Available online at www.isr-publications.com/jnsa
J. Nonlinear Sci. Appl., 11 (2018), 1302–1312
Research Article

ISSN : 2008-1898

Journal of Nonlinear Sciences and Applications

Journal Homepage: www.isr-publications.com/jnsa

Numerical solution for a nonlinear obstacle problem

Ling Rao[a,*], Shih-Sen Chang[b]

[a]Department of Mathematics, Nanjing University of Science and Technology, Nanjing, China.
[b]Center for General Educatin, China Medical University, Taichung, 40402, Taiwan.

Communicated by R. Saadati

Abstract

A monotone iterations algorithm combined with the finite difference method is constructed for an obstacle problem with semilinear elliptic partial differential equations of second order. By means of Dirac delta function to improve the computation procedure of the discretization, the finite difference method is still practicable even though the obstacle boundary is irregular. The numerical simulations show that our proposed methods are feasible and effective for the nonlinear obstacle problem.

Keywords: Finite difference method, nonlinear obstacle problem, variational inequality, semilinear elliptic partial differential equation.

2010 MSC: 35J87, 35J61.

1. Introduction

The obstacle problems are frequently appeared in many engineering problems [4], such as the Stefan problem, the filtration dam problem, the subsonic flow problem, etc. They are extremely difficult to be solved. Hence, it is necessary to develop a simple and accurate numerical scheme to deal with the obstacle problems for engineering applications.

During the past decades, there were some numerical schemes adopted for solving the obstacle problems. In [6], Korman et al. proved existence, uniqueness and regularity of solutions for the obstacle problem with semilinear elliptic equations of second order. And computationally effective algorithms were provided by constructing a monotone sequence of iterations which converges to a solution. We can see that at each step of iterations one solves the obstacle problem with a linear elliptic equation. Effectively solving the adjoint linear elliptic equation at each step plays an important role when using this method.

In this paper we provide an alternative algorithm based on monotone iterations combined with the finite difference method for solving the nonlinear obstacle problem. We use the Lagrange multiplier-based fictitious domain method [5, 10] for solving the adjoint linear elliptic equation. The fictitious

*Corresponding author
Email addresses: lingrao@sina.com.cn (Ling Rao), changss2013@163.com (Shih-Sen Chang)

doi: 10.22436/jnsa.011.12.02

Received: 2018-03-14 *Revised:* 2018-08-05 *Accepted:* 2018-08-28

domain methods extend the governing equations to a simple domain (such as a box) and the boundary conditions along the body are enforced by introducing suitable Lagrange multipliers. Conventionally the involving linear elliptic variational problems are solved by the finite element method. The mesh of the extended domain is constructed with a rectangular triangulated mesh by locally fitting this mesh to the irregular obstacle boundary. But we will meet the trouble of computing the boundary integrals due to obstacle boundary unfitted structured grid. There is an increased interest in solution algorithms for non-body-conforming grids.

Our method based on finite difference method to solve the extended problem, see [9]. The basic idea is originated from the immersed boundary method, see [7, 8, 11]. By means of a Dirac delta function we transfer the variational forms of the linear elliptic equations to strong forms. We can still apply finite difference method to solve the strong forms in the extended simple domain, even though the obstacle boundary is irregular. Comparing with the conventional finite element method, our computation procedure of the finite difference discretizations need less computational effort and memory requirement due to using Dirac function.

The paper is organized as follows. In Section 2, we describe a nonlinear obstacle problem and its monotone iteration algorithm. In Section 3, the algorithm based on fictitious domain method is provided for solving the adjoint linear elliptic equation. We use Dirac delta function to improve the computation procedure of the discretization. In Section 4, we do numerical experiments to show that our proposed methods are feasible and effective.

2. Nonlinear obstacle problem and monotone iterations

Let L denote the operator

$$Lu \equiv -\sum_{i,j=1}^{n} a_{i,j}(x)\frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=1}^{n} b_i(x)\frac{\partial u}{\partial x_i} + c(x)u \quad \text{in } \Omega,$$

where Ω is a bounded domain in R^n. We consider the following nonlinear obstacle problem

$$\left.\begin{aligned} Lu &\leqslant f(x,u) \\ u &\leqslant q(x) \\ (Lu - f(x,u))(u-q) &= 0 \end{aligned}\right\} \quad \text{a.e. in } \Omega, \qquad u = g(x) \text{ on } \partial\Omega. \tag{2.1}$$

Assume that L is a uniformly elliptic operator with coefficients in $C^{\alpha}(\overline{\Omega})$, $q(x) \in C^2(\Omega)$, $g(x) \in C^{2+\alpha}(\Omega)$, $g(x) < q(x)$ on $\partial\Omega$, $\partial\Omega \in C^{2+\alpha}(\Omega)$. In this paper, we first apply a monotone iterations method to solve (2.1). The method is described in the following Lemma (see [6]).

