Flexible Complementarity Solvers for Large-Scale Applications*

Steven J. Benson† Todd S. Munson‡

Submitted: July 17, 2003

Abstract

Discretizations of infinite-dimensional variational inequalities lead to linear and nonlinear complementarity problems with many degrees of freedom. To solve these problems in a parallel computing environment, we propose two active-set methods that solve only one linear system of equations per iteration. The linear solver, preconditioner, and matrix structures can be chosen by the user for a particular application to achieve high parallel performance. The parallel scalability of these methods is demonstrated for some discretizations of infinite-dimensional variational inequalities.

1 Introduction

Achieving high performance for numerical methods in parallel computing environments demands that the user have the ability to customize algorithms, linear solvers, and data structures for their particular problems. Closed environments or algorithms requiring specific linear algebra constrain the choices available to the user, inevitably leading to inefficiency. This paper concerns a flexible, open environment, the Toolkit for Advanced Optimization, and two algorithms for solving complementarity problems implemented so that the user can exploit problem structure. The benefits of this design are significant reductions in solution time and good parallel scalability on targeted complementarity problems.

Complementarity problems arise in various application areas (see, e.g., [22, 18]). In this paper, we are interested particularly in applications arising from infinite-dimensional box-constrained variational inequalities where a discretization of the problem corresponds to a large-scale linear or nonlinear complementarity problem [15, 25]. One example of this type, the journal bearing problem

*This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing, Office of Science, U.S. Department of Energy, under Contract W-31-109-Eng-38.

†Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois 60439, benson@mcs.anl.gov

‡Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois 60439, tmunson@mcs.anl.gov

[3], involves determining the pressure distribution of a thin film of lubricant between two circular cylinders. This problem is posed as an elliptic partial differential equation with a free boundary. A finite-difference scheme for solving instances of this problem has $n_x n_y$ degrees of freedom, where n_x and n_y are the number of points in the spatial discretizations. The large number of degrees of freedom and structure imposed by the finite-difference scheme implies that high performance can be achieved when solving this problem in a parallel environment. Other applications that can be posed as infinite-dimensional variational inequalities include pricing American options [23, 34], nonlinear obstacle problems [30], and optimal control problems [33].

The Toolkit for Advanced Optimization (TAO) provides a flexible environment for solving optimization and complementarity problems in parallel [8]. Several numerical linear algebra packages have been incorporated into TAO [4, 24, 16]. These packages offer many algorithmic choices so that a user can select the most appropriate method for a particular application. Novice users can rely on the provided defaults, while more advanced users have the freedom to select their problem representation and linear algebra to improve parallel performance. All of the source code for TAO can be downloaded from [7], which includes our implementations of several algorithms for solving complementarity problems.

Section 2 discusses the design philosophy of and facilities provided by the Toolkit for Advanced Optimization. The complementarity algorithms used to solve the discretized nonlinear complementarity problems in a parallel computing environment are discussed in Section 3. Two methods were implemented and tested: a semismooth method [11, 27] and a reduced space method similar to those presented in [14, 17]. These algorithms are attractive because they solve only a single system of linear equations per iteration. These solves can be performed by using preconditioned iterative methods [31], taking advantage of the research lavished on numerical linear algebra for solving partial differential equations. Numerical results on the MCPLIB collection [12] of complementarity problems are presented in Section 4 to demonstrate that the methods are reasonably robust on general problems. The parallel performance achieved on several discretizations of infinite-dimensional variational inequalities shows the benefits of customizing the linear algebra: both good parallel performance and scalability are achieved.

2 Design Philosophy

Traditionally, the numerical methods used in an algorithm implementation to solve linear systems of equations are chosen by the developer. Since most floating point operations are typically performed by the linear system solver, considerable effort has been expended to select and implement one that is efficient and robust for a diverse collection of applications. Efficiency for particular problem instances has been sacrificed in exchange for robustness and general applicability. However, linear solvers tailored to a particular application may

save a significant amount of computation and allow the method to find solutions to larger problems than previously possible. While developers choose a linear solver without any knowledge of the application, many applications have structure that can be exploited. The Toolkit for Advanced Optimization (TAO) [7, 9, 8] was specifically designed to allow the user to customize the linear solver to their application so that they can achieve high performance and parallel scalability.

One linear solver choice is an LU factorization, which can be applied to arbitrary nonsingular systems. Many robust implementations have been developed that utilize sparsity while maintaining numerical stability. The complementarity methods in TAO allow for the use of the LU factorizations from LAPACK [1] for dense systems and LUSOL [28] for sparse systems. Nonetheless, these factorizations do not exploit symmetry, strong monotonicity, or other features of the matrix, such as block structure. Furthermore, direct factorizations can impose excessive memory requirements even when the given matrix is sparse. Access to Cholesky factorizations are provided when a symmetric positive definite linear system is solved. In this case, the number of floating-point operations can be reduced by a factor of two, and further gains can be achieved by stable reorderings of the matrix based solely on the sparsity pattern.

