

A preconditioner for solving large scale variational inequality problems

Un preconditionatore per risolvere disequazioni variazionali di grandi dimensioni

Valeria Ruggiero

Dip.di Matematica, Università di Ferrara

Via Saragat 1, Ferrara

`rgv@unife.it`

Federica Tinti

Dip.di Matematica Pura ed Applicata, Università di Modena

Via Campi 213/B, Modena

`tntfrc@unife.it`

We consider the classical variational inequality problem, denoted by $VIP(F,C)$, which is to find $x^* \in C$ such that

$$(0.1) \quad \langle F(x^*), x - x^* \rangle \geq 0, \forall x \in C$$

where C is a nonempty closed convex subset of \mathfrak{R}^n , $\langle \cdot, \cdot \rangle$ the usual inner product in \mathfrak{R}^n and $F : \mathfrak{R}^n \rightarrow \mathfrak{R}^n$ is a continuous function.

We assume that the feasible set C can be represented as follows

$$(0.2) \quad C = \{x \in \mathfrak{R}^n | h(x) = 0, g(x) \geq 0, \Pi_l x \geq l, \Pi_u x \leq u\},$$

where $h : \mathfrak{R}^n \rightarrow \mathfrak{R}^p$, $g : \mathfrak{R}^n \rightarrow \mathfrak{R}^m$, $\Pi_l \in \mathfrak{R}^{nl \times n}$ and $\Pi_u \in \mathfrak{R}^{nu \times n}$, $l \in \mathfrak{R}^{nl}$, $u \in \mathfrak{R}^{nu}$; Π_l (or Π_u) denotes the matrix given by the rows of the identity matrix whose indices are equal to those of the entries of x which are bounded below (above); nl and nu denote the number of entries of the vector x subject to lower and upper bounds, respectively. If $x_i \geq l_j$ for some i and, simultaneously, $x_i \leq u_k$, we assume $l_j \neq u_k$. This hypothesis means that there are no fixed variables; otherwise, the problem can be reduced by eliminating them.

When problem (0.1) is a large scale variational inequality, a well know solution approach is to find a numerical solution of the Karush–Kuhn–Tucker (KKT) conditions for $VIP(C,F)$:

$$(0.3) \quad \begin{aligned} L(x, \lambda, \mu, \kappa_l, \kappa_u) &= 0 \\ h(x) &= 0 \\ \mu^T g(x) &= 0 \quad g(x) \geq 0 \quad \mu \geq 0 \\ \kappa_l^T (\Pi_l x - l) &= 0 \quad \Pi_l x - l \geq 0 \quad \kappa_l \geq 0 \\ \kappa_u^T (u - \Pi_u x) &= 0 \quad u - \Pi_u x \geq 0 \quad \kappa_u \geq 0 \end{aligned}$$

where $L(x, \lambda, \mu, \kappa_l, \kappa_u) = F(x) - \nabla h(x)\lambda - \nabla g(x)\mu - \Pi_l^T \kappa_l + \Pi_u^T \kappa_u$ is the Lagrangian function of the variational inequality. Here $\nabla h(x)^T$ and $\nabla g(x)^T$ denote the Jacobian matrices of $h(x)$ and $g(x)$ respectively. A vector $w = (x, \lambda, \mu, \kappa_l, \kappa_u) \in \mathfrak{R}^{n+p+m+nl+nu}$ satisfying (0.3) is called a KKT-vector, while the corresponding x vector is KKT-point. In the following, we will often refer to a KKT-vector simply as a KKT-point.

For the relationship between a KKT point and the solution of VIP(F,C) see [5].

In [2] (see also [1]), by Fischer and Burmeister function [4], the KKT-conditions for variational inequality problems are reformulated as a system of nonlinear semismooth equations $\Phi(w) = 0$, that can be solved by a non-smooth Newton method [9]. This local algorithm is globalized by damping the search direction to force a sufficient decrease of a merit function [2], such as the least square merit function $\Psi(w) = \frac{1}{2} \|\Phi(w)\|_2^2$. Thus, at each step of this iterative scheme, it is required the computation of the solution of a sparse linear system. In case of large-scale problems, the use of direct solvers can be very expensive and memory consuming. In [3], the nonlinear semismooth system is solved by a generalization of the Inexact Newton method (see [6]), that enables us to use an iterative solver for the inner linear systems. Following this approach, the local and global convergence of the whole method is guaranteed also when the descent direction for the merit function is computed approximately at each step; furthermore, we can devise for the inner scheme an *adaptive* stopping rule that allows to avoid unnecessary inner iterations when we are far from the solution, (i.e. for the initial outer iterations).

