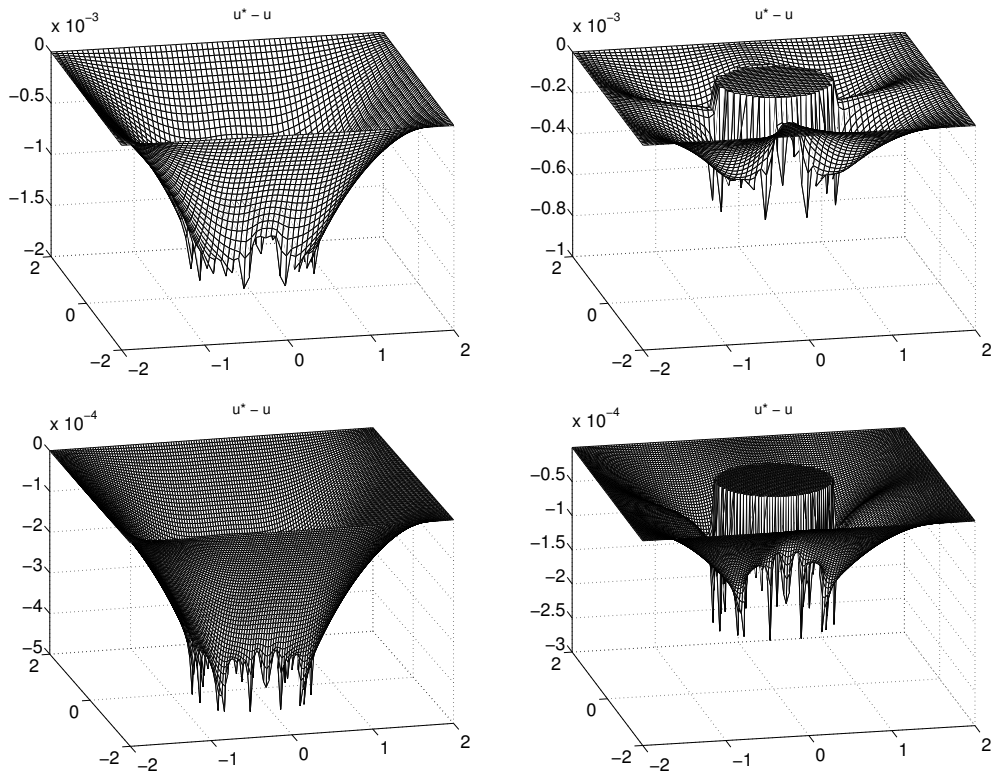

Fig. 4. Difference between the obtained result u^* and the analytical solution u. Top: $\bar{n} = 63$, bottom: $\bar{n} = 127$. Left: approach 1, right: approach 2.

fact, we observe from Table 2, that the result of approach 1 is approximately equally far away from the analytical solution and from the result of [18], whereas the result of approach 2 is clearly closer to the results of [18] than to the analytical solution.

The plots in Figure 3 clearly show the fact which is not so visible in Figure 4: The result of approach 1 is below the analytical solution at the contact region, whereas the result of approach 2 is slightly above the analytical solution at the contact region. This is due to different type of smoothing used in the two formulations. The results U of [18] are closer to the analytical solution at the contact region than the results of our approaches with $\varepsilon, \hat{\varepsilon} > 0$. This is why $u^* - U$ is plotted on the left and $U - u^*$ on the right in Figure 5.

Convergence with respect to the smoothing parameters. In Tables 3–4, the error $e(u^*, \bar{u})$ is given for different values of $\varepsilon, \hat{\varepsilon}$, and $\bar{n}$ for Example 3. Here, PB was used as a solution algorithm and we only considered the values of $\varepsilon, \hat{\varepsilon} > 0$ for which PB converged to the same solution from both $\varphi_0 = 0$ and $\varphi_0 = 1$.

For approach 1, only the values $\varepsilon = h, h^2$ were considered. The results are as can be expected; they are less accurate for larger value of ε, see Table 3.

