A Domain Decomposition Based Two-Level Newton Scheme for Nonlinear Problems

Deepak V. Kulkarni and Daniel A. Tortorelli*

University of Illinois,
Department of Mechanical and Industrial Engineering
Urbana, Illinois 61801

Summary. We present two non-overlapping domain decomposition based two-level Newton schemes for solving nonlinear problems and demonstrate their effectiveness by analyzing systems with balanced and unbalanced nonlinearities. They both have been implemented in parallel and show good scalability. The implementations accommodate non-symmetric matrices and unstructured meshes.

1 Introduction

One can refer to the paper by Keyes [1992] and the book by Smith et al. [1996] for in-depth reviews of domain decomposition (DD) methods. Of particular interest here are non-overlapping schemes such as iterative substructuring (Bjørstad et al. [2001]) and FETI methods (Farhat et al. [2001]).

When solving non-linear boundary value problems (BVPs) via domain decomposition it is common to use Newton type algorithms and then to apply existing DD approaches to the ensuing linearized problems (Knoll and Keyes [2002]). The NK-Schwarz scheme (Keyes [1995]) as the name suggests, uses a Krylov scheme equipped with a Schwarz preconditioner to solve this linear update equation.

If the non-linear effects are unbalanced, i.e., the nonlinearity has a significant spatial variation, then the Jacobian becomes ill-conditioned and hence the NK-Schwarz scheme is not effective cf. Cai and Keyes [2002]. A scheme that proves effective for problems with unbalanced nonlinearities is the multi level Newton Schwarz (MLN-Schwarz) scheme that was originally introduced to solve multi-physics problems (Bächtold et al. [1995], Aluru and White [1999]). Kim et al. [2003] implemented a serial version of the MLN-Schwarz to solve fluid-structure interaction problems which had the flavor of a multiplicative Schwarz approach. The scheme employs a global consistency equation in

* Both the authors would like to acknowledge the support provided by NSF under grant no. DMR 01-21695

the place of the standard residual equation. In the case of unbalanced nonlinearities, the Jacobian for the global consistency equation appears to be better conditioned (Cai and Keyes [2002]). However, the MLN-Schwarz requires the full solution of the sub-domain residual equations for each global Newton iteration. Hence the scheme is not efficient for problems with balanced nonlinearities.

Another scheme that is used to resolve BVPs with unbalanced nonlinearities is the ASPIN method. Cai et al. [2001] introduced this as a nonlinearly preconditioned version of the NK-Schwarz scheme. In comparison to the MLN-Schwarz, the ASPIN method has been implemented in parallel and accommodates both overlapping and non-overlapping domains. However, like the MLN-Schwarz scheme, the ASPIN method is inefficient for problems with balanced nonlinearities.

In this work we present two non-overlapping DD schemes to solve nonlinear BVPs. The first scheme, which we call the modified Newton Krylov Schur (MNK-Schur) approach, is based on a Newton Krylov Schur (NK-Schur) approach. Since our method uses a two-level Newton scheme, it efficiently solves problems with unbalanced nonlinearities thereby incorporating the advantages of both the NK-Schwarz and ASPIN methods. The second method modifies the MLN-Schwarz method to obtain a non-overlapping DD scheme. We show that this scheme is in fact a special case of our MNK-Schur approach.

2 Two-level Newton Krylov Schur Approach

For our finite element (or similar) computations we partition the domain into n non-overlapping sub-domains and represent the discretized nodal response vector $\mathbf{u}$ and global residual vector $\mathbf{R}(\mathbf{u})$ for the entire domain Ω as:

$$\mathbf{u} = \begin{Bmatrix} \mathbf{u}_1^S \\ \vdots \\ \mathbf{u}_n^S \\ \mathbf{u}^I \end{Bmatrix} \qquad \mathbf{R}(\mathbf{u}) = \begin{Bmatrix} \mathbf{R}_1^S(\mathbf{u}_1^S, \mathbf{u}^I) \\ \vdots \\ \mathbf{R}_n^S(\mathbf{u}_n^S, \mathbf{u}^I) \\ \boldsymbol{\mathcal{R}}(\mathbf{u}_1^S, \mathbf{u}_2^S, \ldots, \mathbf{u}_n^S, \mathbf{u}^I) \end{Bmatrix} \tag{1}$$

