J Sci Comput (2011) 47: 258–280
DOI 10.1007/s10915-010-9436-4

Parallel Two-Grid Semismooth Newton-Krylov-Schwarz Method for Nonlinear Complementarity Problems

Haijian Yang · Xiao-Chuan Cai

Received: 18 June 2010 / Revised: 3 November 2010 / Accepted: 4 November 2010 /
Published online: 18 November 2010

Abstract We develop scalable parallel domain decomposition algorithms for nonlinear complementarity problems including, for example, obstacle problems and free boundary value problems. Semismooth Newton is a popular approach for such problems, however, the method is not suitable for large scale calculations because the number of Newton iterations is not scalable with respect to the grid size; i.e., when the grid is refined, the number of Newton iterations often increases drastically. In this paper, we introduce a family of Newton-Krylov-Schwarz methods based on a smoothed grid sequencing method, a semismooth inexact Newton method, and a two-grid restricted overlapping Schwarz preconditioner. We show numerically that such an approach is totally scalable in the sense that the number of Newton iterations and the number of linear iterations are both nearly independent of the grid size and the number of processors. In addition, the method is not sensitive to the sharp discontinuity often associated with obstacle problems. We present numerical results for several large scale calculations obtained on machines with hundreds of processors.

Keywords Complementarity problem · Semismooth Newton · Two-level methods · Grid sequencing · Parallel computing · Schwarz preconditioners

1 Introduction

Complementarity problems have many important applications in engineering modeling, operations research and the pricing of stock options; see, e.g., [8, 15, 20], and there are growing interests in developing efficient and robust algorithms for solving these semismooth

The research was supported in part by DOE under DE-SC0001774 and FC-02-06ER25784, and in part by NSF under grant DMS 0913089.

H. Yang
College of Mathematics and Econometrics, Hunan University, Changsha, Hunan, 410082, P.R. China
e-mail: haijianyang@gmail.com

X.-C. Cai (✉)
Department of Computer Science, University of Colorado at Boulder, Boulder, CO, 80309, USA
e-mail: cai@cs.colorado.edu

problems on large scale supercomputers. One popular approach is the class of semismooth methods which solves the complementarity problem by first reformulating it as a semismooth system of equations and then applying a generalized Newton method to solve this system. There are extensive theoretical and numerical results associated with this approach, see, e.g., [7, 9, 16, 23]. However, all existing approaches have scalability problems in the sense that when the degree of freedoms in the problem increases the number of nonlinear or linear iterations increases drastically. This prevents the algorithms from being useful for solving problems with large number of unknowns and on machines with large number of processors.

In this paper, we develop a class of general purpose two-grid Newton-Krylov-Schwarz (NKS) algorithms for complementarity problems associated with partial differential equations. The methods are based on an inexact semismooth Newton method, a smoothed grid sequencing method and a two-level cascade restricted overlapping Schwarz preconditioning technique. As it turns out, with an appropriate grid sequencing, the convergence rate of the semismooth Newton method can be made to be nearly independent of the number of unknowns of the system using either the Fischer-Burmeister function or the minimum function. This is a tremendous improvement over the approaches developed in [24, 38], in which the number of semismooth Newton iterations nearly doubles when the grid is refined by a factor of 2. Using the two-level restricted Schwarz preconditioner with sufficient overlap, the number of linear iterations also becomes nearly independent of the number of unknowns of the system. More important, both the linear and nonlinear iterations are nearly independent of the number of processors in our numerical experiments on machines with hundreds of processors.

We briefly mention a few related publications that partially motivated our current work. In [23], Kanzow developed an inexact semismooth Newton method and several algebraic preconditioners, such as Jacobi, Gauss-Seidel, and ILU, were studied together with some Krylov subspace methods for solving the Jacobian systems. Both the Fischer-Burmeister function and the minimum function were used in the paper. The results were not quite satisfactory, and the conclusion was that further research was necessary to identify suitable preconditioners for complementarity problems. In [38], Zvan et al. introduced a penalty method for complementarity problems, in which, the solution is obtained with an inexact Newton method and the linear approximate Jacobian systems are solved with the incomplete LU preconditioned CG method. The method was interesting, but the number of iterations is not scalable with respect to the size of the problem.

Multigrid is another class of methods being investigated by researchers for complementarity problems. Due to the nonsmoothness of the problem, special modifications are necessary. In [26], a monotone multigrid method was introduced to solve the obstacle problem and Gauss-Seidel type relaxation is used as the smoother. In [30], a multigrid method called the projected full approximation scheme was introduced to solve the linear complementarity problem. This multigrid method is used to accelerate the convergence of the basic relaxation method. Although this is a promising technique, multigrid methods are usually strongly coupled to the type of discretization used, and hence are complex to implement in general purpose software, as mentioned in [17].

In [21], Hintermüller et al. investigated the connection between the primal-dual active set method and the semismooth Newton method, and showed that the primal-dual active set method is a specific semismooth Newton method. In this method, the basic iteration consists of two steps: first, the domain is decomposed into active and inactive parts based on a certain criterion, and then a reduced linear system associated with the inactive set is solved. The main difficulty in this method is to find the active and inactive parts. In other words, if we can find the correct interface location of the active and inactive parts, then the corresponding complementarity problems are much easier to solve. In order to produce an estimate of the optimal active set, in [27] Morales et al. proposed a method that combines a projective Gauss-Seidel iteration with a subspace minimization step for symmetric linear complementarity problems. This method performs two types of iterations. First, the projective Gauss-Seidel method is used to generate an estimate of the active set, then with this estimate a sequence of subspace minimization steps is performed. Although this technique provides a good estimate of the active set after only a few iterations, but this method is not easy to implement for nonlinear complementarity problems.

We focus on Newton-Krylov methods [2–4, 6, 11, 12] that are not strongly coupled to the type of discretization used and are increasingly popular for the solution of nonlinear problems on large scale parallel computers. The nonlinear function derived from complementarity problems is generally not differentiable in certain part of the computational domain. For these components, some modified derivatives are used instead. To make the inexact Newton scalable, i.e., the number of iterations is nearly independent of the number of unknowns, we use a simple grid sequencing method which employs an interpolated coarse grid solution as the initial guess for the fine grid system. This strategy is very simple and has been used in many situations, but as far as we know, it has not been applied by others to complementarity problems. Our experiments show that this strategy provides an extremely good estimate of the active set. In terms of the total computing time, the most expensive part of the computation is in the solution of the Jacobian systems. To make the Jacobian solve scalable, we use a cascade type two-level restricted overlapping Schwarz preconditioned GMRES method. With this collection of algorithms, we are able to achieve good speedup on an IBM BG/L with up to 512 processors and for problems with more than 4 millions unknowns.

The rest of the paper is organized as follows. In Sect. 2, we recall the properties of some inexact semismooth Newton methods based on the Fischer-Burmeister function and the minimum function, respectively, and then we describe the details of the algorithms including grid sequencing and overlapping Schwarz preconditioning. Section 3 is devoted to numerical experiments and parallel performance of the methods. Finally, we offer some general remarks on the algorithms in Sect. 4.

2 Semismooth Function Approaches for Complementarity Problems

Let $\Omega \in R^2$ be a bounded open domain on which a linear or nonlinear differential operator $L(u)$ is defined. Many problems, including free boundary value problems, obstacle problems, minimal surface problems and optimal control problems [15], can be described as finding a function $u(x)$ defined in certain space such that

$$\begin{cases} L\mathrm{u}(x) \geq 0, & x \in \Omega \\ \mathrm{u}(x) \geq \Phi, & x \in \Omega \\ (\mathrm{u}(x) - \Phi)L\mathrm{u}(x) = 0, & x \in \Omega \end{cases} \tag{2.1}$$

with some boundary conditions assumed for $u(x)$, $x \in \partial\Omega$. Here Φ is given and often called an obstacle. We are only interested in discrete problems, therefore, we assume u_h is a vector in R^n representing the nodal values of $u(x)$ and $F(u_h)$ is a discretized version of $L(u)$. In the numerical experiments section we show some examples of $F(u_h)$ obtained by finite difference methods. Other methods, such as finite element or finite volume, can also be used.

In this paper, we consider the following complementarity problem:

$$\begin{array}{ll} \text{find} & u_h \in R^n, \\ \text{such that} & u_h \geq \phi, \quad F(u_h) \geq 0, \quad (u_h - \phi)^T F(u_h) = 0, \end{array} \tag{2.2}$$

where $F = (F_1, \dots, F_n)^T : R^n \to R^n$ denotes a continuously differentiable function from the discretization of a partial differential equation, and $\phi \in R^n$ is a given vector. In this paper, we focus on the nonlinear complementarity problem from the discretization of (2.1), even though our method is not limited to the obstacle problem (2.1). We refer interested readers to the survey papers [14, 20] on other problems.