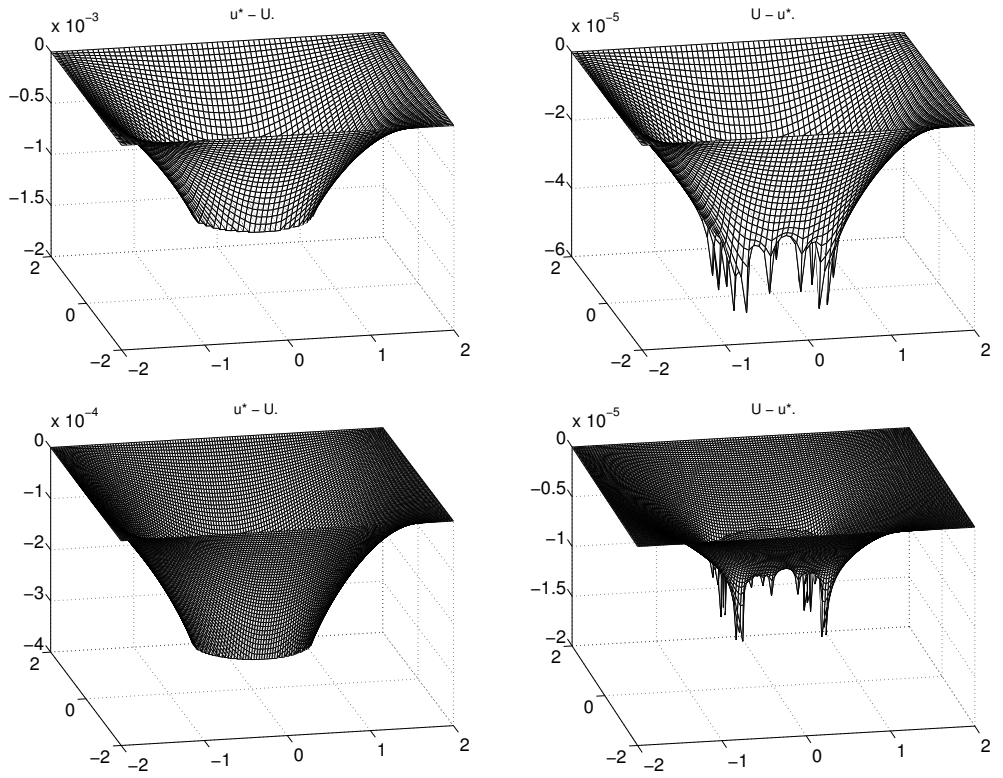

Fig. 5. Difference between the obtained result u^* and the finite element result U of [18]. Top: $\bar{n} = 63$, bottom: $\bar{n} = 127$. Left: approach 1, right: approach 2.

For approach 2, the values $\hat{\varepsilon} = h^2, h^3, h^4, h^5, h^6, 0$ were considered. The results in Table 4 show that the value of $\hat{\varepsilon}$ does not have a significant effect on the average error between the obtained result and the analytical solution. Actually, the average error seems to grow slightly when $\hat{\varepsilon}$ gets smaller. In Figure 6, the differences between the computed solution u^* and the analytical solution u are plotted for $\bar{n} = 50$ and for different values of $\hat{\varepsilon}$. It can be seen that for large value of $\hat{\varepsilon}$, the result is far away from the analytical one at the contact region but close to it outside the contact region. As $\hat{\varepsilon}$ gets smaller, the result gets closer to the analytical one at the contact region but goes further away outside the contact region.

For $\varepsilon, \hat{\varepsilon} > 0$, the results of our new approaches are always below or above the analytical solution at the contact region, due to the smoothing used in the two formulations. However, for $\varepsilon, \hat{\varepsilon} = 0$, no error is introduced and hence, the results will be exact at the contact region. In Figure 6, also a plot for $\hat{\varepsilon} = 0$ is

Table 3
Error $e(u^*, \bar{u})$ with respect to ε for approach 1.