Aside from direct methods, many iterative techniques can be applied to solve linear systems of equations. These techniques typically do not impose the memory requirements of direct methods. Two of the most common techniques are GMRES [32] for general nonsingular matrices and the conjugate gradient method [20] for symmetric positive definite matrices. Convergence of these methods can be significantly improved by using a preconditioner, such as an incomplete factorization, which works well on general problems, or an application-specific preconditioner, such as an overlapping additive Schwartz method [5].

In a parallel environment, the cost of message passing magnifies the importance of an appropriate selection of the linear system solver and preconditioner. Parallel implementations of GMRES, conjugate gradients, and many other iterative methods, along with scalable preconditioners such as incomplete factorizations and additive Schwartz methods, are provided in the PETSc toolkit [5, 6]. These linear system solvers use the Message Passing Interface (MPI) [21] for communication between processors.

Providing the user with sufficient flexibility in the choice of linear solver requires careful selection and implementation of the complementarity algorithms. In particular, the interface to the numerical objects, iterative methods and linear algebra, must be separated from their implementations. Object-oriented techniques permit the use of serial or parallel data structures with minimal changes to the interface and allow new implementations of linear system solvers and associated linear algebra to be incorporated. TAO was designed to enable this flexibility and leverages many of the existing parallel linear algebra tools, such as HYPRE [24], PETSc, and Trilinos [16]. The optimization methods in TAO, which include the complementarity methods and solvers to minimize an objective function with bounds on the variables, are written to scale on parallel

machines and expose many of the algorithmic details, such as the linear solver, to the user. This design philosophy assumes that the user has knowledge of an application that can be used to improve the parallel performance.

3 Complementarity Algorithms

The finite-dimensional nonlinear complementarity problem defined by a given function $F : \Re^n \to \Re^n$ is to calculate an $x^* \in \Re^n$ such that $x^* \geq 0$, $F(x^*) \geq 0$, and $\langle x^*, F(x^*) \rangle = 0$. The algorithms implemented in this study use the current iterate to define an active set, solve a reduced system to calculate a direction, and then perform a line search to compute a new iterate that sufficiently decreases a merit function.

3.1 Active-Set Semismooth Method

The semismooth algorithm reformulates a given complementarity problem as a nonsmooth system of equations satisfying a semismooth property by using an NCP function. The resulting nonsmooth system of equations is solved with Newton's method where an element of the B-subdifferential plays the role of the Jacobian matrix in the direction calculation. The active-set semismooth method implemented in TAO is based on this idea.

Mathematically, a function $\phi : \Re^2 \to \Re$ is said to be an NCP function if $\phi(a, b) = 0$ if and only if $a \geq 0$, $b \geq 0$, and $ab = 0$. By defining

$$\Phi(x) := \begin{bmatrix} \phi(x_1, F_1(x)) \\ \phi(x_2, F_2(x)) \\ \vdots \\ \phi(x_n, F_n(x)) \end{bmatrix}$$

for any NCP function ϕ, the nonlinear complementarity problem can be reformulated as finding an $x^* \in \Re^n$ such that $\Phi(x^*) = 0$. In order to globalize a Newton method for solving this system of equations, the merit function

$$\Psi(x) := \frac{1}{2} \|\Phi(x)\|_2^2$$

is typically used in a line search.

The Fischer-Burmeister function [19],

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

is one NCP function, where Φ_{FB} denotes the reformulation of the nonlinear complementarity problem implied by using ϕ_{FB}. Even though Φ_{FB} is not continuously differentiable, a Newton method can still be constructed for finding $\Phi_{FB}(x) = 0$.

The function Φ_{FB} is differentiable almost everywhere and therefore admits a B-subdifferential [29]

$$\partial_B \Phi_{FB}(x) := \left\{ H \in \Re^{n\times n} \mid \exists \left\{x^k\right\} \subseteq D_{\Phi_{FB}} \text{ with } \lim_{x^k \to x} \nabla \Phi_{FB}(x^k) = H \right\},$$

where $D_{\Phi_{FB}}$ denotes the set of points where Φ_{FB} is differentiable. In particular,

$$\partial_B \Phi_{FB}(x) \subseteq \{D_a(x) + D_b(x)\nabla F(x)\}$$

for nonnegative diagonal matrices $D_a(x)$ and $D_b(x)$ defined componentwise as follows [17]:

(a) If $\|(x_i, F_i(x))\|_2 > 0$, then

$$\begin{aligned} [D_a(x)]_{i,i} &:= 1 - \frac{x_i}{\|(x_i, F_i(x))\|_2} \\ [D_b(x)]_{i,i} &:= 1 - \frac{F_i(x)}{\|(x_i, F_i(x))\|_2}. \end{aligned}$$

(b) Otherwise

$$\left([D_a(x)]_{i,i}, [D_b(x)]_{i,i}\right) \in \{(1-\alpha, 1-\beta) \mid \|(\alpha,\beta)\|_2 \leq 1\}.$$

Furthermore, the merit function, $\Psi_{FB}(x) := \frac{1}{2}\|\Phi_{FB}(x)\|_2^2$ is continuously differentiable with $\nabla \Psi_{FB}(x) = H^T \Phi_{FB}(x)$ for any $H \in \partial_B \Phi_{FB}(x)$.