2.1 Semismooth Newton Methods

The semismooth method is based on a reformulation of the complementarity problem (2.2) as the following nonlinear system of equations

$$\mathcal{F}(u_h) = 0, \tag{2.3}$$

where $\mathcal{F} : R^n \to R^n$ is componentwise given by

$$\mathcal{F}_i(u_h) \equiv \varphi((u_h - \phi)_i, F_i(u_h)), \quad i = 1, 2, \dots, n \tag{2.4}$$

for some function $\varphi : R^2 \to R$ with the following property

$$\varphi(a, b) = 0 \iff a \geq 0,\ b \geq 0,\ ab = 0. \tag{2.5}$$

Obviously, a vector $u_h^* \in R^n$ is the solution of the complementarity problem (2.2) if and only if u_h^* solves the system of (2.3). If we apply a Newton-type method with a generalized Jacobian to the system (2.3), then it leads to the class of semismooth methods; see, e.g., [9, 13, 22, 23, 37]. In the following, we consider the reformulations (2.3) and (2.4) of the complementarity problem (2.2) based on two different functions φ satisfying (2.5). One is the Fischer-Burmeister function

$$\varphi_{FB}(a, b) := a + b - \sqrt{a^2 + b^2} \tag{2.6}$$

which was introduced in [16], and the other is the minimum function

$$\varphi_{MIN}(a, b) := \min\{a, b\} \tag{2.7}$$

from [23]. In fact, the Fischer-Burmeister function is differentiable everywhere except at the point $(a, b) = (0, 0)$, while the minimum function is piecewise smooth with its non-differentiable points forming the line $\{(a, b)^T \in R^2 : a = b\}$.

The semismooth function is a class of functions between smooth and Lipschitzian functions. There are many functions that are semismooth such as convex functions, smooth functions and piecewise smooth functions. The classical Newton method for a smooth system $f(u_h) = 0$ is an iterative process. Suppose u_h^0 is the initial guess, then at point u_h^k, it solves the Jacobian problem

$$f(u_h^k) + f'(u_h^k) s_k = 0$$

to obtain the search direction s_k, then the next point is set to be $u_h^{k+1} = u_h^k + s_k$. The semismooth Newton method is to solve a system of semismooth equations $\mathcal{F}(u_h) = 0$, where $\mathcal{F}$ is semismooth, in which the Jacobian problem is replaced by

$$\mathcal{F}(u_h^k) + J_k s_k = 0, \tag{2.8}$$

where J_k is a generalized Jacobian of $\mathcal{F}(u_h^k)$ to be introduced below. It is proved (see [32]) that, under certain assumptions, the semismooth Newton method enjoys the same properties as smooth equations, such as locally superlinear convergence. A major advantage of these two reformulations is that the merit function

$$\Psi(u_h) := \frac{1}{2}\mathcal{F}(u_h)^T \mathcal{F}(u_h) \tag{2.9}$$

turns out to be continuously differentiable although the equation operator itself is nonsmooth [23]. Motivated by these observations, we next describe the structure of the Jacobian system (2.8). For both the Fischer-Burmeister function and the minimum function, the generalized Jacobian matrix J_k is of the form

$$J_k = D_a^k + D_b^k F^{'}(u_h^k) \tag{2.10}$$

with diagonal matrices (depending on the iteration index k)

$$D_a^k = \mathrm{diag}(d_{a_1}, \dots, d_{a_n}), \qquad D_b^k = \mathrm{diag}(d_{b_1}, \dots, d_{b_n}) \tag{2.11}$$

consisting of the partial derivatives of the mapping φ with respect to the first variable a and the second variable b, respectively, or a suitable approximation to these partial derivatives at those points where φ is not differentiable. Let $a_i = (u_h^k - \phi)_i$ and $b_i = F_i(u_h^k)$, the values of D_a^k and D_b^k in (2.11) corresponding to the Fischer-Burmeister function take the form

$$d_{a_i} := \begin{cases} 1 - a_i/\sqrt{a_i^2 + b_i^2}, & \text{if } a_i^2 + b_i^2 \neq 0, \\ 1, & \text{if } a_i^2 + b_i^2 = 0, \end{cases}$$

and

$$d_{b_i} := \begin{cases} 1 - b_i/\sqrt{a_i^2 + b_i^2}, & \text{if } a_i^2 + b_i^2 \neq 0, \\ 1, & \text{if } a_i^2 + b_i^2 = 0. \end{cases}$$

Similarly, when using the minimum function $\mathcal{F}(u_h) := \min\{(u_h - \phi), F(u_h)\}$, the values of D_a^k and D_b^k in (2.11) assume the form

$$d_{a_i} := \begin{cases} 1, & a_i < b_i, \\ 0, & a_i \geq b_i, \end{cases}$$

and

$$d_{b_i} := \begin{cases} 0, & a_i < b_i, \\ 1, & a_i \geq b_i. \end{cases}$$

Instead of solving (2.8) exactly, we use the so-called inexact Newton method. The inexactness of Newton's method is reflected in the fact that we do not solve the Jacobian systems exactly. The accuracy of the Jacobian solver is determined by a relative tolerance $\eta_r \in [0, 1)$, an absolute tolerance $\eta_a \in [0, 1)$ and the condition

$$\|\mathcal{F}(u_h^k) + J_k s_k\| \leq \max\{\eta_r \|\mathcal{F}(u_h^k)\|, \eta_a\}. \tag{2.12}$$

Once we have the search direction s_k, we obtain the next approximate solution by

$$u_h^{k+1} = u_h^k + \alpha^k s_k,$$

where the step length α^k is calculated by a cubic backtracking method [10], based on the merit function (2.9).

When using Newton's method to solve complementarity problems associated with partial differential equations discretized on a grid, one of the major problems is the deterioration of the convergence rate when the grid is refined. In particular, we and others have observed that when the grid is refined by a factor of 2, the number of Newton iterations increases by a factor of 2. After many experiments, we find a solution to the problem that is "grid sequencing" which is a popular approach for many problems but has not been used for complementarity problems. The idea consists of two steps. First, compute the solution of the problem on a coarse grid. Then interpolate the coarse solution to obtain the initial guess for the fine grid Newton iteration.

In order to use the grid sequencing method, we assume there are two grids covering Ω, a coarse grid of size H and a fine grid of size h, and we assume there is a coarse to fine grid interpolation operator I_H^h. Two nonlinear systems $\mathcal{F}_h(u_h) = 0$ and $\mathcal{F}_H(u_H) = 0$ are obtained by discretizing the continuous problem (2.1) on the two grids, respectively. In practice, it may take some effort to determine the proper size of the coarse grid. On one hand, the size of the coarse grid has to be small so that we can solve the coarse grid problem reasonably fast, and on the other hand, the coarse grid has to be sufficiently large so that the solution has certain accuracy. The idea can be used recursively, but we only study the two-grid case in this paper.

Another technique we find very useful is the smoothing of the interpolated coarse grid solution on the fine grid and the use of it as the fine grid initial guess. Suppose u_H^* is the final solution of the coarse grid system. In the simple grid sequencing method, one obtains the initial guess u_h^0 for the fine grid system using the interpolation $I_H^h u_H^*$. For many complementarity problems, the residual function $\mathcal{F}_h(u_h)$ is not smooth and is sensitive to the change of u_h. In other words, some small nonsmoothness of u_h may introduce large jumps in the residual function $\mathcal{F}_h(u_h)$. To overcome this problem, we propose to smooth the function $\tilde{u}_h^0 = I_H^h u_H^*$ using the following weights

$$\begin{array}{ccccc} \frac{1}{16} & & \frac{1}{8} & & \frac{1}{16} \\ & \searrow & \downarrow & \swarrow & \\ \frac{1}{8} & \rightarrow & \frac{1}{4} & \leftarrow & \frac{1}{8} \\ & \nearrow & \uparrow & \nwarrow & \\ \frac{1}{16} & & \frac{1}{8} & & \frac{1}{16} \end{array} \tag{2.13}$$

several times, and then take the smoothed $\tilde{u}_h^0$ as the initial guess u_h^0 for the fine grid Newton iteration. In Fig. 1, we show the surface plots of $\mathcal{F}_h(I_H^h u_H^*)$ without smoothing (left figure), and $\mathcal{F}_h(u_H^0)$ with one sweep of smoothing (right figure) for an obstacle problem. More details of this example (**Test 1**) will be discussed in the numerical experiments section of this paper.

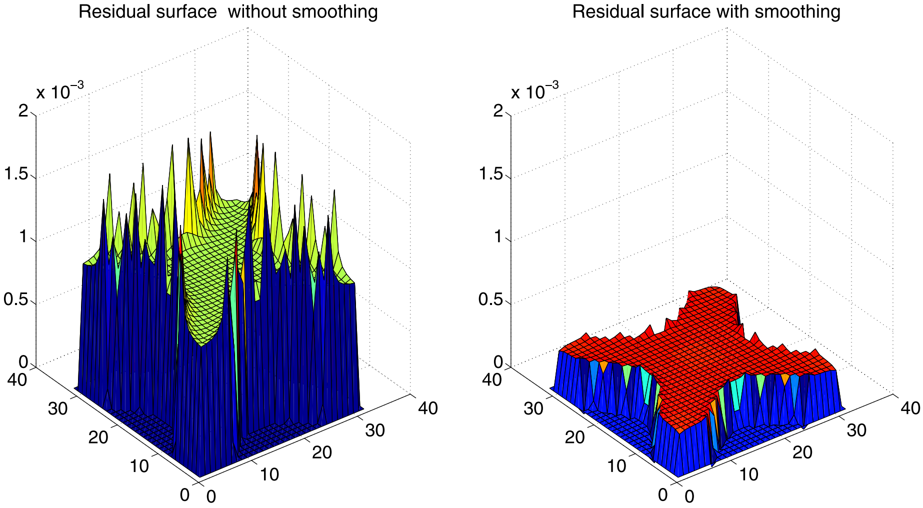


Fig. 1 The effect of smoothing of the interpolated coarse grid solution on the fine grid

2.2 Two-level preconditioning

The most expensive step in Newton type methods for solving complementarity problems is the Jacobian solve (2.12). The scalability of the whole algorithm depends primarily on how the Jacobian matrix is preconditioned. In stead of solving (2.12), we solve a right-preconditioned problem

$$\|\mathcal{F}(u_h^k) + J_k M_k^{-1}(M_k s_k)\| \le \max\{\eta_r \|\mathcal{F}(u_h^k)\|, \eta_a\},$$

where M_k^{-1} is an overlapping Schwarz preconditioner [31, 35, 36].