ε /$\bar{n}$	10	30	50
h	$6.54 \cdot 10^{-2}$	$2.14 \cdot 10^{-2}$	$1.25 \cdot 10^{-2}$
h^2	$2.78 \cdot 10^{-2}$	$3.55 \cdot 10^{-3}$	$1.27 \cdot 10^{-3}$

Table 4
Error $e(u^*, \bar{u})$ with respect to $\hat{\varepsilon}$ for approach 2.

$\hat{\varepsilon}$ /$\bar{n}$	10	30	50
h^2	$4.93 \cdot 10^{-3}$	$6.92 \cdot 10^{-4}$	$2.84 \cdot 10^{-4}$
h^3	$5.62 \cdot 10^{-3}$	$6.59 \cdot 10^{-4}$	$2.68 \cdot 10^{-4}$
h^4	$6.01 \cdot 10^{-3}$	$8.83 \cdot 10^{-4}$	$3.40 \cdot 10^{-4}$
h^5	$5.82 \cdot 10^{-3}$	$9.83 \cdot 10^{-4}$	$3.63 \cdot 10^{-4}$
h^6	$6.34 \cdot 10^{-3}$	$1.02 \cdot 10^{-3}$	$3.70 \cdot 10^{-4}$
0	$6.68 \cdot 10^{-3}$	$1.01 \cdot 10^{-3}$	$3.56 \cdot 10^{-4}$

included. There is no visible error between this result and the one with $\hat{\varepsilon} = h^6$, but a closer examination shows that the result with $\hat{\varepsilon} = 0$ really is exact at the contact region.

Other conclusions from the numerical experiments Using both solution algorithms, it takes considerably more time to solve the problem of approach 2 than that of approach 1. For GR, this is due to the fact that smaller value of α must be used for the solution of the problem of approach 2. Moreover, both the algorithms become even slower when the value of the smoothing parameter $\varepsilon, \hat{\varepsilon}$ is increased. This is a surprising observation, because usually, when this kind of smoothing is used, the opposite behaviour is observed. Finally, Figure 7 illustrates how the contact region develops during the GR iterations of approach 2 for $\bar{n} = 100$.

5 Conclusion

A level set related method was proposed for solving free boundary problems coming from contact with obstacles. Two different approaches were described and applied for solving an unilateral obstacle problem. The results obtained were promising: even if the considered problems are nonsmooth and nonconvex, we obtained the same results using two different solution algorithms. Also the accuracy of the results was reasonable, taken into account that a small regularization error was introduced in the methods, in order to make the problems differentiable. In this work, we have only tested the proposed methods for solving the obstacle problem (1). The idea is applicable also for other free boundary problems dealing with contact regions.

