Nonlinear Overlapping Domain Decomposition Methods

Xiao-Chuan Cai[1]

Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309,
cai@cs.colorado.edu

Summary. We discuss some overlapping domain decomposition algorithms for solving sparse nonlinear system of equations arising from the discretization of partial differential equations. All algorithms are derived using the three basic algorithms: *Newton* for local or global nonlinear systems, *Krylov* for the linear Jacobian system inside Newton, and *Schwarz* for linear and/or nonlinear preconditioning. The two key issues with nonlinear solvers are robustness and parallel scalability. Both issues can be addressed if a good combination of Newton, Krylov and Schwarz is selected, and the right selection is often dependent on the particular type of nonlinearity and the computing platform.

1 Introduction

For solving partial differential equations on large scale parallel computers, domain decomposition is a natural choice. Overlapping Schwarz methods and non-overlapping iterative substructuring methods are the two major classes of domain decomposition methods [12, 13, 15]. In this paper we only consider overlapping methods for solving large sparse nonlinear system of equations arising from the discretization of nonlinear partial differential equations, i.e., for a given nonlinear function $F : R^n \rightarrow R^n$, we compute a vector $u \in R^n$, such that

$$F(u) = 0, \tag{1}$$

starting from an initial guess $u^{(0)} \in R^n$. Here $F = (F_1, \ldots, F_n)^T$, $F_i = F_i(u_1, \ldots, u_n)$, and $u = (u_1, \ldots, u_n)^T$. One of the popularly used techniques for solving (1) is the so-called *inexact Newton* algorithms (IN) which are described briefly here. Suppose $u^{(k)}$ is the current approximate solution and $J = F'(u^{(k)})$, a new approximate solution $u^{(k+1)}$ can be computed through the following steps: first find an inexact Newton direction $p^{(k)}$ by solving the Jacobian system

$$Jp^{(k)} = F(u^{(k)}) \tag{2}$$

such that $\|F(u^{(k)}) - Jp^{(k)}\| \leq \eta_k \|F(u^{(k)})\|$, then compute the new approximate solution

$$u^{(k+1)} = u^{(k)} - \lambda^{(k)} p^{(k)}. \tag{3}$$

Here $\eta_k \in [0,1)$ is a scalar that determines how accurately the Jacobian system needs to be solved using, for example, Krylov subspace methods. $\lambda^{(k)}$ is another scalar that determines the step length in the selected inexact Newton direction. Sometimes when J is not explicitly available, one can use the matrix-free version [11]. IN has several well-known features.

(a) Fast convergence. If the initial guess is close enough to the desired solution then the convergence is very fast (quadratic) provided that the η_k's are sufficiently small.
(b) Non-robustness. The convergence, or fast convergence, happens only if a good initial guess is available. Generally it is difficult to obtain such an initial guess especially for nonlinear equations that have unbalanced nonlinearities [12]. The step length $\lambda^{(k)}$ is often determined by the components with the strongest nonlinearities, and this may lead to an extended period of stagnation in the nonlinear residual curve. We say that the nonlinearities are "unbalanced" when $\lambda^{(k)}$, in effect, is determined by a subset of the overall degrees of freedom.
(c) Scalability. The parallel scalability of the method is mostly determined by how the Jacobian system (2) is solved.

There are a number of strategies [7, 8, 10], such as linesearch, trust region, continuation or better ways to choose the forcing term, to make the algorithm more robust or converge faster, however, these strategies are all based on certain global knowledge of F or J. In other words, all equations in the system are treated equally as if they were some of the worst equations in the system. Other ways to look at the global nature of IN are

(d) To advance from $u^{(k)}$ to $u^{(k+1)}$, all n variables and equations need to be updated even though in many situations n can be very large, but only a small number of components of $u^{(k)}$ receive significant updates.
(e) If a small number of components of the initial guess $u^{(0)}$ are not acceptable, the entire $u^{(0)}$ is declared bad.
(f) There are two global control variables η_k and $\lambda^{(k)}$. Any slight change of F may result in the change of η_k or $\lambda^{(k)}$, and any slight change of η_k or $\lambda^{(k)}$ may result in some global function evaluations and/or the solving of global Jacobian systems. For example, if the search direction $p^{(k)}$ has one unacceptable component, then the entire steplength is reduced.