Lemma 2.1. *In addition to the conditions above, assume that there exists $\varphi \in C^2(\overline{\Omega})$ (a subsolution) such that*

$$L\varphi \leqslant f(x,\varphi),\ \varphi(x) \leqslant q(x) \text{ in } \Omega, \qquad \varphi(x) \leqslant q(x) \text{ on } \partial\Omega,$$

and that $f(x,u)$ is C^{α} in x and uniformly Lipschitz in u, for $x \in \overline{\Omega}$ and $\varphi(x) \leqslant u \leqslant q(x)$. Then the problem (2.1) has a solution $u(x) \in H^{2,p}(\Omega) \bigcap H^{2,\infty}_{loc}(\Omega)$ for any p. Start with $u^0 = q(x)$, and find a sequence of functions $\{u^n(x)\}$ by solving the following variational inequalities $(n = 0, 1, \cdots)$

$$\left.\begin{aligned} Lu^{n+1} &\leqslant f(x,u^n) \\ u^{n+1} &\leqslant q(x) \\ (Lu^{n+1} - f(x,u^n))(u^{n+1}-q) &= 0 \end{aligned}\right\} \quad \text{a.e. in } \Omega, \qquad u^{n+1} = g(x) \quad \text{on } \partial\Omega. \tag{2.2}$$

Then the sequence $\{u^n(x)\}$ *is monotone decreasing*

$$u^0 \geqslant u^1 \geqslant u^2 \geqslant \cdots,$$

and converges to the maximal solution $u(x)$ *of* (2.1). *Moreover* $u(x) \geqslant \varphi(x)$.

It is shown in Lemma 2.1 that at each step of iterations for solving (2.1) one need to solve the linear obstacle problem (2.2). We solve (2.2) by a dual method [4] as below.

For each given u^n, find $\{v^m(x)\}$ by

$$\lambda^0 \in L^2_+(\Omega) \text{ chosen arbitrarily (e.g. zero)},$$

$$\begin{cases} Lv^{m+1} = f(x, u^n) + \lambda^m & \text{in } \Omega, \\ v^{m+1} = 0 & \text{on } \partial\Omega, \end{cases} \tag{2.3}$$

$$\lambda^{m+1} = (\lambda^m - \rho(q - v^{m+1}))^+, \tag{2.4}$$

where ρ is a properly chosen constant (see e.g. [4]). Then the limit of $\{v^m\}$ is u^{n+1}.

The above method implies that solving (2.1) ultimately comes down to solving a series of linear elliptic equations (2.3). If Ω is a simple domain (such as a box), then we can construct a rectangular triangulated mesh on it and directly apply the finite difference method to solve (2.3). The aim of this paper is to study how to let the finite difference method be still useful for solving equations (2.3) when the shape of Ω is irregular. We describe the algorithm presented by [9] in the next section.

3. Fictitious domain based approach

Consider the general forms of linear elliptic equations (2.3)

$$\begin{cases} Lu = f, & \text{in } \Omega, \\ u = g_0, & \text{on } \gamma, \end{cases} \tag{3.1}$$

where γ is the boundary of the domain Ω, $g_0 \in H^{\frac{1}{2}}(\gamma)$, $f \in L^2(\Omega)$.

We solve the problem in the extended rectangular domain $\hat{\Omega}$: $\overline{\hat{\Omega}} = \overline{B} \cup \overline{\Omega}$, $\partial\overline{\hat{\Omega}} = \Gamma$, see Figure 1. Arbitrarily give $g_1 \in H^{\frac{1}{2}}(\Gamma)$, (e.g. zero). According to the boundary Lagrangian fictitious domain method [5], the Dirichlet boundary condition on γ is enforced by introducing Lagrangian multiplier on the boundary. Problem (3.1) is equivalent to the following variational problem:

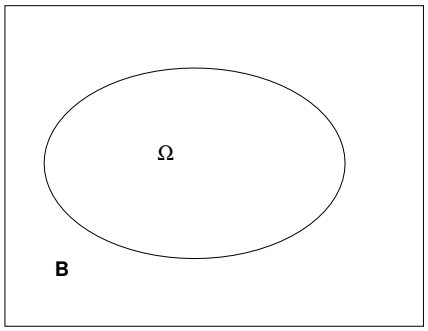


Figure 1: The extended domain of obstacle Ω.