The main computational task in a semismooth Newton method [11] for solving the nonsmooth system of equations $\Phi_{FB}(x) = 0$ is to calculate a direction by solving the linear system of equations

$$H^k d^k = -\Phi_{FB}(x^k),$$

where H^k is any element of $\partial_B \Phi_{FB}(x^k)$. An Armijo line search [2] along this direction is then applied to obtain a new iterate that sufficiently decreases the merit function. When the full system is solved to find the Newton direction, one must use a method suitable for nonsymmetric matrices because of the row scaling implied by the characterization of the B-subdifferential. Alternatively, one can solve a reduced system, where only a nonnegative diagonal perturbation is made to $\nabla F(x)$. This reduced system will be symmetric whenever $\nabla F(x)$ is symmetric, removing the restrictions placed on the linear solver for the full-space system.

The reduced system is obtained by selecting active and inactive sets of constraints. For a fixed $0 \leq \epsilon < 1$, define

$$\begin{aligned} \mathcal{A}(x) &:= \{i \in \{1,\ldots,n\} \mid [D_b(x)]_{i,i} \leq \epsilon\} \\ \mathcal{I}(x) &:= \{i \in \{1,\ldots,n\} \mid [D_b(x)]_{i,i} > \epsilon\}, \end{aligned}$$

where $\mathcal{A}(x)$ denotes the active constraints at x and $\mathcal{I}(x)$ the inactive constraints. The characterization of the B-subdifferential can be used to show that

$$i \in \mathcal{A}(x) \;\Rightarrow\; x_i \leq \kappa F_i(x)$$

for some $\kappa > 0$. The latter characterization is used by the active-set method in in [17], where they show that for all x in a sufficiently small neighborhood of x^*, the strongly active components are correctly identified.

We are now ready to state our active-set semismooth algorithm.

Algorithm 3.1 *Active-Set Semismooth Method*

1. *Let $F : \Re^n \to \Re^n$, $x^0 \in \Re^n$, $\epsilon \in [0,1)$, $\rho > 0$, $p > 2$, $\beta < 1$, and $\sigma \in (0, \frac{1}{2})$ be given. Set $k = 0$.*

2. *If $\|\Phi(x^k)\|_2 \leq tol$, then stop.*

3. *Otherwise, choose $D_a(x^k)$ and $D_b(x^k)$ so that*

$$D_a(x^k) + D_b(x^k)\nabla F(x^k) \in \partial_B \Phi_{FB}(x^k),$$

and determine $\mathcal{A}^k := \mathcal{A}(x^k)$ and $\mathcal{I}^k := \mathcal{I}(x^k)$.

4. *Let*

$$d^k_{\mathcal{A}^k} = -[D_a(x^k)]^{-1}_{\mathcal{A}^k,\mathcal{A}^k}\Phi_{FB}(x^k)_{\mathcal{A}^k},$$

and approximately solve the reduced system

$$\left([D_b(x^k)]^{-1}_{\mathcal{I}^k,\mathcal{I}^k}[D_a(x^k)]_{\mathcal{I}^k,\mathcal{I}^k} + [\nabla F(x^k)]_{\mathcal{I}^k,\mathcal{I}^k}\right) d^k_{\mathcal{I}^k} = \\ -[D_b(x^k)]^{-1}_{\mathcal{I}^k,\mathcal{I}^k}\Phi_{FB}(x^k)_{\mathcal{I}^k} - [\nabla F(x^k)]_{\mathcal{I}^k,\mathcal{A}^k} d^k_{\mathcal{A}^k}$$

to find $d^k_{\mathcal{I}^k}$.

5. *If the descent test*

$$\nabla \Psi_{FB}(x^k)^T d^k \leq -\rho \|d^k\|_2^p$$

is not satisfied, set $d^k = -\nabla \Psi_{FB}(x^k)$.

6. *Calculate the smallest $i \in \{0, 1, \ldots\}$ such that*

$$\Psi_{FB}(x^k + \beta^i d^k) \leq \Psi_{FB}(x^k) + \sigma \beta^k \nabla \Psi_{FB}(x^k)^T d^k,$$

and set $x^{k+1} = x^k + \beta^i d^k$.