Note that these global operations can be expensive when n is large and when the number of processors is large. Using domain decomposition methods, more localized treatments can be applied based on the location or the physical nature of the nonlinearities, and the number of global operations can be made small in some situations.

We should point out that the words "local" and "global" have different meanings in the context of domain decomposition methods [15] than in the context of nonlinear equation solvers [7], among others. In nonlinear solvers, "local" means a small neighborhood of the exact solution of the nonlinear system, and "global" means a

relatively large neighborhood of the exact solution of the nonlinear system. In domain decomposition, "local" means some subregions in the computational domain and "global" means the whole computational domain.

All the algorithms to be discussed in the paper are constructed with a combination of the three basic techniques: Newton, Krylov and Schwarz. Newton is the basic nonlinear solver that is used for either the system defined on the whole space or some subspaces (subdomain subspace or coarse subspace). Krylov is the basic linear solver that is used inside a Newton solver. Schwarz is a preconditioner for either the linear or the nonlinear solver. Many algorithms can be derived with different combinations of the three basic algorithms. For a given class of problems and computing platform, a special combination might be necessary in order to obtain the best performance. The three basic algorithms are all well understood individually, however, the construction of the best combination remains a challenge. The same can be said for the software. All software components are readily available in PETSc [1], but some of the advanced combinations have to be programmed by the user.

We next define (informally) some notations for describing domain decomposition methods. u is understood as a discrete (or coefficients of a finite element) function defined on the computational domain Ω which is already partitioned into a set of subdomains $\{\Omega_1^\delta, \cdots, \Omega_N^\delta\}$. Here Ω_i^δ is an δ-extension of Ω_i, and the collection of $\{\Omega_i\}$ is a non-overlapping partition of Ω. We define R_i^δ as a restriction operator associated with Ω_i^δ and R_i^0 as the restriction operator associated with Ω_i. We denote $u_{\Omega_i^\delta}$ as the restriction of u on Ω_i^δ, and $u_{\partial\Omega_i^\delta}$ as the restriction of u on the "boundary" of Ω_i^δ. Here we use the word "domain" to denote the mesh points in the interior of the domain and "boundary" to denote the mesh points on the boundary of the domain. Similarly, we may restrict the nonlinear function to a subdomain, such as $F_{\Omega_i^\delta}$. For boundary value problems considered in this paper, we assume

$$F_{\Omega_i^\delta}(u) = F_{\Omega_i^\delta}(u_{\Omega_i^\delta}, u_{\partial\Omega_i^\delta}).$$

That is to say that there are no "global equations" in the system that may couple the equations defined at a mesh point to equations defined outside a small neighborhood.

The rest of the paper is organized as follows. In Section 2, we discuss the most popular overlapping nonlinear domain decomposition method, Newton-Krylov-Schwarz algorithm, and in Sections 3–6, we discuss some more advanced nonlinear methods. Some final remarks are given in Section 7.

2 Newton-Krylov-Schwarz Algorithms

Newton-Krylov-Schwarz (NKS) is simply the application of a linear Schwarz preconditioner for solving the Jacobian equation (2) in the inexact Newton algorithm [2, 3]. Depending on what type of Schwarz preconditioner is used (additive, multiplicative, restricted, one-level, two-level, etc), there are several NKS algorithms. Let us define the subdomain preconditioners as

$$J_i = R_i^{\delta} J (R_i^{\delta})^T, \;\; i = 1, \ldots, N,$$

then the additive Schwarz preconditioner can be written as

$$M_{AS}^{-1} = \sum_{i=1}^{N} (R_i^{\delta})^T J_i^{-1} R_i^{\delta}.$$

Because of its simplicity, NKS has become one of the most popular domain decomposition methods for solving nonlinear PDEs and is the default nonlinear solver in PETSc [1]. The nonlinear properties of NKS are exactly the same as that of inexact Newton. For example, the initial guess has to be sufficiently close to the solution in order to obtain convergence, and fast convergence can be achieved when the nonlinearity is well balanced. NKS addresses the scalability issue (c) of IN well, but not the other issues (a, b, d–f).