Previous numerical experience [7] show that an effective iterative inner solver is the LSQR method [8] combined with a suitable preconditioner, for example the incomplete LU factorization of the matrix

$$\begin{bmatrix} D_{\kappa_l} & 0 & 0 & D_l \Pi_l & 0 \\ 0 & D_{\kappa_u} & 0 & D_u \Pi_u & 0 \\ 0 & 0 & D_\mu & D_g (\nabla g(x))^T & 0 \\ \Pi_l^T & -\Pi_u^T & \nabla g(x) & \nabla_x L(w) & \nabla h(x) \\ 0 & 0 & 0 & (\nabla h(x))^T & 0 \end{bmatrix}$$

where $\nabla_x L(w) = -\nabla F(x) + \sum_{i=1}^m \nabla^2 g_i(x) \mu_i + \sum_{i=1}^p \nabla^2 h_i(x) \lambda_i$; $\nabla^2 g_i(x)$ and $\nabla^2 h_i(x)$ are hessian matrices of inequality and equality constrains, $D_{\kappa_l}, D_{\kappa_u}, D_\mu, D_l, D_u, D_g$ are diagonal matrices obtained as a generalized jacobian of the Fischer and Burmeister functions. Nevertheless for large-scale problems, the computation of this sparse preconditioner can be still expensive and memory consuming. For example, this happens when all the variables x have to satisfy box constraints.

In order to avoid these drawbacks, we introduce a preconditioner that admits a block-factorization; in this way, we use as preconditioner the matrix $\hat{H} = \overline{LU}$ where

$$\overline{L} = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \Pi_l^T (\tilde{D}_{\kappa_l})^{-1} & -\Pi_u^T (\tilde{D}_{\kappa_u})^{-1} & \hat{L} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\bar{U} = \begin{bmatrix} \tilde{D}_{\kappa_l} & 0 & 0 & D_l \Pi_l & 0 \\ 0 & \tilde{D}_{\kappa_u} & 0 & D_u \Pi_u & 0 \\ 0 & 0 & & & \\ 0 & 0 & & \hat{U} & \\ 0 & 0 & & & \end{bmatrix}$$

$\tilde{D}_{\kappa_l}, \tilde{D}_{\kappa_u}$ are diagonal matrices obtained as an approximation of the $D_{\kappa_l}, D_{\kappa_u}$ and \hat{L} and \hat{U} are the factors of the incomplete factorization of the matrix \hat{B} :

$$\hat{B} = \begin{bmatrix} D_\mu & D_g(\nabla g(x))^T & 0 \\ \nabla g(x) & A & \nabla h(x) \\ 0 & (\nabla h(x))^T & 0 \end{bmatrix}$$

with $A = \nabla_x L(w) - (\Pi_l)^T (\tilde{D}_{\kappa_l})^{-1} D_l \Pi_l + (\Pi_u)^T (\tilde{D}_{\kappa_u})^{-1} D_u \Pi_u$.

In this preconditioner we compute an incomplete LU factorization by factorizing submatrices \hat{B} of smaller size and we obtain a better performance of the semismooth inexact method in terms of execution time and of total number of inner iterations.

Finally in order to evaluate the effectiveness of the proposed approach, we report the numerical results obtained on some well-known test problems.

REFERENCES

1. T. De Luca and F. Facchinei and C. Kanzow *A semismooth equation approach to the solution of nonlinear complementarity problems*, Mathematical Programming 75(3), (1996), 407-439
2. F. Facchinei and A. Fischer and C. Kanzow, *A semismooth, Newton method for variational inequalities: Theoretical results and preliminary numerical experience*, Technical Report available at <http://www.dis.uniroma1.it/facchinei/>, (1995).
3. F. Facchinei and A. Fischer and C. Kanzow, *Inexact Newton methods for semismooth equations with applications to variational inequality problems*, Nonlinear optimization and applications. Proceeding of the 21th workshop, Erice, Italy, June 13-21, 1995. *New York, NY:Plenum Press*, (1996), 125-139.
4. A. Fischer, *A special Newton-type optimization method*, Optimization 24(5), (1992), 269-284.
5. P.T. Harker, *Accelerating the convergence of the diagonalization and projection algorithms for finite-dimensional variational inequalities*, Mathematical Programming 41, (1988), 29-59
6. J. M. Martínez and L. Qi, *Inexact Newton Methods for Solving Nonsmooth Equations*, Technical Report *Applied Mathematics Report 93/9*, School of Mathematics, University of New South Wales, Sydney, Australia, (1999).
7. T. D. Munson and F. Facchinei and M. C. Ferris and A. Fischer and C. Kanzow, *The semismooth algorithm for large scale complementarity problems*, INFORMS Journal on Computing 13, (2001), 294-311.

8. C. C. Paige and M. A. Saunders, *LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares*, ACM Transactions on Mathematical Software 8(1), (1982), 43-71.
9. L. Qi and J. Sun, *A nonsmooth version of Newton method*, Mathematical Programming 58, (1993), 353-367.