7. *Set $k = k + 1$, and go to Step 2.*

The advantages of this active-set method are that the linear systems of equations solved to calculate the Newton directions are of a reduced size and they remain symmetric, positive definite if $\nabla F(x)$ is symmetric, positive definite. Therefore, the choice of preconditioner and iterative method is not restricted to nonsymmetric methods.

The implementation of the active-set method in TAO uses $\rho = 10^{-10}$, $p = 2.1$, $\beta = \frac{1}{2}$, and $\sigma = 10^{-4}$. The value of ϵ is dynamically chosen as

$$\epsilon(x^k) = \frac{\min\left\{\frac{1}{2}\|\Phi(x^k)\|_2^2, 10^{-2}\right\}}{1 + \|\nabla F(x^k)\|_1}.$$

This choice forces the active-set identification to go to zero as we approach a solution to the complementarity problem and deals with possible scaling problems with the Jacobian of F, since small values for $[D_b]_{i,i}$ can still have an effect on the direction calculation when the Jacobian is poorly scaled. For mixed complementarity problems where we have both finite lower and upper bounds, the Billups formulation is used [10]. The same active-set identification technique is used with this formulation.

3.2 Reduced-Space Method

The reduced-space method implemented in TAO also selects an active set and solves a reduced linear system of equations to calculate a direction. The iterates in this method remain within the variable bounds, while no such guarantee is made with the active-set semismooth method. This method is motivated by the simplicity of implementation in a distributed memory computing environment and the computational efficiency of similar methods observed on several classes of problems [14].

The active and inactive sets used within the reduced-space method are defined as

$$\begin{aligned} \mathcal{A}(x) &:= \{i \in \{1, \ldots, n\} \mid x_i = 0 \text{ and } F_i(x) > 0\} \\ \mathcal{I}(x) &:= \{i \in \{1, \ldots, n\} \mid x_i > 0 \text{ or } F_i(x) \leq 0\}. \end{aligned}$$

The active set denotes the variables where the lower bound is active and the function value can be ignored. The inactive set contains the remainder of the variables. At every iteration of the reduced-space method a direction is calculated by approximately solving the linear system equation

$$\left[\nabla F(x^k)\right]_{\mathcal{I}^k, \mathcal{I}^k} d_{\mathcal{I}^k} = -F_{\mathcal{I}^k}(x^k)$$

and setting $d_{\mathcal{A}^k}$ to zero.

A projected line search is then applied to generate the next iterate x^{k+1} such that

$$x^{k+1} = \pi[x^k + \alpha d^k], \tag{1}$$

where π is the projection onto the variable bounds. The step size α is chosen such that $\|F_\Omega(\pi[x^k + \alpha d^k])\|_2 \leq (1 - \sigma\alpha)\|F_\Omega(x)\|_2$, where $F_\Omega(x)$ is defined component-wise by

$$[F_\Omega(x)]_i = \begin{cases} F_i(x) & \text{if} \quad x_i > 0 \\ \min\{F_i(x), 0\} & \text{if} \quad x_i = 0. \end{cases} \tag{2}$$

The line search tries step lengths $\alpha = \beta^j$ for $\beta \in (0,1)$ and positive integers j such that $\beta^j > \gamma$. There are no guarantees the direction calculated is a descent direction, so the line search can terminate with either a new point of sufficient improvement or a minimum step length. The implementation uses $\sigma = 10^{-4}$, $\beta = 0.5$, and a minimum stepsize of $\gamma = 10^{-12}$ for the parameter choices.

If the line search fails to identify such a point, this method performs the same line search in the direction $d^k = -F(x^k)$. The algorithm continues until there is a second failure in the line search or a stationary point has been found.

The complete reduced-space algorithm follows.

Algorithm 3.2 *Active-Set Reduced Space Method*

1. *Let $F : \Re^n \to \Re^n$ and $x^0 \in \Re^n_+$ be given, and set $k = 0$.*
2. *If $\|F_\Omega(x^k)\|_2 \leq tol$, then stop.*
3. *Let*

$$\begin{aligned} \mathcal{A}(x) &:= \{i \in \{1,\ldots,n\} \mid x_i = 0 \text{ and } F_i(x) > 0\} \\ \mathcal{I}(x) &:= \{i \in \{1,\ldots,n\} \mid x_i > 0 \text{ or } F_i(x) \leq 0\}, \end{aligned}$$

   set $d_{\mathcal{A}^k} = 0$, and approximately solve the linear system

$$\left[\nabla F(x^k)\right]_{\mathcal{I}^k,\mathcal{I}^k} d_{\mathcal{I}^k} = -F_{\mathcal{I}^k}(x^k)$$

   to calculate a direction.