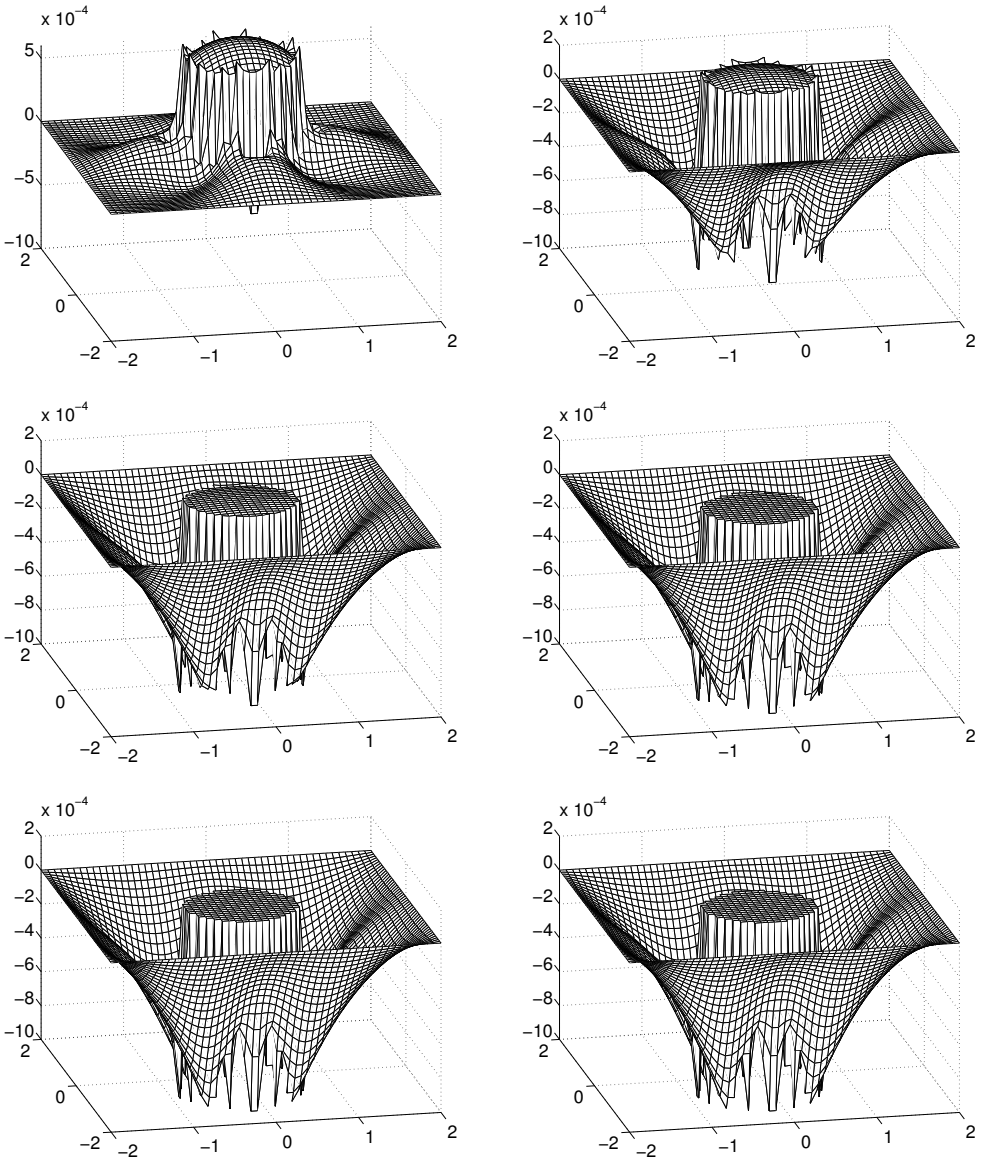

Fig. 6. Difference $(u^* - u)$ between the result of PB and the analytical solution for different values of ε: Top: left: $\varepsilon = h^2$, right: $\varepsilon = h^3$; Middle: left: $\varepsilon = h^4$, right: $\varepsilon = h^5$; Bottom: left: $\varepsilon = h^6$, right: $\varepsilon = 0$.

A Appendix

In this appendix, we give some details about the proximal bundle method. For that purpose, let us consider a general unconstrained nonlinear optimization problem

$$\min_{x \in R^n} \quad \mathcal{J}(x), \tag{A.1}$$

where $\mathcal{J} : R^n \to R$ is a locally Lipschitz continuous function. We assume that at each point $x \in R^n$, we can evaluate the function value $\mathcal{J}(x)$ and an arbitrary subgradient $g(x)$ from the subdifferential $\partial\mathcal{J}(x)$. For the convergence of the