To define the Schwarz preconditioner, we need to obtain an overlapping partition of Ω. We first divide Ω into non-overlapping subdomains Ω_i, $i = 1, \dots, N_s$, and then expand each Ω_i to Ω_i^δ, i.e., $\Omega_i \subset \Omega_i^\delta \subset \Omega$. The overlap $\delta > 0$ is defined as the minimum distance between $\partial\Omega_i^\delta$ and $\partial\Omega_i$, in the interior of Ω_i. For boundary subdomains we simply cut off the part outside Ω. Let $H_D > 0$ denote the characteristic diameter of $\{\Omega_i\}$.

Let N and N_i denote the number of grid points associated to Ω and Ω_i^δ, respectively. Let J be the $N \times N$ sparse matrix of the Jacobian system

$$Jp = b. \tag{2.14}$$

We define the $N_i \times N$ matrix R_i^δ as follows: its element $(R_i^\delta)_{l_1,l_2}$ is either 1 if the integer indices $1 \le l_1 \le N_i$ and $1 \le l_2 \le N$ are related to the same grid point and this grid point belongs to Ω_i^δ or 0 otherwise. The multiplication of R_i^δ with a $N \times 1$ vector generates a shorter $N_i \times 1$ vector by discarding all components corresponding to grid points outside Ω_i^δ. The $N_i \times N$ matrix R_i^0 is similarly defined, with the difference that its application to a $N \times 1$ vector also zeroes all those components corresponding to grid points on $\Omega_i^\delta \setminus \Omega_i$. We denote by J_i the $N_i \times N_i$ subdomain matrix given by

$$J_i = R_i^\delta \; J \; (R_i^\delta)^T.$$

We assume J_i is nonsingular and denote by B_i^{-1} either the inverse of or a preconditioner for J_i. The one-level restricted additive Schwarz (RAS) preconditioner for J is defined as

$$M_{RAS}^{-1} = \sum_{i=1}^{N_s} (R_i^0)^T B_i^{-1} R_i^\delta. \tag{2.15}$$

The one-level preconditioner works well when the number of processors is small, however, for the case of large number of processors, a coarse preconditioner is often incorporated into M_{RAS}^{-1} to reduce the number of iterations. We refer to [5, 6, 18, 28] for further analysis and examples of one-level restricted additive Schwarz preconditioning techniques. There are several ways to include a coarse preconditioner to the one-level preconditioner M_{RAS}^{-1} [35, 36]. We choose to use a method that is very similar to the grid sequencing technique just discussed, and the same coarse grid is employed for the linear preconditioner. At each step of Newton iteration, similar to the Jacobian matrix J, we calculate a coarse grid approximation J_c by using (2.10) and $I_h^H u_h^k$, where I_h^H is a restriction operator from the fine grid to the coarse grid.

When the coarse grid is small, a sequential or parallel direct method is suitable for computing $J_c^{-1} w_c$ for any coarse vector w_c, one can then define the coarse grid preconditioner as

$$M_c^{-1} = I_H^h J_c^{-1} I_h^H.$$

Then, the cascade type two-level Schwarz preconditioner can be defined as

$$M^{-1} = M_c^{-1} + M_{RAS}^{-1} - M_{RAS}^{-1} J M_c^{-1}.$$

In many applications, the coarse grid has to be reasonably fine and this makes it not possible to use any direct methods. In this case, which is the situation that we are interested in this paper, a preconditioned iterative method has to be used to solve the coarse problem

$$x_c = J_c^{-1} w_c \quad \text{or} \quad J_c x_c = w_c.$$

Similar to the one-level preconditioner on the fine grid, we introduce a one-level preconditioner on the coarse grid

$$B_c^{-1} = \sum_{i=1}^{N_s} (R_{c,i}^0)^T B_{c,i}^{-1} R_{c,i}^{\delta_c}.$$

Here $B_{c,i}$ is the restriction of J_c on the subdomain $\Omega_i^{\delta_c}$, and $R_{c,i}^{\delta_c}$ and $R_{c,i}^0$ are the restriction operators on the coarse grids defined on $\Omega_i^{\delta_c}$ and Ω_i, respectively. $x_c = J_c^{-1} w_c$ is computed by approximately solving the following problem

$$\|w_c - J_c B_c^{-1} x_c'\| \le \max\{\eta_r^c \|w_c\|, \eta_a^c\},$$

where η_r^c, η_a^c are pre-selected tolerances, and then set $x_c = B_c^{-1} x_c'$. Depending on the tolerances, the coarse problem may be solved very accurately or less accurately. More discussion will be given in the numerical experiments section of the paper. When an iterative method is used for solving the coarse grid problem, the overall preconditioner is no longer is static matrix; i.e., it is an iterative procedure. In other words, the preconditioner changes from iteration to iteration. Depending on if the matrix is symmetric and positive definite, or if the preconditioner changes during the iteration, there are several choices of the outer iterative methods including CG, GMRES and FGMRES [34]. Because the coarse solve is iterative, the two-level preconditioner changes during the outer iteration, therefore, in theory, we should use FGMRES, which is more expensive in terms of floating point operations and memory requirement. However, as we find out in our numerical experiments, the regular GMRES converges with no problem. Therefore, in this paper, we simply use GMRES for all cases.

3 Numerical Experiments

We report some results of our numerical experiments. The parallel software is developed using the Portable, Extensible Toolkit for Scientific Computing (PETSc) library of Argonne National Laboratory [1]. The numerical tests are carried out on an IBM BlueGene/L using up to 512 compute node. Each node has 512 MB of memory.

3.1 Test cases

We study three model problems defined on $\Omega = (0, 1) \times (0, 1)$. First, we consider an obstacle problem (**Test 1**): find $u(x)$ such that

$$\begin{cases} -\Delta u(x) + C \geq 0, & x \in \Omega, \\ u(x) \geq -\mathrm{d}(x, \partial\Omega), & x \in \Omega, \\ (u(x) + \mathrm{d}(x, \partial\Omega))(-\Delta u(x) + C) = 0, & x \in \Omega, \\ u(x) = 0, & x \in \partial\Omega, \end{cases} \tag{3.1}$$

where the $\mathrm{d}(x, \partial\Omega)$-operator measures the distance from a point x to the domain boundary $\partial\Omega$. The problem describes the elastoplastic torsion of a cylindrical bar with quadratical cross section [24, 25, 30]. As the parameter C becomes smaller, the problem becomes more difficult to solve. In [25], the problem was considered to be very difficult with the parameter $C = 20$. In [30], Oosterlee presents level curves of the solution and the small region of "inactive" points, where $-\Delta u(x) + C = 0$ is valid. This region, whose size depends on parameter C, represents the plastic region, whereas the active points, where the second constraint with equality sign is valid, represents the elastic region (see [33] for more details).

Secondly, following Nochetto et al. [29], we consider the problem (**Test 2**):

$$\begin{cases} -\Delta u(x) - g(x) \geq 0, & x \in \Omega, \\ u(x) \geq 0, & x \in \Omega, \\ u(x)(-\Delta u(x) - g(x)) = 0, & x \in \Omega, \end{cases} \tag{3.2}$$

where

$$g(x) = \begin{cases} -4(2|x|^2 + 2(|x|^2 - r^2)), & |x| > r, \\ -8r^2(1 - (|x|^2 - r^2)), & |x| \leq r. \end{cases}$$

Here r is a given constant which controls the region of "inactive" points. The problem has the exact solution

$$u(x) = \left(\max\{|x|^2 - r^2, 0\}\right)^2.$$

The boundary condition is defined by using the exact solution.

Finally, we study the obstacle Bratu problem from [13, 33] (**Test 3**):

$$\begin{cases} -\Delta u(x) + \lambda e^{-u(x)} \geq 0, & x \in \Omega, \\ u(x) \geq \Phi, & x \in \Omega, \\ (u(x) - \Phi)(-\Delta u(x) + \lambda e^{-u(x)}) = 0, & x \in \Omega, \end{cases} \tag{3.3}$$

where the parameter $\lambda \geq 0$ and the boundary condition $u(x) = 0$. Here we are interested in the particular instance given by

$$\Phi = -4, \quad \lambda = 1$$

which are also used as a complementarity test problem in, for example, [13, 19, 23].