Find $\overline{u} \in \overline{V_g}$, $\lambda \in H^{-\frac{1}{2}}(\gamma)$, such that

$$a_{\hat{\Omega}}(\overline{u}, z) = \int_{\hat{\Omega}} \overline{f} z \, dx + \int_{\gamma} \lambda z \, d\gamma, \quad \forall z \in \overline{V_0}, \tag{3.2}$$

$$\int_\gamma \mu(\overline{u}-g_0)d\gamma = 0, \quad \forall \mu \in H^{-\frac{1}{2}}(\gamma), \tag{3.3}$$

where $\overline{u}$ and $\overline{f}$ are the extensions of u and f in $\hat{\Omega}$, respectively, and $\overline{u}|_\Omega = u$, $\overline{f}|_\Omega = f$,

$$\overline{V}_g = \{z | z \in H^1(\hat{\Omega}),\ z = g_1 \text{ on } \Gamma\},$$

$$\overline{V_0} = \{z | z \in H^1(\hat{\Omega}),\ z = 0 \text{ on } \Gamma\},$$

$$a_{\hat{\Omega}}(\overline{u}, z) \equiv \int_{\hat{\Omega}} \left[\sum_{i,j=1}^{n} a_{i,j}(x) \frac{\partial \overline{u}}{\partial x_i} \frac{\partial z}{\partial x_j} + \sum_{i=1}^{n} [b_i(x) + \sum_{j=1}^{n} \frac{\partial a_{ij}}{\partial x_j}] \frac{\partial \overline{u}}{\partial x_i} z + c(x)\overline{u}z \right] dx, \quad \forall z \in \overline{V_0}.$$

Using Green formula, we have

$$(Lu,\ z) = a_{\hat{\Omega}}(\overline{u}, z) - (\frac{\partial u}{\partial \nu},\ z)_{L_2(\partial \hat{\Omega}),}$$

where $u \in H^2(\hat{\Omega})$, $z \in H^2(\hat{\Omega})$, $\frac{\partial u}{\partial \nu} = \sum_{i,j=1}^{n} a_{i,j}(x) \cos(n,\ x_i) \frac{\partial u}{\partial x_j}$.

We use Generalized Conjugate Gradient (GCG) [9] to solve problems (3.2)–(3.3). The algorithm is presented as follows.

Algorithm 3.1.

Step 0: Initialization.

–Set initial Lagrangian multipliers: $\lambda^0 \in L^2(\gamma)$ and a number $\varepsilon \geqslant 0$ small enough for the convergence criterion.

– Find $\overline{u}^0 \in \overline{V}_g$ by

$$a_{\hat{\Omega}}(\overline{u}^0, z) = \int_{\hat{\Omega}} \overline{f} z d\mathbf{x} + \int_\gamma \lambda^0 z d\gamma, \quad \forall z \in \overline{V}_0. \tag{3.4}$$

–Calculate $g^0 \in L^2(\gamma)$ by

$$\int_\gamma g^0 \mu d\gamma = \int_\gamma \mu(\overline{u}^0 - g_0)d\gamma, \quad \forall \mu \in L^2(\gamma).$$

–Set the initial descent direction $w^0 = g^0$.

To obtain λ^{n+1}, $\overline{u}^{n+1}$, g^{n+1} and w^{n+1} from λ^n, $\overline{u}^n$, g^n and w^n, one proceeds as follows.

Step 1: Find descent direction.

– Solve $\widetilde{\overline{u}}^n \in \overline{V}_0$ by

$$a_{\hat{\Omega}}(\widetilde{\overline{u}}^n, z) = \int_\gamma w^n z d\gamma, \quad \forall z \in \overline{V}_0. \tag{3.5}$$

– Calculate ρ_n by

$$\rho_n = \int_\gamma |g^n|^2 d\gamma / \int_\gamma \widetilde{\overline{u}}^n w^n d\gamma. \tag{3.6}$$

– Find the new solution by

$$\overline{u}^{n+1} = \overline{u}^n - \rho_n \widetilde{\overline{u}}^n,$$
$$\lambda^{n+1} = \lambda^n - \rho_n w^n.$$

– Calculate the new gradient $g^{n+1} \in L^2(\gamma)$ by

$$\int_\gamma g^{n+1} \mu d\gamma = \int_\gamma g^n \mu d\gamma - \rho_n \int_\gamma \widetilde{\overline{u}}^n \mu d\gamma, \quad \forall \mu \in L^2(\gamma).$$

Step 2: Construct convergence criterion and update descent direction.

– If $\int_\gamma |g^{n+1}|^2 d\gamma / \int_\gamma |g^0|^2 d\gamma \leqslant \varepsilon$, then take the solution being $\lambda = \lambda^{n+1}$, $\overline{u} = \overline{u}^{n+1}$. Otherwise

$$\gamma_n = \int_\gamma |g^{n+1}|^2 d\gamma / \int_\gamma |g^n|^2 d\gamma,$$

$$w^{n+1} = g^{n+1} + \gamma_n w^n.$$

Set $n = n+1$ and return to **Step 1**.

It can be seen from the above algorithm that we need to calculate elliptic variational problems (3.4) and (3.5). Conventionally we solve them by the finite element method. In the computation procedure of the finite element discretizations, the mesh of the extended domain is constructed from a rectangular triangulated mesh by locally fitting this mesh to the irregular obstacle boundary. The conventional finite element discretizations result the problem in the solution of huge algebraic system of equations and we will meet the trouble of computing the boundary integrals. In order to avoid these difficulties and solve the extended problem more efficiently, we use Dirac delta function to improve the computation procedure of the discretizations. We describe the method presented by [9] as follows.