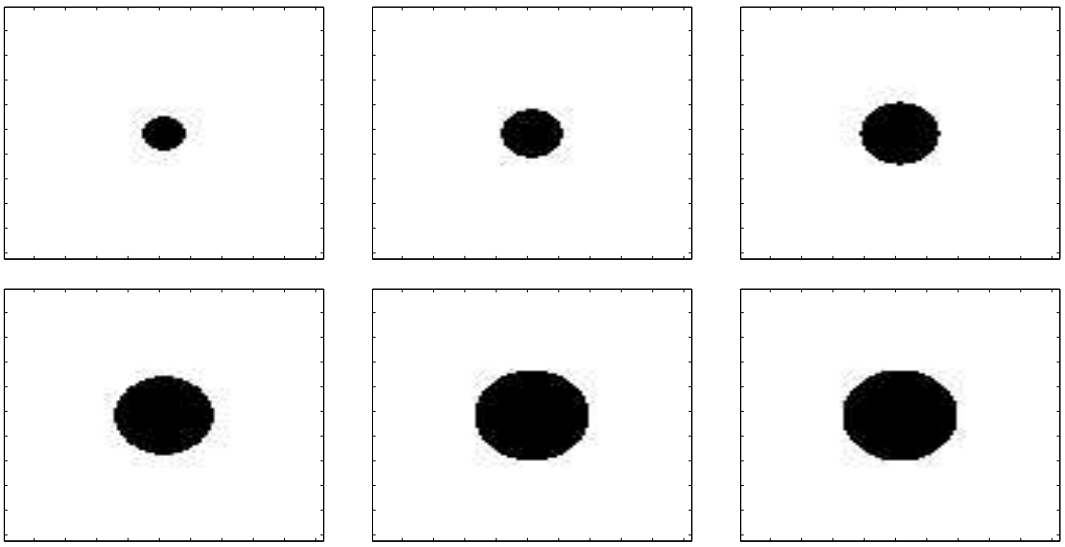

Fig. 7. The figures illustrate how the contact region develops during the iterations. Top, from left to right: $it = 20500, 21000, 22000$, bottom, from left to right: $it = 25000, 50000, 100000$.

method, in the nonconvex case, $\mathcal{J}$ is further assumed to be upper semismooth [13,14]. In what follows, we describe how the search direction is obtained in the proximal bundle method and what kind of line search it uses. For more details, see [13,15,16].

Search direction: The aim is to produce a sequence $\{x^k\}_{k=0}^{\infty} \subset R^n$ converging to some local minimum of problem (A.1). Suppose that at the k-th iteration of the algorithm, we have the current iterate x^k, some trial points $y^j \in R^n$ (from past iterations), and subgradients $g^j \in \partial\mathcal{J}(y^j)$ for $j \in G^k$, where the index set G^k is a nonempty subset of $\{0, \ldots, k\}$. The trial points y^j are used to form a local approximation model of the function $\mathcal{J}$ in a neighbourhood of the considered point x.

The idea behind the proximal bundle method is to approximate the objective function from below by a piecewise linear function. For this purpose, $\mathcal{J}$ is replaced with the so-called *cutting-plane model*

$$\hat{\mathcal{J}}_{\alpha}^{k}(x) := \max_{j \in G^k} \{\mathcal{J}(x^k) + (g^j)^T(x - x^k) - \alpha_j^k\} \tag{A.2}$$

with the *linearization error*

$$\alpha_j^k := \mathcal{J}(x^k) - \mathcal{J}(y^j) - (g^j)^T(x^k - y^j), \quad \text{for all} \quad j \in G^k. \tag{A.3}$$

Now, if $\mathcal{J}$ is convex, then the cutting-plane model $\hat{\mathcal{J}}_{\alpha}^{k}$ gives an underestimate for $\mathcal{J}$ and the nonnegative linearization error α_j^k measures how well the model

approximates the original problem. In the nonconvex case, these facts are not valid anymore: α_j^k may have a tiny (or even negative) value, even if the trial point y^j lies far away from the current iterate x^k, making the corresponding subgradient g^j useless. For these reasons, the linearization error α_j^k is replaced with the so-called *subgradient locality measure* (cf. [15])

$$\beta_j^k := \max\{|\alpha_j^k|, \gamma(s_j^k)^2\}, \tag{A.4}$$

where $\gamma \geq 0$ is the *distance measure parameter* ($\gamma = 0$ if $\mathcal{J}$ is convex), and

$$s_j^k := ||x^j - y^j|| + \sum_{i=j}^{k-1} ||x^{i+1} - x^i|| \tag{A.5}$$

is the *distance measure* estimating $||x^k - y^j||$ without the need to store the trial points y^j. Then, obviously, $\beta_j^k \geq 0$ for all $j \in G^k$ and $\min_{x \in K} \hat{\mathcal{J}}_\beta^k(x) \leq \mathcal{J}(x^k)$.