where $\mathbf{u}^I$ corresponds to the interface nodal Degrees Of Freedom (DOF) and $\mathbf{u}_j^S$ corresponds to the internal sub-domain and Neumann boundary nodal DOF of sub-domain j. In the above equation the interface residual $\boldsymbol{\mathcal{R}}$ is assembled as

$$\boldsymbol{\mathcal{R}}(\mathbf{u}_1^S, \mathbf{u}_2^S, \ldots, \mathbf{u}_n^S, \mathbf{u}^I) = \sum_{j=1}^{n} \mathbf{R}_j^I(\mathbf{u}_j^S, \mathbf{u}^I) \tag{2}$$

where $\mathbf{R}_j^I(\mathbf{u}_j^S, \mathbf{u}^I)$ represents the contribution of sub-domain j to the interface nodal residual vector. Note that the residual $\mathbf{R}(\mathbf{u})$ is a reordering of the residual that one would form without decomposition techniques. We apply Newton's method to the nonlinear residual equation $(1)_2$ and obtain the update equation:

$$\begin{bmatrix} \frac{\partial \mathbf{R}_1^S}{\partial \mathbf{u}_1^S} & \cdots & \mathbf{0} & \frac{\partial \mathbf{R}_1^S}{\partial \mathbf{u}^I} \\ \vdots & \ddots & \mathbf{0} & \vdots \\ \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{R}_n^S}{\partial \mathbf{u}_n^S} & \frac{\partial \mathbf{R}_n^S}{\partial \mathbf{u}^I} \\ \frac{\partial \boldsymbol{\mathcal{R}}^I}{\partial \mathbf{u}_1^S} & \cdots & \frac{\partial \boldsymbol{\mathcal{R}}}{\partial \mathbf{u}_n^S} & \frac{\partial \boldsymbol{\mathcal{R}}}{\partial \mathbf{u}^I} \end{bmatrix} \begin{Bmatrix} \Delta \mathbf{u}_1^S \\ \vdots \\ \Delta \mathbf{u}_n^S \\ \Delta \mathbf{u}^I \end{Bmatrix} = - \begin{Bmatrix} \mathbf{R}_1^S \\ \vdots \\ \mathbf{R}_n^S \\ \boldsymbol{\mathcal{R}} \end{Bmatrix} \tag{3}$$

On applying block Gauss elimination we obtain the Schur's complement representation of the above. We first solve for the interface increment $\Delta \mathbf{u}^I$ from

$$\sum_{j=1}^{n} \left[\frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}^I} - \frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}_j^S} \left[\frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}_j^S} \right]^{-1} \frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}^I} \right] \Delta \mathbf{u}^I = -\boldsymbol{\mathcal{R}}^I + \sum_{j=1}^{n} \frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}_j^S} \left[\frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}_j^S} \right]^{-1} \mathbf{R}_j^S \tag{4}$$

then we solve for the sub-domain increments from

$$\left[\frac{D\mathbf{R}_j^S}{D\mathbf{u}_j^S} \right] \Delta \mathbf{u}_j^S = -\mathbf{R}_j^S - \frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}^I} \Delta \mathbf{u}^I \tag{5}$$

and finally we update the interface and sub-domain DOF vectors as

$$\mathbf{u}^I = \mathbf{u}^I + \Delta \mathbf{u}^I; \qquad \mathbf{u}_j^S = \mathbf{u}_j^S + \Delta \mathbf{u}_j^S \tag{6}$$

The process repeats until convergence of equation $(1)_2$.

The algorithm as described above is equivalent to a NK-Schur approach. However, as discussed in the previous section this algorithm has poor convergence if the Jacobian in equation (3) is ill-conditioned. To alleviate this problem we augment the algorithm with a lower level sub-domain Newton scheme to obtain our MNK-Schur algorithm. After updating $\mathbf{u}^S$ and $\mathbf{u}^I$ at every Newton iteration (cf. equation (6)) we perform additional sub-domain iterations keeping the interface unknowns fixed via

$$\left[\frac{D\mathbf{R}_j^S}{D\mathbf{u}_j^S} \right] \Delta \mathbf{u}_j^S = -\mathbf{R}_j^S; \qquad \mathbf{u}_j^S = \mathbf{u}_j^S + \Delta \mathbf{u}_j^S \tag{7}$$

We may or may not iterate until sub-domain convergence is obtained, i.e., until $\mathbf{R}_j^S \approx \mathbf{0}$. In either case we revert to the NK-Schur approach and repeat equations (3)-(7) until $\mathbf{R}(\mathbf{u}) \approx \mathbf{0}$ (cf. equation $(1)_2$).