For the discretization we use the standard second-order five-point finite difference method on a uniform grid. Taking **Test 1** and the minimum function as an example, we build the discrete semismooth function at a grid point, formally defined in (2.3), as

$$\mathcal{F}_{ij}(u_h) = \min\left\{u_{i,j} + \mathrm{d}(x, \partial\Omega), 4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} + Ch^2\right\}. \tag{3.4}$$

Note that for better results, we scale the function with h^2. We can define the semismooth function corresponding to the Fischer-Burmeister function in a similar way.

The initial guess u_h^0 for the global Newton iteration is the obstacle ϕ in (2.2) for **Test 1** and **Test 2**, and is -1.0 for **Test 3**. We stop the fine grid Newton iteration if

$$\|\mathcal{F}(u_h^k)\| \leq \max\{10^{-6}\|\mathcal{F}(u_h^0)\|, 10^{-10}\}.$$

The fine grid Jacobian system is solved with GMRES(30), and the iteration is stopped if the tolerance

$$\|\mathcal{F}(u_h^k) + J_k s_k\| \leq \max\{10^{-4}\|\mathcal{F}(u_h^k)\|, 10^{-10}\}$$

is satisfied. The subdomain problems are solved with LU factorization. Throughout this section,"*np*" stands for the number of processors which is the same as the number of subdomains, "INB' the number of inexact Newton iterations, "RAS" the number of RAS preconditioned GMRES iterations per Newton iteration, and "Time" the total compute time in seconds.

3.2 One-Level Results

We first study the one-level method for **Test 1** with $C = 5$ and overlap $\delta = 3$. In Table 1, we present the number of Newton iterations, the average number of GMRES iterations per Newton, and the total computing time. On a fixed grid, the number of Newton iterations is independent of the number of processors, but the number of GMRES iterations increases as the number of processors increases for both the Fischer-Burmeister function and the minimum function. The major problem with the one-level method shows up, if we look at the scalability for a fixed number of processors. For each row in the table, every time we refine the grid by a factor of 2, the number of Newton iterations doubles. This problem prohibits the use of the method for high resolution applications.

In Table 2, we show the effect of overlap on the performance of the algorithm. The grid is 512×512. We can see that in general the algorithm converges better as the overlap increases. On the other hand, a larger overlap also increases the inter-processor communication and

Table 1 Results for **Test 1** using the one-level method with overlap $\delta = 3$

Mesh	256×256			512×512			1024×1024			2048×2048		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
					The Fischer-Burmeister function							
64	82	11.4	3.3	162	14.7	32.5	320	19.1	384.1	639	24.4	4781.1
128	82	13.6	2.1	162	17.5	17.4	320	22.2	180.4	639	30.8	2236.3
256	82	14.4	1.5	162	18.9	10.2	320	24.3	95.9	639	34.1	1110.0
512	82	17.2	1.1	162	22.6	7.5	320	32.3	62.3	639	38.5	568.9
					The minimum function							
64	80	11.7	2.9	159	15.3	29.5	319	19.9	361.9	637	26.4	4673.0
128	80	14.0	1.9	159	18.3	16.1	319	23.7	173.5	637	33.7	2201.8
256	80	14.9	1.4	159	19.7	9.7	319	26.1	94.4	637	36.5	1104.8
512	80	17.7	1.3	159	23.8	7.3	319	34.5	62.0	637	41.1	567.5

Table 2 Effect of the overlapping size δ using the one-level method and the Fischer-Burmeister function for **Test 1** with the parameter $C = 5$. The grid is 512×512

δ	1			2			3			4		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
64	162	22.2	34.0	162	17.4	33.1	162	14.7	32.5	162	13.1	33.2
128	162	27.6	18.2	162	20.5	17.7	162	17.5	17.4	162	15.6	17.9
256	162	32.1	11.0	162	23.3	10.3	162	18.9	10.2	162	16.8	12.3
512	162	39.2	8.2	162	28.5	7.6	162	22.6	7.5	162	20.0	8.3

the overall computing time. A moderate overlap, $\delta = 3$, provides the best result in terms of timing.

The difficulty of **Test 1** changes with the parameter C. In Table 3 we present results for this test case using the one-level method with overlap $\delta = 3$ for both the Fischer-Burmeister function and the minimum function on a 512×512 grid. It is clear that as the parameter C becomes smaller, both the iteration number and the execution time increase rapidly. In Fig. 2, we present a comparison of the active/inactive sets with the different values of C. The inactive sets are the colored regions and the active sets are the white regions.

Next we look at **Test 2** with $r = 0.7$. The numerical results for this example are given in Table 4. As r decreases, the size of the inactive set increases and the problem becomes harder to solve. Similar to **Test 1**, every time we refine the grid by a factor of 2, the number of Newton iterations doubles.

Finally, we test the one-level method for **Test 3**. As shown in Table 5, the performance of the algorithm is quite different from the other two test cases. At the nonlinear level, the number of Newton iterations is totally independent of the grid size. However, for the linear solver, the number of GMRES iterations increases as the grid is refined. It is interesting to note that the minimum function is now a much better choice than the Fischer-Burmeister function since it requires only half the number of Newton iterations. In [23], for the same test case, the linear systems are solved using a conjugate gradient method. The numerical results in [23] indicate that the penalized Fischer-Burmeister function is much better than

Table 3 Results for **Test 1** with different values of C by using the one-level method with overlap $\delta = 3$ on 512×512 grid

C	5			10			15			20		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
The Fischer-Burmeister function												
64	162	14.7	32.5	82	9.6	14.0	56	7.4	8.7	43	6.3	6.4
128	162	17.5	17.4	82	11.3	7.3	56	8.9	4.5	43	7.7	3.3
256	162	18.9	10.2	82	11.7	4.2	56	9.0	2.6	43	7.5	1.9
512	162	22.6	7.5	82	13.8	3.1	56	10.4	1.9	43	8.6	1.4
The minimum function												
64	159	15.3	29.5	79	10.0	11.2	53	7.5	6.0	40	6.4	3.9
128	159	18.3	15.3	79	11.7	6.1	53	9.0	3.4	40	7.8	2.3
256	159	19.7	9.7	79	12.0	3.7	53	9.0	2.2	40	7.5	1.5
512	159	23.8	7.3	79	14.3	2.8	53	10.6	1.7	40	8.7	1.2

Table 4 Results for **Test 2** using the one-level method with overlap $\delta = 3$

Mesh	256×256			512×512			1024×1024			2048×2048		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
The Fischer-Burmeister function												
64	130	12.7	5.5	258	16.2	53.4	512	20.6	640.1	1021	27.6	8120.8
128	130	15.1	3.6	258	19.4	29.1	512	26.0	310.4	1021	32.9	3716.9
256	130	16.2	2.6	258	21.3	17.3	512	29.0	165.9	1021	35.7	1819.8
512	130	19.4	2.3	258	27.1	13.1	512	34.3	102.7	1021	43.4	969.8
The minimum function												
64	128	12.9	4.9	257	16.6	49.9	513	21.2	616.1	1024	28.7	7972.0
128	128	15.4	3.3	257	19.8	27.9	513	27.0	300.1	1024	34.0	3648.8
256	128	16.5	2.5	257	22.2	16.6	513	30.0	162.7	1024	37.0	1801.6
512	128	19.8	2.3	257	27.9	12.9	513	35.3	101.2	1024	45.3	972.0

the minimum function approach. Also, the number of CG iterations in [23] is very large. This suggests that our RAS preconditioned GMRES is more suitable for this problem.

3.3 Two-Level Results

In this subsection, we present some numerical results using the two-level approach in which a coarse grid is used in the nonlinear solver for generating a better initial guess and also in the linear solver for generating part of the Schwarz preconditioner. We report results for **Test 1** and **Test 2**. In all tests, the initial guess for the global Newton iteration on the coarse grid is the obstacle ϕ in (2.2), and the tolerance conditions on the fine grid are the same as in the one-level method.

One aspect of the two-level NKS algorithm should be highlighted before we proceed to the presentation of numerical experiments. In order to obtain a good initial guess for the fine grid Newton iteration, we need to know how accurately the coarse nonlinear system should be solved. In other words, one has to choose an appropriate stopping condition in order to