We construct a regular Eulerian mesh on $\hat{\Omega}$ by

$$\hat{\Omega}_k = \{x_{ij} \mid x_{ij} = (x_0 + ih, y_0 + jh),\ 0 \leqslant i, j \leqslant I\},$$

where h is the mesh width (for convenience, kept the same both in x- and in y-directions). Assume that the configuration of the simple closed curve γ is given in a parametric form $X(s)$, $0 \leqslant s \leqslant L$. The discretization of the boundary γ employs a Lagrangian mesh, represented as a finite collection of Lagrangian points X_k, $0 \leqslant k \leqslant J$, apart from each other by a distance Δs, usually taken as being $h/2$.

Let $\delta(\cdot)$ be a Dirac delta function. In the following calculation procedure, δ is approximated by the distribution function δ_h. The choice here is given by the product

$$\delta_h(\mathbf{x}) = d_h(x) d_h(y),$$

where

$$d_h(z) = \begin{cases} \dfrac{0.25}{h}\left[1 + \cos\left(\dfrac{\pi z}{2h}\right)\right], & |z| \leqslant c_h, \\ 0, & |z| > c_h, \end{cases} \tag{3.7}$$

where c_h is a constant relevant to h and may be properly chosen. For example, $c_h = 2h$ (see [3]). Using the above Dirac delta function we can transfer the weak forms of the partial differential equations (3.4) and (3.5) to the strong forms and then solve them by linear elliptic equation solvers such as SOR Solver or Fast Poisson Solvers. In mathematical view, we need the following Lemma (see [1]).

Lemma 3.2. *Assume that the simple closed curve γ, the configuration of which is given in a parametric form $X(s)$, $0 \leqslant s \leqslant L$ is Lipschitz continuous, $f \in L^2(0, L)$. Then F defined by*

$$F(x) = \int_0^L f(s)\delta(x - X(s))ds, \qquad \forall x \in \Omega,$$

is a distribution function belonging to $H^{-1}(\Omega)$ defined as follows: for all $v \in H_0^1(\Omega)$

$$_{H^{-1}(\Omega)}\langle F, v\rangle_{H_0^1(\Omega)} = \int_0^L f(s)v(X(s))ds.$$

By Lemma 3.2, we can rewrite the right hand in (3.5) as following form

$$\int_\gamma \omega^n z d\gamma = {}_{H^{-1}(\hat{\Omega})}\langle W^n, z\rangle_{H_0^1(\hat{\Omega})},$$

where

$$W^n(\mathbf{x}) = \int_0^L \omega^n(s)\delta(\mathbf{x} - X(s))ds, \quad \forall \mathbf{x} \in \hat{\Omega}.$$

That is, ω^n calculated on the Lagrangian points are distributed on the Eulerian nodes. Thus we can rewrite (3.5) in the strong form as below

$$L\tilde{\bar{u}}^n = W^n, \quad \text{in } \hat{\Omega}.$$

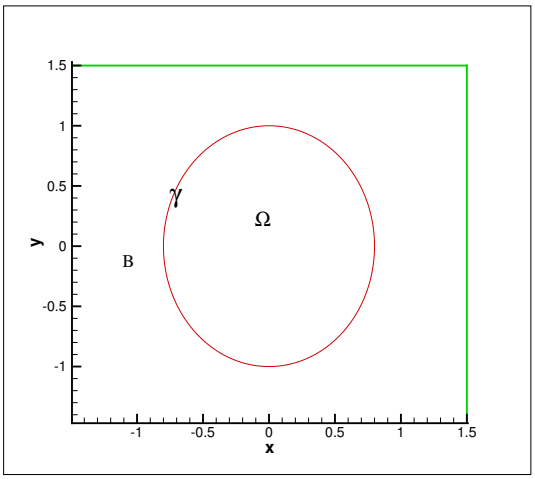
In the same way, (3.4) also can be rewritten in the strong form as below

$$L\bar{u}_0 = \bar{f} + R^0, \quad \text{in } \hat{\Omega},$$

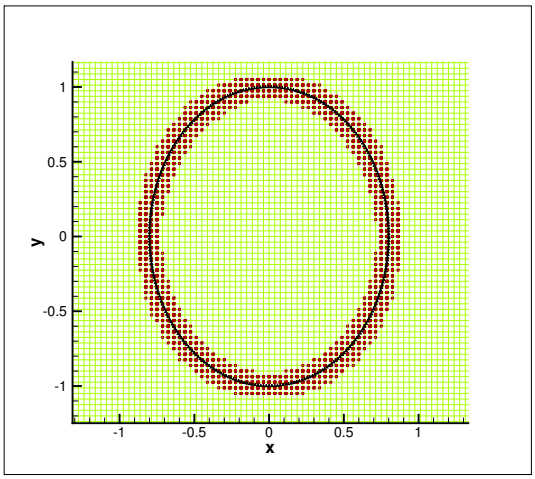
where

$$R^0(\mathbf{x}) = \int_0^L \lambda^0(s)\delta(\mathbf{x} - X(s))ds, \quad \forall \mathbf{x} \in \hat{\Omega}.$$