4. *If possible, calculate the smallest $i \in \{0,1,\ldots,\bar{i}\}$ such that*

$$\|F_\Omega(\pi[x^k + \beta^i d^k])\|_2 \leq (1 - \sigma\beta^i)\|F_\Omega(x^k)\|_2.$$

5. *Otherwise, set $d^k = -F(x^k)$, and apply the line search. If no such i is found providing sufficient decrease, then stop.*
6. *Set $x^{k+1} = \pi[x^k + \beta^i d^k]$ and $k = k+1$, and go to step 2.*

Although no convergence results have been proven for the reduced-space method, it has been demonstrated to be effective, especially on monotone applications. The reduced matrix retains symmetry and positive definiteness when they exist in the Jacobian, permitting the use of a symmetric linear solver. Furthermore, this method lends itself to a parallel implementation because it requires only a few numerical operations other than a linear solver.

4 Computational Results

The complementarity methods were implemented in and are distributed with the Toolkit for Advanced Optimization. Two series of tests were performed on the implementations: one to check the robustness to verify that the methods work

on a significant number of general problems, and the other to test the parallel performance and scalability of the methods on some trial infinite-dimensional variational inequalities. The latter tests also validate the merits of our design philosophy by showing that the customization of the linear system solver does have a significant effect on the overall solution time. For all of these tests, an upper limit of 100 linear system solves was placed on the methods.

4.1 Robustness

We ran the implemented methods on the complementarity problems contained in the MCPLIB collection [12] to demonstrate robustness on a diverse set of problems, including many small complementarity problems and some nonlinear obstacle problems and optimal control problems. These computational tests were performed on Linux workstation containing a Pentium 4 processor with a clock speed of 1.8 GHz and 512 MB of RAM.

In order to test the robustness, we ran the methods with LUSOL as the selected linear system solver. This LU factorization routine is used by several optimization and complementarity packages, including MINOS [28] and PATH [13], and is known to be robust and efficient in a serial environment. We chose to use a direct factorization for this test to eliminate possible problems with the choice of preconditioner, iterative method, and termination tolerances for the inexact linear system solves.

Tables 1 and 2 report the number of successes and failures for each method and model in the test set when using LUSOL. These results indicate that the active-set semismooth method, which solves 73.7% of the problems, is more robust than the reduced-space method, which solves 65.5% of the problems. The overall conclusion to be drawn is that both methods solve a significant fraction of these test problems. While these robustness results can be improved by introducing heuristics into the implementations, we prefer to use the standard methods because they are less complicated and work well on the targeted infinite-dimensional variational inequalities.

4.2 Scalability

To demonstrate the scalability and performance gains that can be made by selecting appropriate linear system solvers, we tested the complementarity algorithms implemented on some discretizations of infinite-dimensional variational inequalities. All of the computational tests in this subsection were performed on a Linux cluster composed of 350 Pentium Xeon processors with a clock speed of 2.4 GHz. Each node has a minimum of 1 GB of RAM is connected by a Myrinet 2000 network.

One benchmark application is the journal bearing model, a variational problem over a two-dimensional region. This problem arises in the determination of the pressure distribution in a thin film of lubricant between two circular cylinders. The infinite-dimensional version of this problem is to find a piecewise

Table 1: Performance of methods on MCPLIB.

	Active-Set Method		Reduced-Space Method	
Problem	Successes	Failures	Successes	Failures
ahn	1	0	1	0
badfree	1	0	1	0
baihaung	1	0	1	0
bert_oc	4	0	4	0
bertsekas	6	0	3	3
billups	0	3	0	3
bishop	0	1	0	1
bratu	0	1	1	0
cammcf	0	1	0	1
cgereg	2	20	14	8
choi	1	0	1	0
colvdual	2	2	2	2
colvnlp	6	0	6	0
cycle	1	0	1	0
danny	6	2	6	2
degen	1	0	1	0
dirkse	1	1	1	1
duopoly	0	1	0	1
eckstein	1	0	0	1
ehl	2	10	6	6
electric	0	1	0	1
explcp	1	0	0	1
exros	1	4	1	4
ferrralph	2	0	2	0
finance	60	0	59	1
fixedpt	0	2	0	2
force	0	2	0	2
freebert	3	4	5	2
fried	5	5	6	4
friedms	1	0	1	0
gafni	3	0	3	0
games	18	7	0	25
golanmcp	0	1	1	0
hanskoop	7	3	5	5
hydroc	2	0	2	0
jel	2	0	2	0
jiangqi	3	0	3	0
josephy	8	0	5	3
kanzow	7	0	7	0
kojshin	8	0	5	3
kyh	0	4	0	4
lincont	0	1	0	1
lstest	0	1	0	1
leyffer	1	0	1	0

Table 2: Performance of methods on MCPLIB.