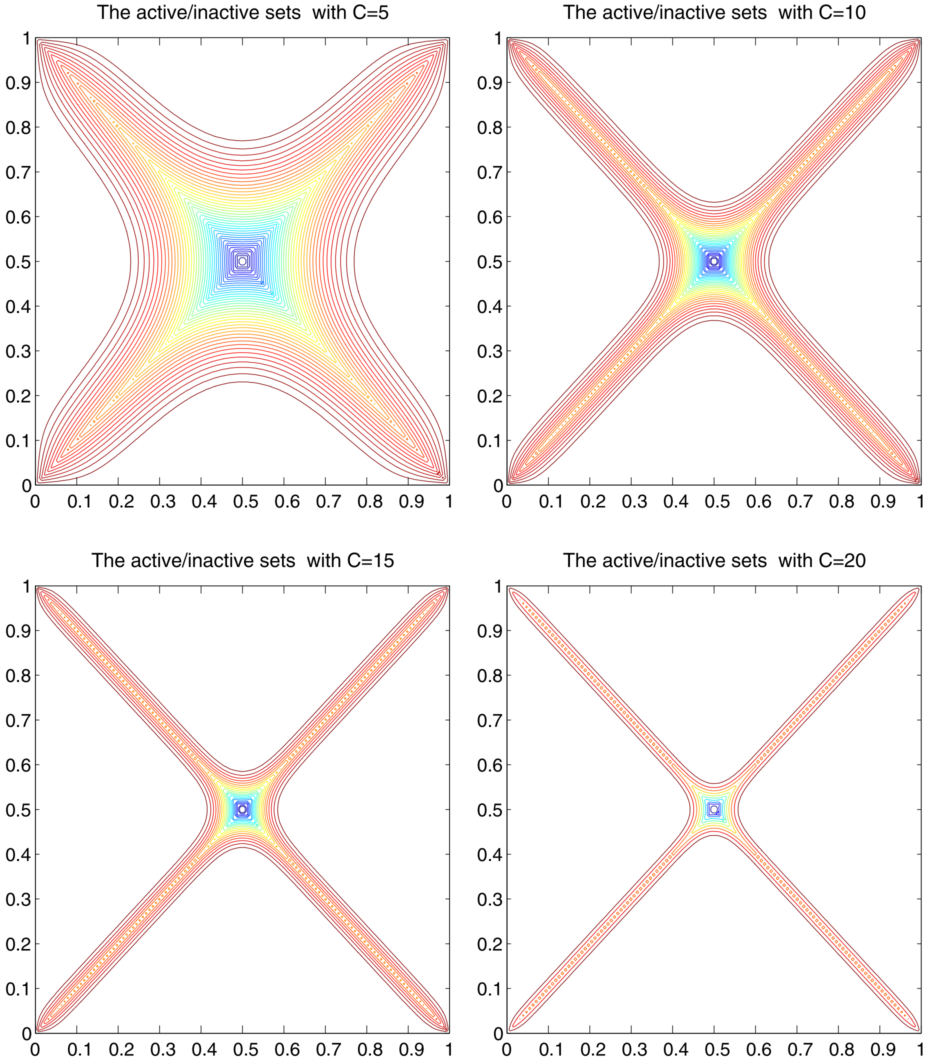


Fig. 2 The active (*colored*) and inactive (*white*) sets with different C for **Test 1** computed on a 128×128 grid

balance the number of Newton iterations on the fine grid and the total computing time. In our experiments, we find that for **Test 1** we should stop the coarse grid Newton iteration if

$$\|\mathcal{F}_H(u_H^k)\| \leq \max\{10^{-4}\|\mathcal{F}_H(u_H^0)\|, 10^{-10}\},$$

and for **Test 2** we should stop the coarse grid Newton iteration if

$$\|\mathcal{F}_H(u_H^k)\| \leq \max\{10^{-6}\|\mathcal{F}_H(u_H^0)\|, 10^{-10}\}.$$

Table 5 Results for **Test 3** using the one-level method with overlap $\delta = 3$

Mesh	256×256			512×512			1024×1024			2048×2048		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
					The Fischer-Burmeister function							
64	4	20.8	0.2	4	20.8	1.2	4	44.8	7.8	4	66.5	50.9
128	4	26.0	0.2	4	37.8	0.7	4	61.0	4.2	4	101.5	29.7
256	4	26.8	0.1	4	49.5	0.5	4	62.8	2.1	4	133.3	17.2
512	4	36.5	0.1	4	58.0	0.3	4	97.0	1.6	4	189.5	10.8
					The minimum function							
64	2	23.0	0.13	2	40.0	0.7	2	65.5	4.9	2	75.5	27.7
128	2	31.0	0.09	2	55.0	0.5	2	86.0	2.7	2	153.0	20.3
256	2	30.5	0.07	2	64.0	0.3	2	99.0	1.5	2	157.0	9.9
512	2	49.5	0.07	2	74.5	0.2	2	123.5	1.0	2	177.5	5.2

Table 6 Effect of the fine grid overlapping size δ for **Test 1**. The semismooth function is the Fischer-Burmeister function. The fine grid is 512×512 and the coarse grid is 128×128. The coarse grid overlap is $\delta_c = 3$. The preconditioner is the two-level RAS

δ	2			4			6			8		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
64	7	23.6	7.1	7	17.9	6.4	7	15.3	6.2	7	13.7	6.7
128	7	29.0	5.9	7	21.4	4.9	7	18.4	4.8	7	16.6	4.9
256	7	36.6	4.9	7	23.9	3.9	7	20.3	3.6	7	18.0	3.5
512	7	46.1	6.9	7	28.1	4.9	7	23.9	4.3	7	21.6	4.3

In all tests, the Jacobian system on the coarse grid is solved with a one-level RAS preconditioned GMRES(30) with the following stopping condition

$$\|\mathcal{F}_H(u_H^k) + J_H^k M_{H,RAS}^{-1}(M_{H,RAS} s_k)\| \leq \max\{10^{-4}\|\mathcal{F}_H(u_H^k)\|, 10^{-10}\},$$

where $M_{H,RAS}^{-1}$ is defined similar to (2.15) on the coarse grid. The subdomain problems are solved with LU factorization.

Consider **Test 1** with $C = 5$. For the sake of brevity, we assume the coarse overlap $\delta_c = 3$. In Table 6, we show the number of iterations and the total computing time with different fine grid overlap δ. The fine grid is 512×512 and the coarse grid is 128×128. We observe that, in general, the algorithm converges better as the overlap increases. However, a larger overlap also increases the inter-processor communication. Consequently, the best choice for the fine grid overlap is $\delta = 6$, which is the value we use for other tests in this subsection.

Using $\delta = 6$ and $\delta_c = 3$, we solve **Test 1** on several different fine grids with the two-level method and the results are summarized in Table 7, for both the Fischer-Burmeister function and the minimum function. The main concern is the size of the coarse grid H, which is taken as $h/2$, $h/4$ and $h/8$, where h is the size of the fine grid. In terms of the total number of Newton iterations, $H = h/2$ is certainly the best, but $H = h/8$ offers the best results in terms of the total computing time. Note that some cases, marked as "$*$", for the fine grid 256×256 are not available because the corresponding coarse grids are too coarse and the discretization is no longer valid. The computing time includes the coarse grid calculation

Table 7 Results for **Test 1** with different fine and coarse grids. The overlapping sizes of the coarse grid and the fine grid are $\delta_c = 3$ and $\delta = 6$, respectively. The preconditioner is the two-level RAS. h and H are the fine and coarse grid sizes, respectively

Mesh	256 × 256			512 × 512			1024 × 1024			2048 × 2048		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
The Fischer-Burmeister function												
$H = h/2$												
64	6	10.8	2.4	5	15.8	11.5	4	21.8	76.0	4	26.3	848.5
128	6	13.0	2.2	5	18.8	8.5	4	26.3	49.0	4	35.8	536.1
256	6	13.8	1.8	5	20.8	6.0	4	30.0	33.6	4	37.8	291.9
512	6	19.3	2.8	5	24.4	5.9	4	34.6	32.9	4	43.0	206.5
$H = h/4$												
64		*		7	15.3	6.2	7	19.0	33.9	6	25.8	201.7
128		*		7	18.4	4.8	7	22.6	21.9	6	32.7	120.8
256		*		7	20.3	3.6	7	25.4	25.4	6	38.7	71.6
512		*		7	23.9	4.3	7	33.9	12.1	6	43.7	56.6
$H = h/8$												
64		*		9	15.7	5.9	9	19.7	31.0	8	26.4	169.9
128		*		9	19.1	4.6	9	23.7	18.8	8	33.9	99.2
256		*		9	21.0	3.5	9	26.6	10.7	9	36.6	54.3
512		*			*		9	34.3	10.8	8	45.3	34.4
The minimum function												
$H = h/2$												
64	2	14.0	1.3	3	14.3	8.3	3	17.7	60.4	2	31.5	777.7
128	2	16.5	1.2	3	16.7	6.0	3	21.7	40.6	2	39.0	446.3
256	2	17.5	1.0	3	18.3	4.3	3	26.3	26.3	2	49.5	260.6
512	2	25.0	1.5	3	20.0	3.9	3	29.7	22.2	2	57.0	178.5
$H = h/4$												
64		*		4	15.5	3.9	4	18.8	21.1	5	19.6	160.3
128		*		4	18.8	2.9	4	24.8	14.6	5	23.2	93.1
256		*		4	20.3	2.3	4	28.0	9.2	5	24.4	50.8
512		*		4	24.5	2.8	4	33.0	7.7	5	30.6	39.7
$H = h/8$												
64		*		7	15.6	4.6	6	21.3	21.7	6	25.7	126.9
128		*		7	18.6	3.5	6	24.8	13.1	6	33.3	74.2
256		*		7	20.7	2.7	6	26.7	7.3	6	37.0	37.0
512		*			*		6	34.8	7.4	6	42.5	25.4

of the initial guess, the smoothing of the coarse solution, and the solving of the fine grid problem. Note the minimum function approach is always faster than the Fischer-Burmeister function approach in terms of all measures.