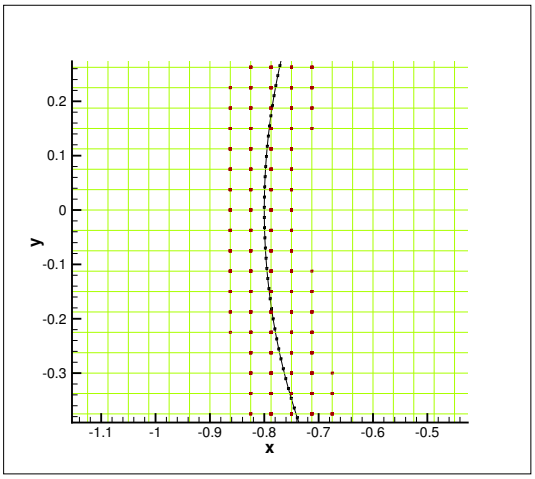
Then we have the discrete algorithm of Algorithm 3.1 for solving (3.2)–(3.3) as follows.



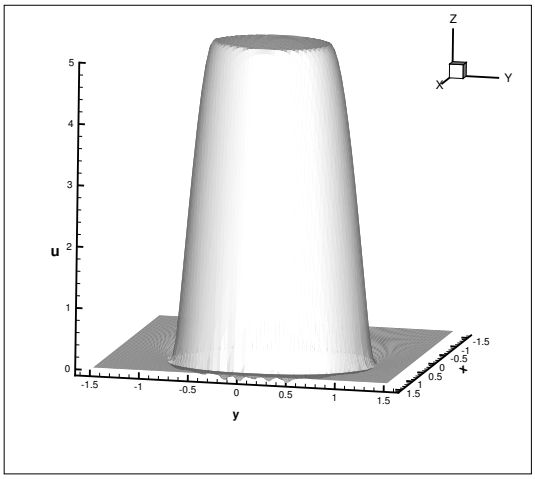
(a) Computational domain of the elliptic obstacle Ω.



(b) Zoom view of Eulerian mesh, Lagrangian mesh and the used Eulerian nodes near the boundary γ.



(c) Local view of Figure(b).



(d) The distribution of the numerical solution.

Figure 2: Solution with the elliptic obstacle Ω and $c_h = 2h$.

Algorithm 3.3.

Step 0: Initialization.

1. Set initial Lagrangian multipliers: $\lambda^0 \in L^2(\gamma)$. Distributed λ^0 on the Lagrangian points $\{X_k\}$ to the Eulerian nodes $\{\mathbf{x}_{ij}\}$ by

$$R^0(\mathbf{x}_{ij}) = \sum_k \lambda_k^0 \delta_h(\mathbf{x}_{ij} - X_k)\Delta s, \quad \forall\, \mathbf{x}_{ij} \in \hat{\Omega}_h. \tag{3.8}$$

2. Find $\overline{u}^0 \in \overline{V_g}$ satisfying

$$L\overline{u}^0 = \overline{f} + R^0, \quad \text{in } \hat{\Omega}, \tag{3.9}$$

where

$$R^0(\mathbf{x}) = \int_0^L \lambda^0(s)\delta(\mathbf{x} - X(s))ds, \quad \forall \mathbf{x} \in \hat{\Omega}. \tag{3.10}$$

The discrete form (3.10) is (3.8).

3. Calculate $\overline{u}^0$ on Lagrangian points with $\overline{u}^0$ on neighboring Eulerian nodes by

$$\overline{u}_k^0 = \sum_{ij} \overline{u}_{ij}^0 \delta_h(\mathbf{x}_{ij} - X_k)h^2, \quad \forall 1 \leqslant k \leqslant N. \tag{3.11}$$

4. Calculate g^0 on Lagrangian points by

$$g_k^0 = \overline{u}_k^0 - g_{0k}, \quad 1 \leqslant k \leqslant N.$$

5. Set the initial descent direction $w_k^0 = g_k^0$, $1 \leqslant n \leqslant N$.

To obtain λ^{n+1}, $\overline{u}^{n+1}$, g^{n+1} and w^{n+1} from λ^n, $\overline{u}^n$, g^n and w^n, one proceeds as follows.

Step 1: Find descent direction.