Problem	Active-Set Method Successes	Active-Set Method Failures	Reduced-Space Method Successes	Reduced-Space Method Failures
mathi	13	0	13	0
methan	1	0	1	0
mr5mcf	0	1	0	1
munson	3	1	3	1
nash	4	0	4	0
ne-hard	0	1	1	0
obstacle	8	0	8	0
optcont	5	0	5	0
pgvon	0	12	2	10
pies	1	0	0	1
pizer	1	3	3	1
powell	5	1	3	3
powellmcp	6	0	6	0
poz	6	0	6	0
qp	1	0	1	0
runge	3	4	0	7
scarf	9	3	6	6
shansim	1	0	1	0
shubik	11	37	10	38
simple-ex	0	1	0	1
simple-red	1	0	1	0
spillmcp	0	1	0	1
sppe	2	1	3	0
sun	1	0	1	0
taji	12	0	12	0
tiebout	0	6	0	6
tinloi	64	0	63	1
tinsmall	63	1	63	1
titan	1	1	1	1
tobin	4	0	4	0
tqbilat	1	1	2	0
trafelas	0	2	0	2
trig	2	1	0	3
vonthmcf	0	1	0	1
xiaohar	4	0	3	1
xu	35	0	5	30
Total	437	156	391	202

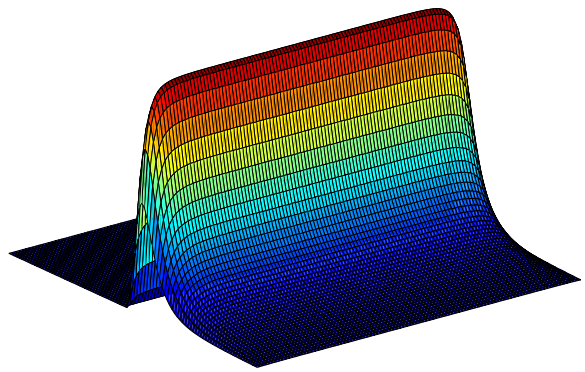


Figure 1: The journal bearing problem with $\varepsilon = 0.9$.

continuously differentiable function, $v : \mathcal{D} \mapsto \mathbb{R}$, such that

$$0 \leq v, \; w_q \Delta v \geq w_l, \quad \text{and } v\,[w_q \Delta v - w_l] = 0$$

almost everywhere on $\mathcal{D}$ with $v = 0$ on $\partial\mathcal{D}$, where $\mathcal{D} = (0, 2\pi) \times (0, 2b)$ for some constant $b > 0$ and

$$w_q(\xi_1, \xi_2) = (1 + \varepsilon \cos \xi_1)^3, \quad w_l(\xi_1, \xi_2) = \varepsilon \sin \xi_1$$

with ε in $(0, 1)$. The eccentricity parameter, ε, influences, in particular, the difficulty of the problem. Figure 1 shows the solution of the journal bearing problem for $\varepsilon = 0.9$. The steep gradient in the solution makes this problem a difficult benchmark. Other elliptic problems tested include the obstacle, elastic-plastic torsion, and combustion models from MINPACK-2; see [3] for a complete description of these models.

Discretization of the journal bearing problem with either finite differences or finite elements leads to a standard complementarity problem. The number of variables is $n = n_x n_y$, where n_x and n_y are, respectively, the number of grid points in each coordinate direction of the domain $\mathcal{D}$. See for [26] a description of the finite-element discretization.

The number of grid points can be very large. In fact, these problems become so large that the Jacobian matrix cannot even be stored on one machine. Direct factorizations exacerbate the computer memory issues because the fill-in associated with a direct factorization is significant. For this reason, iterative linear solvers are necessary.

The Jacobian of this problem has a sparse, symmetric, and positive definite structure. Considerable improvement can be made by using the conjugate gradient method to solve the linear systems with an incomplete factorization used as a

preconditioner. Many incomplete factorization preconditioners require nonzeros along the diagonal, and the journal bearing problem satisfies this requirement.

The problem was discretized into 40,000 variables and then solved with three different solvers: LUSOL, a conjugate gradient method with an incomplete LU preconditioner, and a conjugate gradient method with a block Jacobi preconditioner such that each block contains an incomplete LU factorization. As Table 3 shows, use of the conjugate gradient methods with with an appropriate preconditioner saves a significant amount of time. With the reduced-space method, the savings range from 30% on the combustion problem to over 70% on the elastic-plastic torsion problem.

Table 3: Solution times for three linear solvers (sec).