3 Classical Schwarz Alternating Algorithms

Let $(u^{(0)}_{\Omega_1^{\delta}}, \ldots, u^{(0)}_{\Omega_N^{\delta}})$ be the initial guess for all subdomains. The classical Schwarz alternating algorithm (SA) can be described as follows:

$$\begin{aligned}
&k = 1, \ldots, \textit{ till convergence condition is satisfied} \\
&\quad i = 1, \ldots, N \\
&\qquad \textit{define } u^{(k)}_{\partial\Omega_i^{\delta}} \textit{ using } \left\{ u^{(k-1)}_{\Omega_j^{\delta}}, 1 \le j \le N \right\} \textit{ or } \left\{ u^{(k)}_{\Omega_j^{\delta}}, 1 \le j < i \right\} \\
&\qquad \textit{compute } u^{(k)}_{\Omega_i^{\delta}} \textit{ by solving } F_{\Omega_i^{\delta}} \left(u^{(k)}_{\Omega_i^{\delta}}, u^{(k)}_{\partial\Omega_i^{\delta}} \right) = 0.
\end{aligned} \tag{SA}$$

The algorithm doesn't belong to the class of IN algorithms and, in general, not share properties (a–f). The method is usually not used by itself as a nonlinear solver because of its slow convergence, but in some cases when the nonlinearities are isolated within some of the subdomains, the method can be a good alternative to IN. Note that SA doesn't involve any global operations.

4 Nonlinear Additive Schwarz Preconditioned Inexact Newton Algorithms

The basic idea of nonlinearly preconditioned inexact Newton algorithms [4, 9] is to find the solution $u \in R^n$ of (1) by solving an equivalent system

$$\mathcal{F}(u) = 0 \tag{4}$$

using IN. Systems (1) and (4) are said to be equivalent if they have the same solution. For any given $v \in R^n$, we define a subdomain projection $T_i(v)$, which is a function with support in Ω_i^{δ}, as the solution of the following subspace nonlinear system

$$F_{\Omega_i^\delta}(v - T_i(v)) = 0,$$

for $i = 1, \ldots, N$. Then a nonlinearly preconditioned function is defined as

$$\mathcal{F}(u) = \sum_{i=1}^{N} T_i(u).$$

It can be shown that, under certain conditions, for this particular $\mathcal{F}$, (1) and (4) offer the same solution subject to the error due to different stopping conditions and preconditioners. This algorithm is often referred to as the additive Schwarz preconditioned inexact Newton algorithm (ASPIN). Sometimes we call it a left preconditioned IN because in the linear case (i.e., $F(u) = Ju - b$) $\mathcal{F}(u) = (\sum_{i=1}^{N} (R_i^\delta)^T J_i^{-1} R_i^\delta)(Ju - b)$.

When using IN to solve (4), the Jacobian of $\mathcal{F}$, or its approximation, is needed. Because of the special definition of the function $\mathcal{F}$, its Jacobian can only be given as the sum of matrix-vector products and the explicit elements of $\mathcal{F}'$ are not available.

It is known that for left preconditioned linear iterative methods, the stopping condition is often influenced by the preconditioner. The impact of the preconditioner on the stopping condition can be removed if the preconditioner is applied to the right. Unlike linear preconditioning, the switch from left to right is not trivial in the nonlinear case. A right nonlinear preconditioner will be discuss in a later section of the paper.