In order to calculate the search direction $d^k \in R^n$, the original problem (A.1) is replaced with the cutting plane model

$$\min_d \quad \hat{\mathcal{J}}_\beta^k(x^k + d) + \frac{1}{2}\delta^k d^T d, \tag{A.6}$$

where $\hat{\mathcal{J}}_\beta^k(x)$ denotes (A.2) with α_j^k replaced by β_j^k. In (A.6), the regularizing quadratic penalty term $\frac{1}{2}\delta^k d^T d$ is included in order to guarantee the existence of the solution d^k and keep the approximation local enough. The role of the weighting parameter $\delta^k > 0$ is to improve the convergence rate and to accumulate some second order information about the curvature of $\mathcal{J}$ around x^k. One of the most important questions concerning proximal bundle method is the choice of the weight δ^k. The simplest strategy would be to use a constant weight $\delta^k \equiv \delta^{fix}$ but this can lead to several difficulties [16]. Therefore, the weight is kept as a variable and updated when necessary. For updating δ^k, the safeguarded quadratic interpolation algorithm due to [13] is used.

Notice that problem (A.6) is still a nonsmooth optimization problem. However, due to the piecewise linear nature it can be rewritten (cf. [16, p.106] for details) as a (smooth) quadratic programming subproblem of finding the solution $(d, z) = (d^k, z^k) \in R^{n+1}$ of

$$\begin{cases} \min\limits_{d,z} & z + \frac{1}{2}\delta^k d^T d \\ \text{s. t.} & -\beta_j^k + (g^j)^T d \leq z, \quad \text{for all} \quad j \in G^k. \end{cases} \tag{A.7}$$

Line search: Let us consider the problem of determining the step size into the direction d^k calculated above. Assume that $m_L \in (0, \frac{1}{2}), m_R \in (m_L, 1)$,

and $\bar{t} \in (0,1]$ are fixed line search parameters. First, we shall search for the largest number $t_L^k \in [0,1]$ such that $t_L^k \geq \bar{t}$ and

$$\mathcal{J}(x^k + t_L^k d^k) \leq \mathcal{J}(x^k) + m_L t_L^k z^k, \tag{A.8}$$

where z^k is the predicted amount of descent. If such a parameter exists, we take a *long serious step*

$$x^{k+1} := x^k + t_L^k d^k \quad \text{and} \quad y^{k+1} := x^{k+1}.$$

The long serious step yields a significant decrease in the value of the objective function. Thus, there is no need for detecting discontinuities in the gradient of $\mathcal{J}$, so that we set $g^{k+1} \in \partial\mathcal{J}(x^{k+1})$.

Otherwise, if (A.8) holds but $0 < t_L^k < \bar{t}$, then a *short serious step*

$$x^{k+1} := x^k + t_L^k d^k \quad \text{and} \quad y^{k+1} := x^k + t_R^k d^k$$

is taken. Finally, if $t_L^k = 0$, then we take a *null step*

$$x^{k+1} := x^k \quad \text{and} \quad y^{k+1} := x^k + t_R^k d^k,$$

where $t_R^k > t_L^k$ is such that

$$-\beta_{k+1}^{k+1} + (g^{k+1})^T d^k \geq m_R z^k. \tag{A.9}$$

In the short serious step and null step, there exists discontinuity in the gradient of $\mathcal{J}$. Then, the requirement (A.9) ensures that x^k and y^{k+1} lie on the opposite sides of this discontinuity, and the new subgradient $g^{k+1} \in \partial\mathcal{J}(y^{k+1})$ will force a remarkable modification of the next problem of finding the search direction. In PB, the line search algorithm presented in [16] is used.