Problem	LUSOL	CG - 1 P	CG - 2 P
J Bearing	278	136	99
EP Torsion	172	50	34.6
Obstacle	49	18.4	14.3
Combustion	29.8	20.2	14.1

To demonstrate the parallel efficiency of the complementarity methods, we wrote parallel implementations of the elastic-plastic torsion and obstacle problems using the grid management facilities of PETSc, which relies on MPI [21] for communications between processors. PETSc provides support for discretizing the rectangular region $\mathcal{D}$, partitioning the surface into multiple regions, and assigning each processor to one of these regions. Each processor computes the function on its subdomain with respect to the variables in its region. A preconditioned conjugate gradient method was used to solve the linear systems generated by the complementarity algorithms. The preconditioner was a block Jacobi preconditioner with incomplete LU factorization. In the reduced-space method, the relative tolerance used during the linear solve was 0.01. The results are summarized in Table 4 when using 1–64 processors.

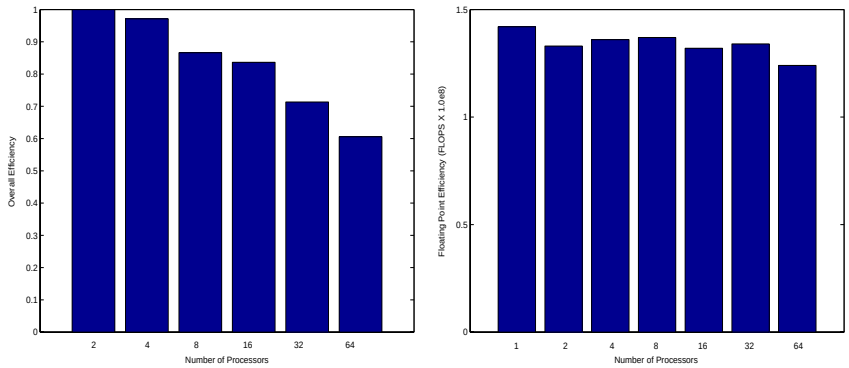
The overall efficiency of our implementation is shown in Figure 2(a). Each bar indicates the performance of the active-set semismooth method relative to its performance on two processors. This number is the ratio of two times the execution time using two processors and the number of processors multiplied by the time needed to solve the problem using one of those processors. With this metric, the overall parallel efficiency active-set semismooth method on the obstacle problems using 16 processors was over 83%, and the overall parallel efficiency using 64 processors was over 60%. The overall efficiency of the reduced-space method was similar.

A second set of tests was performed with the mesh refined according the number of processors used to solve the problem. In these tests each processor owned $10,000$ variables. The mesh was 100×100 when one processor was used and 800×800 when 64 processors were used. Since the methods require more iterations to solve problems with finer meshes, a comparison of running times

Table 4: Scalability.

Problem	Solver	mx	my	Processors						
				1	2	4	8	16	32	64
Obstacle	R-S	800	800	2848	1722	887	484	252	129	71
Obstacle	A-S	800	800	975	582	300	168	87	51	30
JBearing	R-S	800	800	13597	7786	7021	3555	1861	930	544
JBearing	A-S	800	800	14025	7770	7019	3540	1973	1082	648
Eptorsion	R-S	800	800	835	548	287	152	112	58	41
Eptorsion	A-S	800	800	4744	3118	1644	877	463	261	165
Combustion	R-S	800	800	332	228	122	63	31	15	8.6
Combustion	A-S	800	800	339	233	124	67	32	17	10.2

would not demonstrate the scalability of the methods. Instead, we used the rate of floating-point operations as the measure of efficiency. Figure 2(b) shows the average number of floating-point operations performed per second on each processor when the reduced space method was used to solve the torsion problem. As the overhead of message passing increases, the rate of computation on each processor decreases. The problem used over 142 MFlops when one processor was used and that rate only decreased to about 124 MFlops when when 64 processors were used. Our parallel efficiency of both methods by this measure often exceeded 80% on sixty four processors.



(a) Overall Implementation Efficiency (b) Floating-Point Efficiency

Figure 2: Parallel efficiency on the obstacle problem.

5 Conclusion

The complementarity methods implemented within the Toolkit for Advanced Optimization are reasonably robust and can effectively solve discretized versions of infinite-dimensional variational inequalities. In order to achieve high parallel performance, the methods were implemented so that the user has flexibility in the choice of data structures, linear algebra, and algorithms.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Cros, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK User's Guide, second ed.* SIAM, Philadelphia, 1995.

[2] L. Armijo. Minimization of functions having Lipschitz-continuous first partial derivatives. *Pacific Journal of Mathematics*, 16:1–3, 1966.

[3] B. M. Averick, R. G. Carter, J. J. Moré, and G. L. Xue. The MINPACK-2 test problem collection. Technical Report MCS-P153-0692, Argonne National Laboratory, Argonne, Illinois, 1992.

[4] S. Balay, W. Gropp, L. Curfman-McInnes, and B. Smith. PETSc Web page. See `http://www.mcs.anl.gov/petsc`.

[5] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhauser Press, 1997.