5 Nonlinear Elimination Algorithms

The nonlinear elimination algorithm (NE) was introduced in [12] for nonlinear algebraic systems with local high nonlinearities. It was not introduced as a domain decomposition method, but we include it in the paper because it is the main motivation for the algorithm to be discussed in the next section. Suppose that the function F is more nonlinear in the subdomain Ω_i^δ, then we can eliminate all unknowns in this particular subdomain and let Newton work on the rest of the variables and equations. Let $y = u|_{\Omega_i^\delta}$ and $x = u|_{\Omega \setminus \Omega_i^\delta}$, then using the implicit function theorem, under some assumptions, we can solve for y in terms of x; i.e., solve

$$F_{\Omega_i^\delta}(x, y) = 0$$

for y, which symbolically equals to $y = F_{\Omega_i^\delta}^{-1}(x)$. After the elimination, we can use the regular Newton method for the rest of the system which is more balanced, at least in theory,

$$F_{\Omega \setminus \Omega_i^\delta}\left(x, F_{\Omega_i^\delta}^{-1}(x)\right) = 0.$$

The algorithm has some obvious advantages. We mention some of its disadvantages as a motivation for the algorithm to be discussed in the next section. In practice, it is often difficult to tell which components are more nonlinear than the others, and the situation may change from iteration to iteration. The algorithm may introduce

sharp jumps in the residual function near the interface of x and y. Such jumps may lead to slow convergence or divergence. Some improved versions are given in [6]. In the next section, we combine the ideas of ASPIN and NE into a right preconditioned Newton method.

6 Nonlinear Restricted Additive Schwarz Algorithms

In [5], a right preconditioned inexact Newton algorithm was introduced as follows: Find the solution $u \in R^n$ of (1) by first solving a preconditioned nonlinear system

$$F(G(v)) = 0$$

for v, and then obtain $u = G(v)$. For any given $v \in R^n$, we define a subdomain projection $T_i(v)$, which is a function with support in Ω_i^δ, as the solution of the following subspace nonlinear system

$$F_{\Omega_i^\delta}(v + T_i(v)) = 0,$$

for $i = 1, \ldots, N$. Then the nonlinear preconditioning function is defined as

$$G(v) = v + \sum_{i=1}^{N} R_i^0 T_i(v).$$

Here the non-overlapping restriction operator R_i^0 effectively removes the sharp jumps on the interfaces of the overlapping subdomains. In the linear case

$$G(v) = v - \Big(\sum_{i=1}^{N} (R_i^0)^T J_i^{-1} R_i^\delta \Big)(Jv - b),$$

which can be regarded as a restricted additive Schwarz preconditioned Richardson method.

This preconditioner doesn't have to be applied at every outer Newton iteration. It is used only when some local high nonlinearities are sensed, somehow. Below we describe the overall algorithm (NKS-RAS). The goal is to solve equation (1) with a given initial guess $u^{(0)}$. Suppose $u^{(k)}$ is the current solution.

Step 1 (Nonlinearity Checking): *Check local and global stopping conditions.*

- *If the global condition is satisfied, stop.*
- *If local conditions indicate that nonlinearities are not balanced, go to Step* 2*.*
- *If local conditions indicate that nonlinearities are balanced, set* $\tilde{u}^{(k)} = u^{(k)}$*, go to Step* 3*.*

Step 2 (RAS): *Solve local nonlinear problems on the overlapping subdomains to obtain the subdomain corrections* $T_i(u^{(k)})$

$$F_{\Omega_i^\delta}\big(u^{(k)} + T_i(u^{(k)})\big) = 0 \quad \textit{for } i = 1, \ldots, N.$$

Drop the solution in the overlapping part of the subdomain and compute the global function $G(u^{(k)})$ *and set*

$$\tilde{u}^{(k)} = G(u^{(k)}).$$

Go to Step 3.

Step 3 (NKS): *Compute the next approximate solution* $u^{(k+1)}$ *by solving the following system*

$$F(u) = 0$$

with one step of NKS using $\tilde{u}^{(k)}$ *as the initial guess.*

Go to Step 1.

The nonlinearity checking step is important. However, we only have a few ad hoc techniques such as computing the residual norm subdomain by subdomain (or field by field in the case of multi-physics applications). If some of the subdomain (or subfield) norms are much larger than for other subdomains, we label these subdomains as highly nonlinear subdomains and proceed with the RAS elimination step. Otherwise, when the nonlinearity is more or less balanced we bypass the RAS step and go directly to the global NKS step. The subdomain nonlinear systems in Step 2 do not need to be solved very accurately since the solutions are used only to construct an initial guess for Step 3. In NKS-RAS, a nonlinear system is set up on each subdomain, but in practice, not all subdomain nonlinear problem needs to be solved. In the not-too-nonlinear regions, the solver may declare to have converged in 0 iteration.