In the tables, "INB" refers only to the number of Newton iterations on the fine grid. Since the nonlinear system on the coarse grid is also solved with the semismooth Newton-Krylov-Schwraz method, with a one-level preconditioner (the only difference is the Newton stopping condition), the number of Newton iterations on the coarse grid is similar to the one-level case with a similar grid size. Taking **Test 1** with the Fischer-Burmeister function

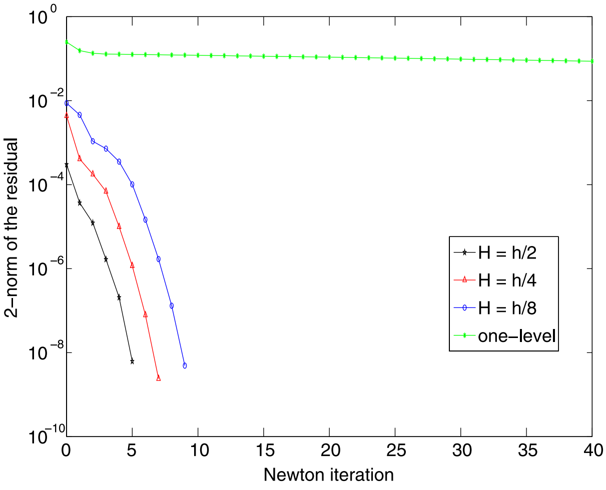


Fig. 3 Nonlinear residual history for **Test 1** with the Fischer-Burmeister function on a 512×512 grid and 256 processors. In the case of two-level, the overlapping sizes of the coarse grid and the fine grid are $\delta_c = 3$ and $\delta = 6$, respectively. In the case of one-level, the overlapping size is $\delta = 3$. h and H are the fine and coarse grid sizes, respectively

approach and $h/2$ as an example, the number of Newton iterations on the coarse grid for the 2048×2048 fine grid (i.e., the coarse grid is 1024×1024) is 318.

If we compare the number of Newton iterations of the one-level method and the two-level method for **Test 1**, it is clear the two-level method is tremendously better. In particular the minimum function approach results in a very small number of Newton iterations and the computing time. In Fig. 3, we show the nonlinear residual history for **Test 1** using the one-level and two-level methods with the Fischer-Burmeister function on a 512×512 grid and 256 processors. From Fig. 3, one can see that the nonlinear system is extremely difficult to solve by using the one-level method. In fact the one-level method converges after 162 Newton iterations. On the other hand, the two-level method converges easily for different grids. The number of global Newton iterations does not change much for the different grids. We mention that in these tests the coarse solution is smoothed before it is interpolated to obtain the initial guess on the fine grid.

Similar results are reported in Table 8 for **Test 2** with $r = 0.7$. From Table 8, we see that when we fix the number of processors and refine the fine grid, the number of Newton iterations increases in some cases. The reason is that the solution of the coarse nonlinear system is not accurate enough for these cases. If we solve the coarse nonlinear systems more accurately, then the number of Newton iterations can be reduced. Table 9 presents some results with different stopping conditions for the coarse Newton iteration. It is clear that the initial guess from the coarse grid solution has to be close enough to the desired solution.

We next consider the parallel scalability issue of the one-level and two-level methods. For a fixed grid, the speedup is defined as the ratio of the total computing time required by using 64 processors and np processors. 64 is the smallest number of processors that the problems can be solved.

The one-level results are shown in Fig. 4. The calculations are done for **Test 1** on a 2048×2048 grid on 64, 128, 256, and 512 processors. In the left figure, the three speedup curves are for the Fischer-Burmeister function, the minimum function and the idea case. Here "ideal" means linear speedup. The corresponding computing times are given in the

Table 8 Results for **Test 2** with different fine and coarse grids. The overlapping sizes of the coarse grid and the fine grid are $\delta_c = 3$ and $\delta = 6$, respectively. The preconditioner is the two-level RAS. h and H are the fine and coarse grid sizes, respectively

Mesh	256×256			512×512			1024×1024			2048×2048		
np	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time	INB	RAS	Time
The Fischer-Burmeister function												
$H = h/2$												
64	5	13.6	3.0	4	18.5	14.4	4	21.3	110.1	9	23.9	1639.9
128	5	16.0	2.7	4	22.0	10.9	4	24.3	70.4	9	27.7	995.4
256	5	17.2	2.3	4	24.0	7.9	4	28.0	48.0	9	30.0	537.3
512	5	23.2	3.4	4	29.3	7.8	4	36.0	39.2	9	35.8	384.2
$H = h/4$												
64		*		7	17.0	7.2	6	22.3	36.0	5	28.0	216.4
128		*		7	19.9	5.7	6	27.0	24.3	5	33.0	132.9
256		*		7	21.7	4.4	6	29.3	15.1	6	32.5	85.2
512		*		7	26.1	5.6	6	32.8	13.4	6	38.5	38.5
$H = h/8$												
64		*		9	18.0	6.7	8	22.5	30.6	8	25.0	173.7
128		*		9	21.2	5.3	8	26.4	19.0	8	28.3	93.5
256		*		9	23.3	4.1	8	30.1	11.2	8	32.4	47.5
512		*			*		8	36.5	11.3	8	40.5	35.3
The minimum function												
$H = h/2$												
64	2	14.0	1.8	3	14.3	11.3	3	18.0	96.0	5	23.0	1390.0
128	2	17.0	1.7	3	17.3	8.5	3	21.7	61.1	5	27.0	788.3
256	2	17.0	1.4	3	19.0	6.2	3	25.3	40.9	5	28.4	421.7
512	2	23.5	1.9	3	23.7	6.1	3	30.3	31.3	5	33.2	280.7
$H = h/4$												
64		*		4	15.8	4.362	4	20.3	24.8	4	23.3	175.8
128		*		4	19.8	3.603	4	25.0	17.4	4	26.3	103.7
256		*		4	20.0	2.708	4	27.3	10.7	4	31.5	63.1
512		*		4	25.5	3.589	4	30.8	9.6	4	42.0	50.6
$H = h/8$												
64		*		6	17.8	4.5	6	22.3	23.1	6	26.3	133.9
128		*		6	20.7	3.5	6	26.5	14.3	6	31.7	75.4
256		*		6	22.8	2.7	6	29.7	8.4	6	35.5	38.5
512		*			*		6	37.5	8.8	6	41.7	27.6

right figure. The three curves are amazingly close to each other. This indicates that the one-level methods are quite scalable at least for this range of number of processors. Of course, good scalability doesn't mean the algorithms are fast in terms of the total computing time, which is often more important.

In Fig. 5, we show the speedup and computing time curves for solving the problem using the two-level methods. The results are obtained for **Test 1** using the minimum function on a 2048×2048 grid on 64, 128, 256, and 512 processors and the coarse grid is chosen as $1/2$, $1/4$ and $1/8$ of the fine grid. In comparison with the one-level method, the two-level methods

Table 9 Effect of different stopping conditions on the coarse grid. The semismooth function is the Fischer-Burmeister function. The fine grid is 512 × 512 and the coarse grid is 256 × 256. The overlapping sizes of the coarse grid and the fine grid are $\delta_c = 3$ and $\delta = 6$, respectively. η_a^c is the absolute convergence tolerance and η_r^c is the relative convergence tolerance of the coarse Newton iteration

Convergence tolerance	$\eta_a^c = 10^{-10},\ \eta_r^c = 10^{-6}$			$\eta_a^c = 10^{-10},\ \eta_r^c = 10^{-8}$		
np	INB	RAS	Time	INB	RAS	Time
64	9	23.9	1639.9	3	30.0	1276.9
128	9	27.7	995.4	3	39.3	733.1
256	9	30.0	537.3	3	43.0	404.6
512	9	35.8	384.2	3	46.3	262.0

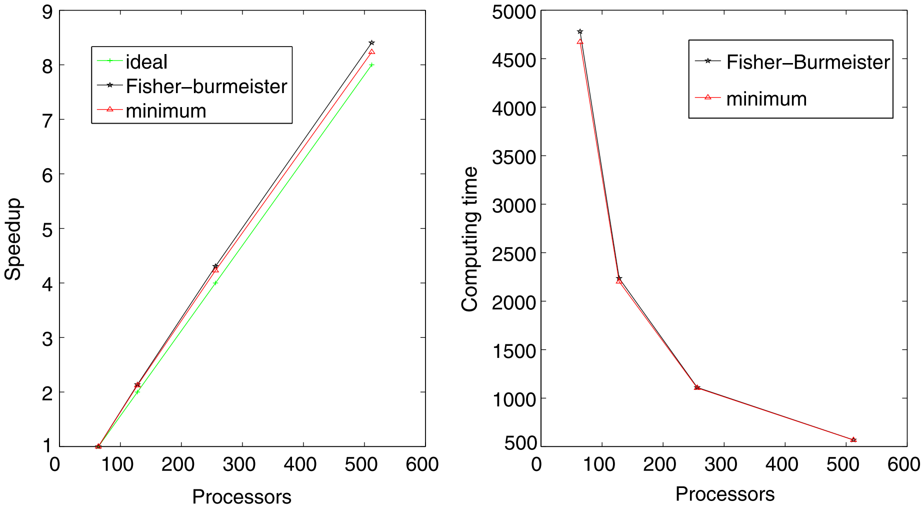


Fig. 4 The speedup and the total computing time of the one-level method with different semismooth functions for **Test 1**. The grid is 2048 × 2048, and the overlapping size is $\delta = 3$

are not as scalable, however, the total computing time is much lower. The computing time curves also show that better timings are achieved when the coarse grid is not too fine.