[6] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. PETSc 2.0 users manual. Technical Report ANL-95/11 - Revision 2.1.0, Argonne National Laboratory, April 2001.

[7] Steve Benson, Lois Curfman McInnes, and Jorge Moré. Toolkit for Advanced Optimization (TAO) Web page. See `http://www.mcs.anl.gov/tao`.

[8] Steve Benson, Lois Curfman McInnes, and Jorge Moré. TAO users manual. Technical Report ANL/MCS-TM-242, Mathematics and Computer Science Division, Argonne National Laboratory, 2000. See `http://www.mcs.anl.gov/tao`.

[9] Steven J. Benson, Lois Curfman McInnes, and Jorge J. More'. A case study in the performance and scalability of optimization algorithms. *ACM Transactions on Mathematical Software*, 27(3):361–376, September 2001.

[10] S. C. Billups. *Algorithms for Complementarity Problems and Generalized Equations.* PhD thesis, University of Wisconsin, Madison, 1995.

[11] T. De Luca, F. Facchinei, and C. Kanzow. A semismooth equation approach to the solution of nonlinear complementarity problems. *Mathematical Programming*, 75:407–439, 1996.

[12] S. P. Dirkse and M. C. Ferris. MCPLIB: A collection of nonlinear mixed complementarity problems. *Optimization Methods and Software*, 5:319–345, 1995.

[13] S. P. Dirkse and M. C. Ferris. The PATH solver: A non-monotone stabilization scheme for mixed complementarity problems. *Optimization Methods and Software*, 5:123–156, 1995.

[14] S. P. Dirkse and M. C. Ferris. Crash techniques for large-scale complementarity problems. In M. C. Ferris and J. S. Pang, editors, *Complementarity and Variational Problems: State of the Art*, pages 40–61, Philadelphia, 1997. SIAM.

[15] I. Ekeland and R. Témam. *Convex Analysis and Variational Problems.* North–Holland, Amsterdam, 1976.

[16] Mike Heroux et al. Trilinos Web page. See `http://www.cs.sandia.gov/\~mheroux/Trilinos/doc/TrilinosIntroduction.ht%m`.

[17] F. Facchinei and J. Soares. A new merit function for nonlinear complementarity problems and a related algorithm. *SIAM Journal on Optimization*, 7:225–247, 1997.

[18] M. C. Ferris and J. S. Pang. Engineering and economic applications of complementarity problems. *SIAM Review*, 39:669–713, 1997.

[19] A. Fischer. A special Newton-type optimization method. *Optimization*, 24:269–284, 1992.

[20] G. H. Golub and C. F. Van Loan. *Matrix Computations, third ed.* The John Hopkins University Press, Baltimore, Maryland, 1996.

[21] William Gropp, Ewing Lusk, and Anthony Skjellum. *Using MPI: Portable Parallel Programming with the Message Passing Interface.* MIT Press, 1994.

[22] P. T. Harker and J. S. Pang. Finite–dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications. *Mathematical Programming*, 48:161–220, 1990.

[23] J. Huang and J. S. Pang. Option pricing and linear complementarity. *Journal of Computational Finance*, 2:31–60, 1998.

[24] *hypre*: High performance preconditioners. http://www.llnl.gov/CASC/hypre/.

[25] D. Kinderlehrer and G. Stampacchia. *An Introduction to Variational Inequalities and Their Applications.* Academic Press, New York, 1980.

[26] J. J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1:93–113, 1991.

[27] T. S. Munson, F. Facchinei, M. C. Ferris, A. Fischer, and C. Kanzow. The semismooth algorithm for large scale complementarity problems. *INFORMS Journal on Computing*, 13:294–311, 2001.

[28] B. A. Murtagh and M. A. Saunders. MINOS 5.0 user's guide. Tech. Rep. SOL 83.20, Stanford University, Stanford, California, 1983.

[29] L. Qi. Convergence analysis of some algorithms for solving nonsmooth equations. *Mathematics of Operations Research*, 18:227–244, 1993.

[30] J. F. Rodrigues. *Obstacle Problems in Mathematical Physics.* Elsevier Publishing Company, Amsterdam, 1987.

[31] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publishing Company, Boston, 1996.

[32] Y. Saad and M. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 44:856–869, 1986.

[33] M. Ulbrich. On a nonsmooth Newton method for nonlinear complementarity problems in function space with applications in optimal control. In M. C. Ferris, O.L. Mangasarian, and J. S. Pang, editors, *Complementarity: Applications, Algorithms and Extensions*, volume 50 of *Applied Optimization*, pages 341–360, Dordrecht, The Netherlands, 2001. Kluwer Academic Publishers.

[34] P. Wilmott, J. Dewynne, and S. Howison. *Option Pricing.* Oxford Financial Press, Oxford, England, 1993.