1. Distributed ω^n on the Lagrangian points to the Eulerian points by

$$W^n(\mathbf{x}) = \int_0^L \omega^n(s)\delta(\mathbf{x} - X(s))ds, \quad \forall \mathbf{x} \in B,$$

which discrete form is

$$W^n(\mathbf{x}_{ij}) = \sum_k \omega_k^n \delta_h(\mathbf{x}_{ij} - X_k)\Delta s, \quad \forall \mathbf{x}_{ij} \in B_h. \tag{3.12}$$

2. Find $\widetilde{\overline{u}}^n \in \overline{V_0}$ by solving:

$$L\widetilde{\overline{u}}^n = W^n, \quad \text{in } \hat{\Omega}. \tag{3.13}$$

3. Calculate $\widetilde{\overline{u}}^n$ on Lagrangian points with $\widetilde{\overline{u}}^n$ on neighboring Eulerian nodes by

$$\widetilde{\overline{u}}^n(X(s)) = \int_{\hat{\Omega}} \widetilde{\overline{u}}^n(\mathbf{x})\delta(\mathbf{x} - X(s))d\mathbf{x}, \tag{3.14}$$

which discrete form is

$$\widetilde{\overline{u}}_k^n = \sum_{ij} \widetilde{\overline{u}}_{ij}^n \delta_h(\mathbf{x}_{ij} - X_k)h^2, \quad \forall 1 \leqslant k \leqslant N. \tag{3.15}$$

4. Calculate ρ_n by

$$\rho_n = \int_\gamma |g^n|^2 d\gamma \Big/ \int_\gamma \widetilde{\overline{u}}^n \omega^n d\gamma,$$

which discrete form is

$$\rho_n = \frac{\sum_k |g_k^n|^2 \Delta s}{\sum_k \widetilde{\overline{u}}_k^n w_k^n \Delta s}.$$

5. Let

$$\lambda_k^{n+1} = \lambda_k^n - \rho_n w_k^n, \quad 1 \leqslant n \leqslant N.$$

6. Calculate the new gradient g^{n+1} by

$$g_k^{n+1} = g_k^n - \rho^n \bar{\tilde{u}}_k^n, \quad 1 \leqslant k \leqslant N.$$

Step 2: Construct convergence criterion and update descent direction. For given $\varepsilon \geqslant 0$ small enough, if

$$\int_\gamma |g^{n+1}|^2 d\gamma \Big/ \int_\gamma |g^0|^2 d\gamma \leqslant \epsilon,$$

that is,

$$\sum_k |g_k^{n+1}|^2 \Delta s \Big/ \sum_k |g_k^0|^2 \Delta s \leqslant \varepsilon,$$

then take $\lambda = \lambda^{n+1}$ on the Lagrangian points, and find $\bar{u} \in \overline{V_g}$ by

$$L\bar{u} = \bar{f} + R, \quad \text{in } \hat{\Omega}, \tag{3.16}$$

where

$$R(\mathbf{x}) = \int_0^L \lambda(s)\delta(\mathbf{x} - X(s))ds, \quad \forall \mathbf{x} \in \hat{\Omega},$$

which discrete form is

$$R(\mathbf{x}_{ij}) = \sum_k \lambda_k \delta_h(\mathbf{x}_{ij} - X_k)\Delta s, \quad \forall x_{ij} \in \hat{\Omega}_h. \tag{3.17}$$

Take $\bar{u}$ as the numerical solution of (3.1).

Otherwise, let

$$\gamma_n = \sum_k |g_k^{n+1}|^2 \Big/ \sum_k |g_k^n|^2,$$

$$w_k^{n+1} = g_k^{n+1} + \gamma_n w_k^n, \quad 1 \leqslant k \leqslant N.$$

Set $n = n + 1$ and return to **Step 1**.

Note that (3.9), (3.13) and (3.16) are defined in the rectangular domain $\hat{\Omega}$. We can solve their discrete forms by known linear elliptic equation Solvers based on the finite difference method. So the proposed method need not make use of the finite element method. The algorithm has a simple structure. By fast linear elliptic equation Solvers such as the fast Fourier transform or cyclic reduction we can increase speed of calculation.

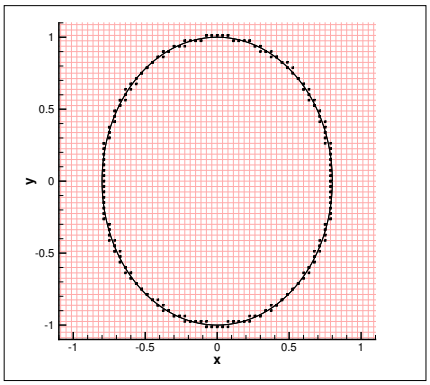
It can be seen from steps 4, 5, and 6 in **Step 1** that the calculations are done on the Lagrangian points. And (3.8), (3.11) and (3.17) show that δ_h plays a key role in Algorithm 3.3. By (3.7), if $|\delta_h(\mathbf{x}_{ij} - X_k)| > c_h$, $\delta_h(\mathbf{x}_{ij} - X_k) = 0$, then only those Eulerian nodes near γ need to be used for calculation and magnitude of c_h affects the precision of the numerical solution. Such approaches can lead to a significant reduction in computational effort and memory requirement.