We should also mention that the use of smoothed grid sequencing plays an important role in the two-level methods. In Fig. 1, the surface plots of the residual function before and after the smoothing are shown and they are quite different. The cost of the smoothing step is very small and fewer number of Newton iterations is needed as a result of the smoothing.

To see the major difference between the one-level and the two-level methods, we show the surface plots of the residual function for the entire history of Newton iterations in Figs. 6 and 7 (some figures are skipped to save space). The calculations are obtained for **Test 1** with $C = 5$ on a 128 × 128 grid. Through many experiments, we find that the most difficult part of the computation is to correctly locate the interface of the elastic and plastic regions. As Newton progresses, the approximate location of the interface shows up as the peak of the surface plot of the residual function. For **Test 1** with $C = 5$, the precise location of

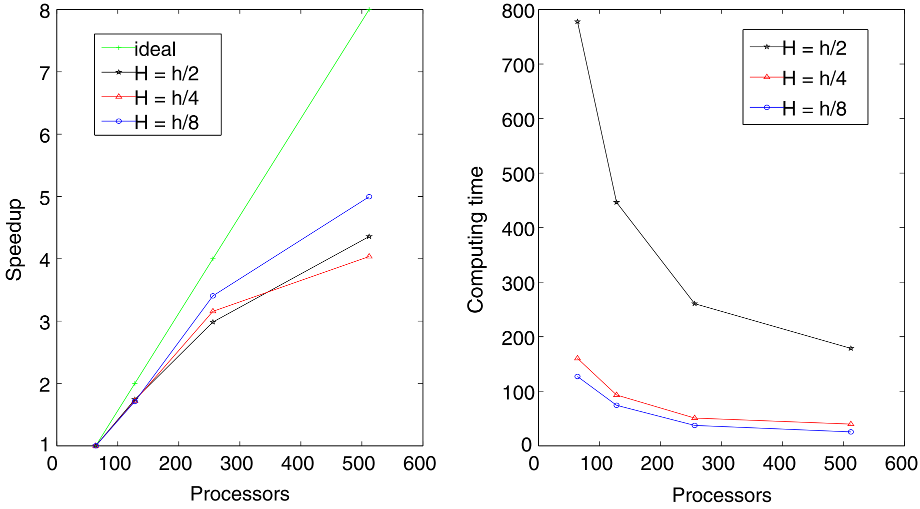
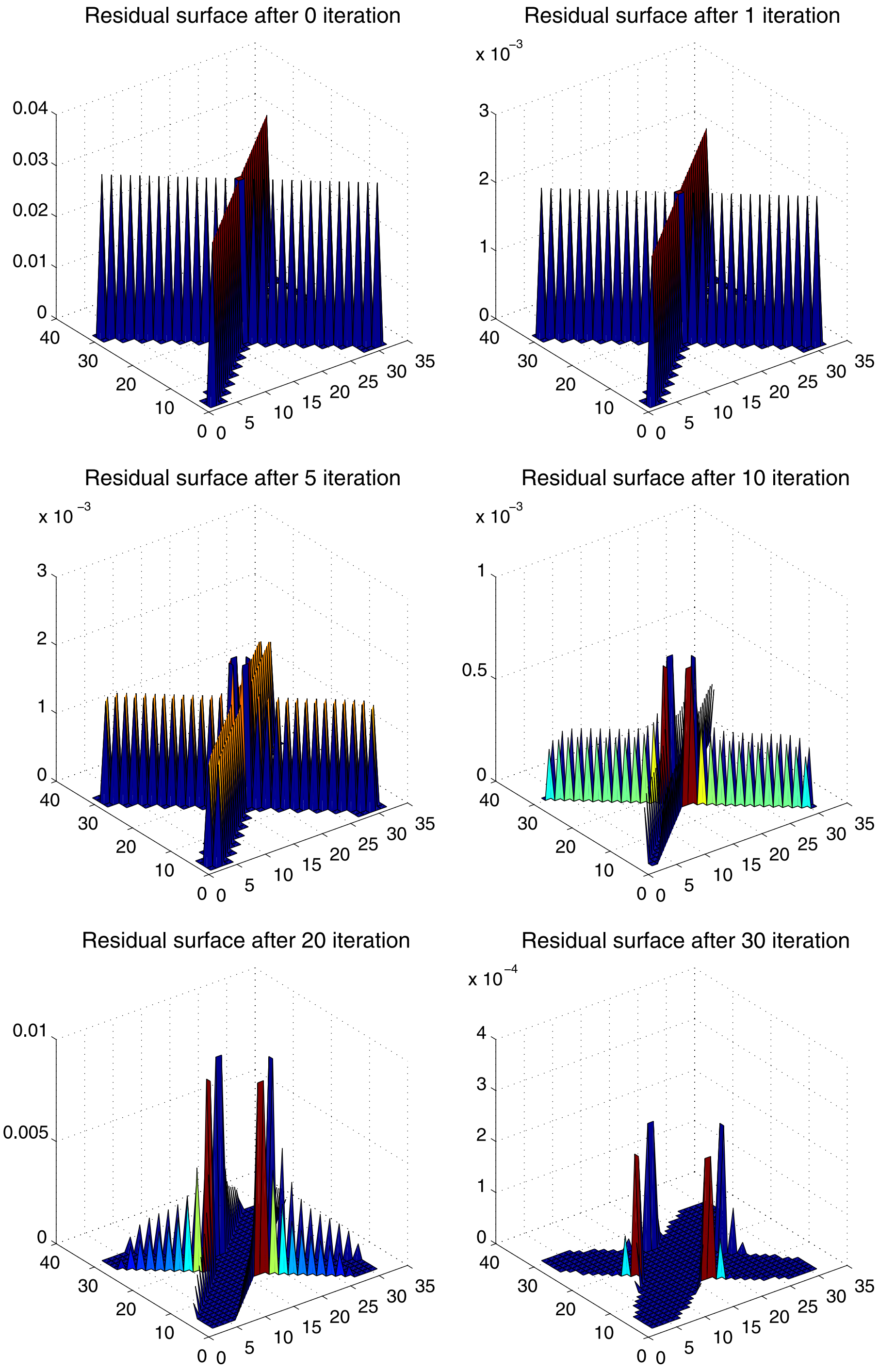


Fig. 5 The speedup and the total computing time of the two-level method with three different coarse grids for **Test 1**. The fine grid is 2048 × 2048. The overlapping sizes of the coarse grid and the fine grid are $\delta_c = 3$ and $\delta = 6$, respectively. The semismooth function is the minimum function. h and H are the fine and coarse grid sizes, respectively

the interface is given in the top-left figure of Fig. 2. In the one-level approach as shown in Fig. 6, all the large components of the residual are located near the computed interface of the elastic and plastic regions. From the top four figures, one can tell that the location is quite wrong even after 10 iterations. The correct interface location begins to show up only after 30 Newton iterations. For this particular experiment, it takes 42 Newton iterations to finally find the solution. In the two-level approach as shown in Fig. 7, because of the smoothed grid sequencing, a reasonably accurate location of the interface appears in the zeroth Newton iteration. With such a good initial guess, only 7 Newton iterations are required to satisfy the desired stopping condition.

4 Some Final Remarks

We developed a family of parallel, highly scalable, two-grid algorithms for solving general complementarity problems associated with linear and nonlinear partial differential equations. In addition to the fine grid, on which the PDE is discretized and the complementarity problem is solved, a coarse grid is introduced to accelerate the nonlinear convergence, and to precondition the linear Jacobian solver in a semismooth Newton iteration. With the help of a smoothed grid sequencing, a semismooth Newton method and a two-level restricted Schwarz preconditioner, we showed numerically that the family of two-grid Newton-Krylov-Schwarz algorithms has a fast and robust convergence and the rate of convergence is nearly independent of the number of unknowns of the problem and the number of processors. Surprisingly good results were obtained for solving some rather difficult obstacle problems with millions of unknowns and on parallel machines with up to 512 processors.



Residual surface after 0 iteration

Residual surface after 1 iteration

Residual surface after 5 iteration

Residual surface after 10 iteration

Residual surface after 20 iteration

Residual surface after 30 iteration

Fig. 6 The history for the residual surface of **Test 1** by using the one-level method. The grid is 128 × 128. The semismooth function is the Fischer-Burmeister function. In this case, 42 iterations are need to reach convergence