Remark 1: If a particular sub-domain is linear, the tangent matrices $\partial \mathbf{R}_j^S / \partial \mathbf{u}_j^S$, $\partial \mathbf{R}_j^S / \partial \mathbf{u}^I$, $\partial \mathbf{R}^I / \partial \mathbf{u}^I$, $\partial \mathbf{R}^I / \partial \mathbf{u}_j^S$ remain constant in all applications of equations (4), (5) and (7)

Remark 2: Various criteria can be used to determine if additional sub-domain iterations are required. In our implementation we perform sub-domain iterations if $|\mathbf{R}_i^S| > |\mathbf{R}_j^S|_{avg}$ and $|\mathbf{R}_j^S| > \varepsilon_{sub-domain}$ where $|\mathbf{R}_j^S|_{avg}$ is the average norm of all sub-domain residuals (where the $\mathbf{R}_j^S$ are evaluated after the update of equation (6) is completed) and $\varepsilon_{sub-domain}$ is a tolerance.

Remark 3: The Newton iteration at the sub-domain level of the MLN-Schwarz and ASPIN methods is precisely the augmented sub-domain iteration introduced in our MNK-Schur method (cf. equations (7)). We introduce here, a non-overlapping multi level Newton Schur (MLN-Schur) approach. The global consistency equations of the MLN-Schwarz and ASPIN methods are replaced by the interface residual equation (2) now expressed as

$$\boldsymbol{\mathcal{R}}\left(\mathbf{u}^I; \hat{\mathbf{u}}_1(\mathbf{u}^I), \hat{\mathbf{u}}_2(\mathbf{u}^I), \cdots, \hat{\mathbf{u}}_n(\mathbf{u}^I)\right) = \sum_{j=1}^{n} \mathbf{R}_j^I\left(\hat{\mathbf{u}}_j(\mathbf{u}^I), \mathbf{u}_j^I\right) = \mathbf{0} \tag{8}$$

The update equation for the above problem is

$$\left[\frac{D\boldsymbol{\mathcal{R}}}{D\mathbf{u}^I}\right] \Delta\mathbf{u}^I = -\boldsymbol{\mathcal{R}}\,; \qquad\qquad \mathbf{u}^I = \mathbf{u}^I + \Delta\mathbf{u}^I \tag{9}$$

where, upon applying chain rule to (8) and differentiating the sub-domain residual equation (here expressed as $\mathbf{R}_j^S(\hat{\mathbf{u}}_j^S(\mathbf{u}^I), \mathbf{u}^I) = \mathbf{0}$) we obtain

$$\frac{D\boldsymbol{\mathcal{R}}}{D\mathbf{u}^I} = \sum_{j=1}^{n}\left[\frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}^I} - \frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}_j}\left[\frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}_j^S}\right]^{-1}\frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}^I}\right] \tag{10}$$

We notice immediately that $D\boldsymbol{\mathcal{R}}/D\mathbf{u}^I$ is the Schur's complement matrix of equation (4). However the right hand side of the above MLN-Schur update equation (9) does not contain the sub-domain residual $\mathbf{R}_j^S$ present in the MNK-Schur update equation (4). This is to be expected in the MLN-Schur scheme because the sub-domain problem is resolved making $\mathbf{R}_j^S \approx \mathbf{0}$. Thus the MLN-Schur scheme is a special case of the MNK-Schur scheme.

3 Implementation

We use a direct solver to resolve the linear sub-domain update equations (5) and (7). It is noted that the use of a direct solver enables us to store the factored sub-domain Jacobian. Obviously, the sub-domain computations are independent of each other and therefore easily parallelized.