4. Numerical experiments

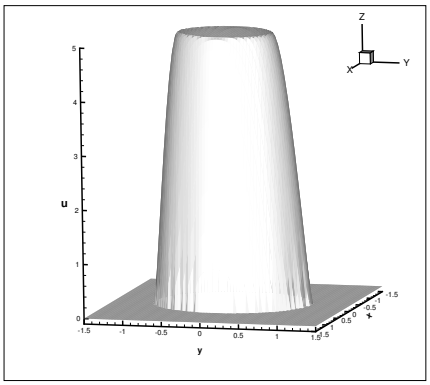
In this section, numerical examples are provided to validate the above proposed schemes for two dimensional obstacle problems with variable coefficients.

Consider two dimensional problem (2.1). Take $q(x, y) = 5$, $g(x, y) = 0$, $f(u) = -u^2 + 30u$, and

$$Lu = -(1 + y^2x^2)\frac{\partial^2 u}{\partial x^2} - (1 + \frac{y}{2})\frac{\partial^2 u}{\partial y^2} + 15u.$$

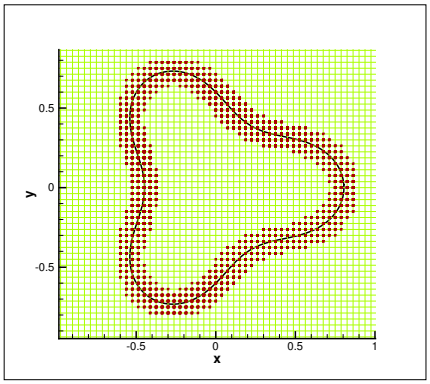


(a) Zoom view of Eulerian mesh, Lagrangian mesh and the used Eulerian nodes near the boundary γ.

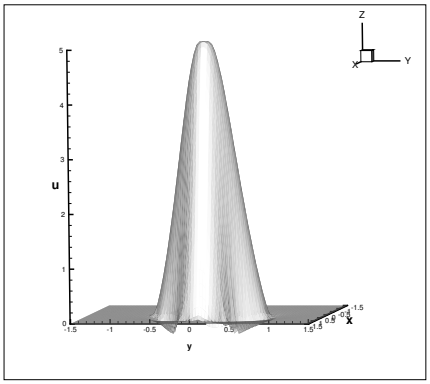


(b) The distribution of the numerical solution.

Figure 3: Solution with the elliptic obstacle Ω and $c_h = 0.5h$.

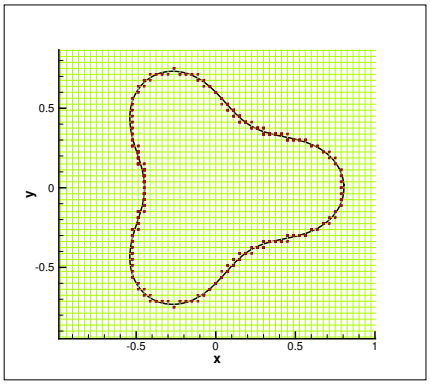


(a) Zoom view of Eulerian mesh, Lagrangian mesh and the used Eulerian nodes near the boundary γ.

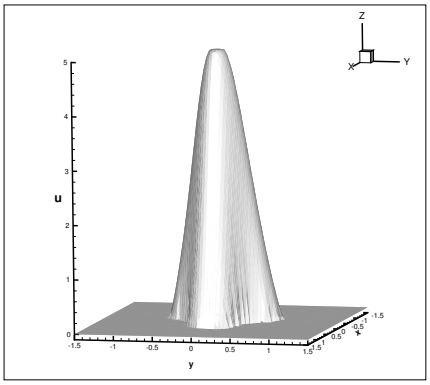


(b) The distribution of the numerical solution.

Figure 4: Solution with the irregular obstacle and $c_h = 2h$.



(a) Zoom view of Eulerian mesh, Lagrangian mesh and the used Eulerian nodes near the boundary γ.



(b) The distribution of the numerical solution.

Figure 5: Solution with the irregular obstacle and $c_h = 0.5h$.

We apply Algorithm 3.3 to monotone iteration equations (2.3) and (2.4) in Section 2, taking $\rho = 10$ in (2.4). Linear elliptic equations (3.9) and (3.13) are solved by SOR Solver. We take the fictitious domain $\overline{\hat{\Omega}} = [-1.5, 1.5] \times [-1.5, 1.5]$. Rectangular Eulerian mesh has 80×80 nodes. Ω is given different irregular shape and the constant c_h in (3.7) is properly chosen different value. We do four numerical tests as below.