7 Concluding Remarks

In this paper, we have given a quick overview of overlapping domain decomposition methods for solving nonlinear partial differential equations. The two key issues of nonlinear methods are robustness and scalability. Both issues can be addressed by using some combinations of the three basic algorithms: Newton, Krylov and Schwarz. Several algorithms are presented in the paper together with some of their advantages and disadvantages. Depending on the particular types of nonlinearities and the computing platform, different combinations of the three basic algorithms may be needed in order to obtain the best performance and robustness. Due to page limit, applications have not been discussed in the paper. Some of them can be found in the references.

Acknowledgement. This research was supported in part by DOE under DE-FC02-01ER25479 and DE-FC02-04ER25595, and in part by NSF under grants ACI-0305666, CNS-0420873, CCF-0634894, and CNS-0722023.

References

[1] Balay, S., Buschelman, K., Gropp, W.D., Kaushik, D., Knepley, M., McInnes, L.C., Smith, B.F., Zhang, H.: *PETSc Users Manual*, ANL, 2008.

[2] Cai, X.-C., Gropp, W., Keyes,D., Melvin, R., Young, D.P.: *Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equation*, SIAM J. Sci. Comput., 19 (1998), 246–265.

[3] Cai, X.-C., Gropp, W.D., Keyes, D.E., Tidriri, M.D.: *Newton-Krylov-Schwarz methods in CFD*, Proceedings of the International Workshop on the Navier-Stokes Equations, Notes in Numerical Fluid Mechanics, R. Rannacher, eds. Vieweg, Braunschweig, 1994.

[4] Cai, X.-C., Keyes, D.E.: *Nonlinearly preconditioned inexact Newton algorithms*, SIAM J. Sci. Comput., 24 (2002), 183–200.

[5] Cai, X.-C., Li, X.: *Inexact Newton methods with nonlinear restricted additive Schwarz preconditioning for problems with high local nonlinearities*, in preparation.

[6] Cai, X.-C., Li, X.: *A domain decomposition based parallel inexact Newton's method with subspace correction for incompressible Navier-Stokes equations*, Lecture Notes in Computer Science, Springer, 2009.

[7] Dennis, J.E., Schnabel, R.B.: *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, 1996.

[8] Eisenstat, S.C., Walker, H.F.: *Choosing the forcing terms in an inexact Newton method*, SIAM J. Sci. Comput., 17 (1996), 16–32.

[9] Hwang, F.-N., Cai, X.-C.: *A parallel nonlinear additive Schwarz preconditioned inexact Newton algorithm for incompressible Navier-Stokes equations,* J. Comput. Phys., 204 (2005), 666–691.

[10] Kelley, C.T., Keyes, D.E.: *Convergence analysis of pseudo-transient continuation*, SIAM J. Numer. Anal., 35 (1998), 508–523.

[11] Knoll, D., Keyes, D.E.: *Jacobian-free Newton-Krylov methods: a survey of approaches and applications*, J. Comput. Phys., 193 (2004), 357–397.

[12] Lanzkron, P.J., Rose, D.J., Wilkes, J.T.: *An analysis of approximate nonlinear elimination*, SIAM J. Sci. Comput., 17 (1996), 538–559.

[13] Smith, B., Bjørstad, P., Gropp, W.: *Domain Decomposition. Parallel Multilevel Methods for Eliptic Partial Differential Equations,* Cambridge University Press, New York, 1996.

[14] Quarteroni, A., Valli, A.: *Domain Decomposition Methods for Partial Differential Equations*, Oxford University Press, Oxford, 1999.

[15] Toselli, A., Widlund, O.: *Domain Decomposition Methods—Algorithms and Theory*, Springer, Berlin, 2005.