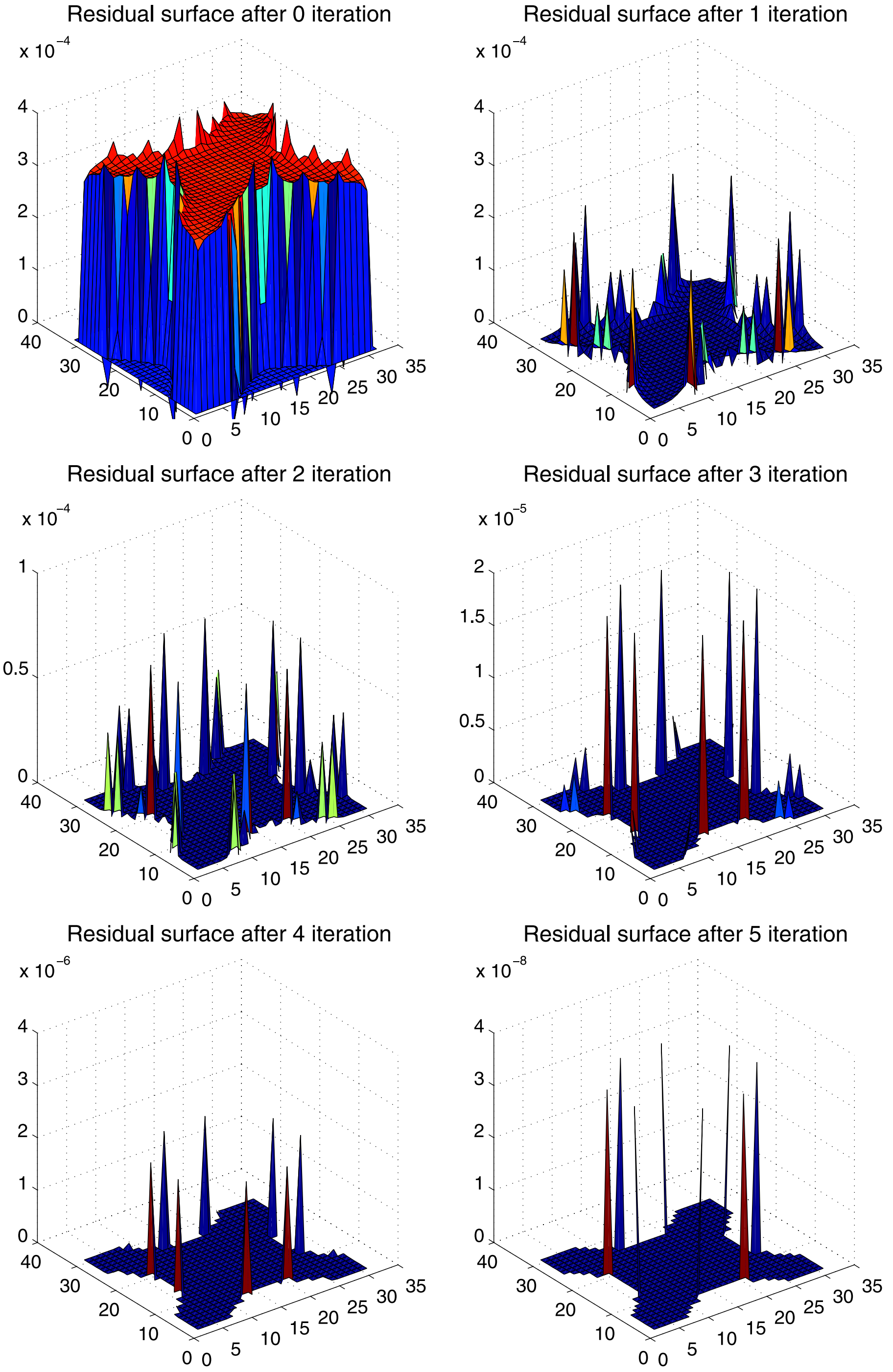


Fig. 7 The history for the residual surface of **Test 1** by using the two-level method. The semismooth function is the Fischer-Burmeister function. The fine grid is 128 × 128 and the coarse grid is 32 × 32. In this case, 7 iterations are need to reach convergence

References

1. Balay, S., Buschelman, K., Gropp, W.D., Kaushik, D., Knepley, M., McInnes, L.C., Smith, B.F., Zhang, H.: PETSc Users Manual. Argonne National Laboratory (2009)
2. Brown, P.N., Saad, Y.: Hybrid Krylov methods for nonlinear systems of equations. SIAM J. Sci. Stat. Comput. **11**, 297–271 (1990)
3. Brown, P.N., Saad, Y.: Convergence theory of nonlinear Newton-Krylov algorithms. SIAM J. Optim. **4**, 297–330 (1994)
4. Cai, X.-C., Gropp, W.D., Keyes, D.E., Melvin, R.G., Young, D.P.: Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equation. SIAM J. Sci. Comput. **19**, 246–265 (1998)
5. Cai, X.-C., Sarkis, M.: A restricted additive Schwarz preconditioner for general sparse linear systems. SIAM J. Sci. Comput. **21**, 792–797 (1999)
6. Cai, X.-C., Dryja, M., Sarkis, M.: Restricted additive Schwarz preconditioners with harmonic overlap for symmetric positive definite linear systems. SIAM J. Numer. Anal. **41**, 1209–1231 (2003)
7. Clarke, F.H.: Optimization and Nonsmooth Analysis. Wiley, New York (1983)
8. Cottle, R., Pang, J.-S., Stone, R.: The Linear Complementarity Problem. Academic Press, Boston (1992)
9. Luca, T.D., Facchinei, F., Kanzow, C.: A semismooth equation approach to the solution of nonlinear complementarity problems. Math. Program. **75**, 407–439 (1996)
10. Dennis, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. SIAM, Philadelphia (1996)
11. Eisenstat, S.C., Walker, H.F.: Globally convergent inexact Newton methods. SIAM J. Optim. **4**, 392–422 (1994)
12. Eisenstat, S.C., Walker, H.F.: Choosing the forcing terms in an inexact Newton method. SIAM J. Sci. Comput. **17**, 16–32 (1996)
13. Facchinei, F., Kanzow, C.: A nonsmooth inexact Newton method for the solution of large-scale nonlinear complementarity problems. Math. Program. **76**, 493–512 (1997)
14. Ferris, M.C., Kanzow, C.: Complementarity and related problems. In: Pardalos, P.M., Resende, M.G.C. (eds.) Handbook of Applied Optimization, pp. 514–530. Oxford University Press, New York (2002)
15. Ferris, M.C., Pang, J.-S.: Engineering and economic applications of complementarity problems. SIAM Rev. **39**, 669–713 (1997)
16. Fischer, A.: A special Newton-type optimization method. Optimization **24**, 269–284 (1992)
17. Forsyth, P.A., Vetzal, K.R.: Quadratic convergence for valuing American options using a penalty method. SIAM J. Sci. Comput. **23**, 2095–2122 (2002)
18. Frommer, A., Szyld, D.B.: An algebraic convergence theory for restricted additive Schwarz methods using weighted max norms. SIAM J. Numer. Anal. **39**, 463–479 (2001)
19. Gabriel, S.A., Pang, J.-S.: An inexact NE/SQP method for solving the nonlinear complementarity problem. Comput. Optim. Appl. **1**, 67–91 (1992)
20. Harker, P.T., Pang, J.-S.: Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications. Math. Program. **48**, 161–220 (1990)
21. Hintermüller, M., Ito, K., Kunish, K.: The primal-dual active set strategy as a semismooth Newton method. SIAM J. Optim. **13**, 865–888 (2003)
22. Jiang, H., Qi, L.: A new nonsmooth equations approach to nonlinear complementarity problems. SIAM J. Control Optim. **35**, 178–193 (1997)
23. Kanzow, C.: Inexact semismooth Newton methods for large-scale complementarity problems. Optim. Meth. Softw. **19**, 309–325 (2004)
24. Kärkkäinen, T., Kunisch, K., Tarvainen, P.: Augmented Lagrangian active set methods for obstacle problems. J. Optim. Theory Appl. **119**, 499–533 (2003)
25. Kornhuber, R.: Monotone multigrid methods for elliptic variational inequalities I. Numer. Math. **69**, 167–184 (1994)
26. Kornhuber, R.: On constrained Newton linearization and multigrid for variational inequalities. Numer. Math. **91**, 699–721 (2002)
27. Morales, J.L., Nocedal, J., Smelyanskiy, M.: An algorithm for the fast solution of symmetric linear complementarity problems. Numer. Math. **111**, 251–266 (2008)
28. Nabben, R., Szyld, D.B.: Convergence theory of restricted multiplicative Schwarz methods. SIAM J. Numer. Anal. **40**, 2318–2336 (2003)
29. Nochetto, R.H., Siebert, K.G., Veeser, A.: Pointwise a posteriori error control for elliptic obstacle problems. Numer. Math. **95**, 631–658 (2003)
30. Oosterlee, C.W.: On multigrid for linear complementarity problems with application to American-style options. Electron. Trans. Numer. Anal. **15**, 165–185 (2003)
31. Prudencio, E., Cai, X.-C.: Parallel multilevel restricted Schwarz preconditioners with pollution removing for PDE-constrained optimization. SIAM J. Sci. Comput. **29**, 964–985 (2007)

32. Qi, L.: Convergence analysis of some algorithms for solving nonsmooth equations. Math. Oper. Res. **18**, 227–244 (1993)
33. Rodrigues, J.-F.: Obstacle Problems in Mathematical Physics. North-Holland, Amsterdam (1987)
34. Saad, Y.: Iterative Methods for Sparse Linear Systems. SIAM, Philadelphia (2003)
35. Smith, B., Bjørstad, P., Gropp, W.: Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations. Cambridge University Press, Cambridge (1996)
36. Toselli, A., Widlund, O.: Domain Decomposition Methods-Algorithms and Theory. Springer, Berlin (2005)
37. Ulbrich, M.: Nonmonotone trust-region methods for bound-constrained semismooth equations with applications to nonlinear mixed complementarity problems. SIAM J. Optim. **11**, 889–917 (2001)
38. Zvan, R., Forsyth, P.A., Vetzal, K.R.: Penalty methods for American options with stochastic volatility. J. Comput. Appl. Math. **91**, 199–218 (1998)