In practice, the iteration is terminated if

$$z^k \geq -tol_{PB},$$

where $tol_{PB} > 0$ is the final accuracy tolerance supplied by the user. However, there are also some other, more complicated, stopping criteria involved [19].

In the numerical experiments, we chose the following values for the line search parameters: $m_L = 0.01, m_R = 0.5$, and $\bar{t} = 0.1$, and for the distance measure parameter, we chose $\gamma = 0.01$. As a stopping criterion in 1D, we used $tol_{PB} = 10^{-5}$. In 2D, different stopping criteria were used for different problem sizes to obtain the desired accuracy, namely, $tol_{PB} = 10^{-5}$ for $\bar{n} = 10$, $tol_{PB} = 10^{-6}$ for $\bar{n} = 30$, and $tol_{PB} = 10^{-7}$ for $\bar{n} = 50$.

Acknowledgements

The authors want to thank Dr. Marko Mäkelä at the University of Jyväskylä for the possibility to use the PB code in the numerical experiments.

References

[1] S. Osher, J. A. Sethian, Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.

[2] S. Osher, R. Fedkiw, Level set methods and dynamic implicit surfaces, Vol. 153 of Applied Mathematical Sciences, Springer-Verlag, New York, 2003.

[3] S. Osher, R. P. Fedkiw, Level set methods: an overview and some recent results, J. Comput. Phys. 169 (2) (2001) 463–502.

[4] J. A. Sethian, Level set methods and fast marching methods, 2nd Edition, Vol. 3 of Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 1999, evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science.

[5] T. F. Chan, L. A. Vese, Image segmentation using level sets and the piecewise constant mumford-shah model, Tech. rep., CAM Report 00-14, UCLA, Math. Depart., revised December 2000 (April 2000).

[6] T. F. Chan, X.-C. Tai, Level set and total variation regularization for elliptic inverse problems with discontinuous coefficients, J. Comput. Phys. (2003) to appear.

[7] S. Osher, F. Santosa, Level set methods for optimization problems involving geometry and constraints. I. Frequencies of a two-density inhomogeneous drum, J. Comput. Phys. 171 (1) (2001) 272–288.

[8] F. Santosa, A level-set approach for inverse problems involving obstacles, ESAIM Contrôle Optim. Calc. Var. 1 (1995/96) 17–33 (electronic).

[9] J. A. Sethian, J. Strain, Crystal growth and dendritic solidification, J. Comput. Phys. 98 (2) (1992) 231–253.

[10] R. Glowinski, Numerical methods for nonlinear variational problems, Springer-Verlag, 1984.

[11] F. H. Clarke, Optimization and Nonsmooth Analysis, John Wiley & Sons, New York, 1983.

[12] J. E. Kelley, The cutting plane method for solving convex programs, Journal of the SIAM 8 (1960) 703–712.

[13] K. C. Kiwiel, Proximity control in bundle methods for convex nondifferentiable optimization, Math. Program. 46 (1990) 105–122.

[14] H. Schramm, J. Zowe, A version of the bundle idea for minimizing a nonsmooth function: conceptual idea, convergence analysis, numerical results, SIAM J. Optim. 2 (1) (1992) 121–152.

[15] K. C. Kiwiel, Methods of Descent for Nondifferentiable Optimization, Lecture Notes in Mathematics 1133, Springer-Verlag, Berlin-Heidelberg, 1985.

[16] M. M. Mäkelä, P. Neittaanmäki, Nonsmooth Optimization. Analysis and Algorithms with Applications to Optimal Control, World Scientific, Singapore, 1992.

[17] K. C. Kiwiel, A method for solving certain quadratic programming problems arising in nonsmooth optimization, IMA J. Numer. Anal. 6 (1986) 137–152.

[18] X.-C. Tai, Rate of convergence for some constraint decomposition methods for nonlinear variational inequalities, Numer. Math. 93 (4) (2003) 755–786.

[19] M. M. Mäkelä, T. Männikkö, Numerical solution of nonsmooth optimal control problems with an application to the continuous casting process, Adv. Math. Sci. Appl. 4 (2) (1994) 491–515.