Test 1. Take $c_h = 2h$, , h is the length of Eulerian grids. Ω shown in Figure 2 (a) is an elliptic obstacle. Results are shown in Figure 2 where (b) and (c) show zoom view of Eulerian mesh, Lagrangian mesh and the used Eulerian nodes near the boundary γ of Ω for calculating the solution. Figure 2 (d) is the distribution of numerical solution.

Test 2. Take $c_h = 0.5h$. Ω is the same elliptic obstacle as in **Test 1**. Results are shown in Figure 3. Figure 3 (a) shows zoom view of Eulerian mesh, Lagrangian mesh and the used Eulerian nodes near the boundary γ of Ω for calculating the solution. We see the distribution of the used Eulerian nodes near boundary γ for calculating the solution is different from that in **Test 1** due to different c_h chosen. And the value of the numerical solution u on the boundary γ in Figure 3 (b) is equal to 0, satisfying boundary condition in (2.1): $u(x, y) = g = 0$ on $\partial\Omega$. But in Figure 2 (d), u on γ has error.

Test 3. Take $c_h = 2h$. Ω is an irregular obstacle shown in Figure 4 (a). Results are shown in Figure 4. We observe that the value of u on γ has error as in **Test 1**.

Test 4. Take $c_h = 0.5h$. Ω is the same irregular obstacle as in **Test 3**. Results are shown in Figure 5. The value of the numerical solution u on the boundary γ in Figure 3 (b) is equal to 0, satisfying boundary condition in (2.1): $u(x, y) = g = 0$ on $\partial\Omega$, as in **Test 2**.

4.1. Conclusions

1. Figures 3 (b) and 5 (b) show that the numerical solutions satisfy boundary condition in (2.1): $u(x, y) = g = 0$ on $\partial\Omega$. Besides, the influence of the obstacle, $q(x, y) = 5$, can be found clearly in the figures. From the numerical verifications from the two tests, we believe that the convergent solutions in Figures 3 (b) and 5 (b) are correct. The distribution of the numerical solution in Figure 5 seems to be identical with that in [2]. Our computations are run on a personal computer with intel core CPU @ 2.30 GHz and 2.0 GB RAM. Each numerical test takes about less 12 minutes of CPU time. The results of numerical experiments show that our proposed methods are feasible and effective for the nonlinear obstacle problem (2.1).

2. Figures 2 (b), 3 (a), 4 (a) and 5 (a) show that the magnitude of c_h affects the distribution of the Eulerian nodes used near boundary γ of Ω for calculating the solution. We see that the smaller c_h is, the more accurate u on γ is. So if we choose c_h smaller then we will obtain more accurate numerical solution.

3. When we design Fortran program to solve the above problem, the Eulerian mesh on the extended domain $\hat{\Omega}$ is fixed even if the shape of Ω changes. We only need change the data representing boundary γ in the program. So the Fortran program is in common use for different Ω.

References

[1] D. Boffi, L. Gastaldi, *A finite element approach for the immersed boundary method*, Comput. Structures, **81** (2003), 491–501. 3

[2] H.-F. Chan, C.-M. Fan, C.-W. Kuo, *Generalized finite difference method for solving two-dimensional non-linear obstacle problems*, Eng. Anal. Bound. Elem., **37** (2013), 1189–1196. 4.1

[3] S. A. Enriquez-Remigio, A. M. Roma, *Incompressible flows in elastic domains: an immersed boundary method approach*, Appl. Math. Model., **29** (2005), 35–54. 3

[4] R. Glowinski, J.-L. Lions, R. Tremolieres, *Numerical analysis of variational inequalities*, North-Holland Publishing Co., Amsterdam-New York, (1976). 1, 2, 2

[5] R. Glowinski, T.-W. Pan, J. Periaux, *A fictitious domain method for Dirichlet problem and applications*, Comput. Methods Appl. Mech. Engrg., **111** (1994), 283–303. 1, 3

[6] P. Korman, A. W. Leung, S. Stojanovic, *Monotone iterations for nonlinear obstacle problem*, J. Austral. Math. Soc. Ser. B, **31** (1990), 259–276. 1, 2

[7] C. S. Peskin, *Numerical analysis of blood flow in the heart*, J. Computational Phys., **25** (1977), 220–252. 1

[8] C. S. Peskin, *The immersed boundary method*, Acta Numer., **11** (2002), 479–517. 1

[9] L. Rao, H. Q. Chen, *The technique of the immersed boundary method: application to solving shape optimization problem*, J. Appl. Math. Phy., **5** (2017), 329–340. 1, 2, 3, 3

[10] V. K. Saulev, *On the solution of some boundary value problems on high performance computers by fictitious domain method*, Siberian Math. J., **4** (1963), 912–925. 1

[11] A. L. F. L. E. Silva, A. Silveira-Neto, J. J. R.Damasceno, *Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method*, J. Computational Phys., **189** (2003), 351–370. 1