We employ an iterative method (e.g. GMRES, Saad and Schultz [1986]) to solve the interface update equation (4) in parallel. The iterative scheme requires multiple evaluations of the matrix-vector product $\left[D\boldsymbol{\mathcal{R}}/D\mathbf{u}^I\right]\mathbf{s}$ until equation (4) converges. Expanding this product we see that

$$\frac{D\boldsymbol{\mathcal{R}}}{D\mathbf{u}^I}\,\mathbf{s} = \sum_{j=1}^{n}\left[\frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}^I}\,\mathbf{s} - \frac{\partial \mathbf{R}^I}{\partial \mathbf{u}_j^S}\overbrace{\left[\frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}_j^S}\right]^{-1}\left\{\frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}^I}\,\mathbf{s}\right\}}^{\mathbf{x}_j}\right] \tag{11}$$

It is emphasized that each matrix-vector product required by the iterative solver involves a back solve with the already factored sub-domain Jacobian matrices to obtain $\mathbf{x}_j$. In fact the interface Jacobian matrix $D\boldsymbol{\mathcal{R}}/D\mathbf{u}^I$ is never assembled, only its effect on the vector $\mathbf{s}$ is evaluated, i.e., at the interface level this is a matrix free method. However, we form a preconditioner $\mathbf{P} = [\widetilde{\frac{D\boldsymbol{\mathcal{R}}}{D\mathbf{u}^I}}]^{-1}$ where

$$\widetilde{\frac{D\boldsymbol{\mathcal{R}}}{D\mathbf{u}^I}} = \sum_{j=1}^{n} \left[\frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}_j^I} - \left(\frac{\partial \mathbf{R}_j^I}{\partial \mathbf{u}_j^S} \, \mathbf{D}_j^{-1} \, \frac{\partial \mathbf{R}_j^S}{\partial \mathbf{u}_j^I} \right) \right] \tag{12}$$

and $\mathbf{D}_j$ is the diagonal of the sub-domain Jacobian $\partial \mathbf{R}_j^S / \partial \mathbf{u}_j^S$.

The proposed scheme is summarized in algorithm (1) where $\varepsilon_{sub-domain}$, ε_{global} and $\varepsilon_{iterative}$ are prescribed tolerances.

Algorithm 1 Modified Newton Krylov Schur Algorithm

Partition the mesh
Initialize $\mathbf{u}^I$, $\mathbf{u}^S$, compute $\mathbf{R}_j^I$, $\mathbf{R}_j^S$, $\partial \mathbf{R}_j^I / \partial \mathbf{u}_j^I$, $\partial \mathbf{R}_j^S / \partial \mathbf{u}_j^I$, $\partial \mathbf{R}_j^I / \partial \mathbf{u}_j^S$, factor $\partial \mathbf{R}_j^s / \partial \mathbf{u}_j^S$ and assemble the interface preconditioner, cf. equation (12)
repeat {Newton iterations}
  repeat { iterative solver computation of $\Delta \mathbf{u}^I$ }
    • Evaluate $(D\boldsymbol{\mathcal{R}}/D\mathbf{u}^I)\,\mathbf{s}$ via equation (11)
    • Update $\mathbf{s}$
  until $|(D\boldsymbol{\mathcal{R}}/D\mathbf{u}^I)\mathbf{s} + \boldsymbol{\mathcal{R}} - \sum_{j=1}^{n} \left[\partial \mathbf{R}_j^S / \partial \mathbf{u}_j^S\right]^{-1} \mathbf{R}_j^S| < \varepsilon_{iterative}$ (cf. eq. (4))
  • Solve $j = 1, 2, \ldots, n$ local sub-domain update equations (5)
  • Update $\mathbf{u}^I$ and $\mathbf{u}_j^S$ via equation (6)
  repeat { sub-domain Newton iterations}
    • Solve Newton update equation (7) and store $\partial \mathbf{R}_j^S / \partial \mathbf{u}_j^S$ in factored form
    • Update sub-domain response $\mathbf{u}_j^S$ via equation (7)
  until $|\mathbf{R}_j^S| < \varepsilon_{sub-domain}$ or $|\mathbf{R}_j^S| < |\mathbf{R}_j^S|_{avg}$
  • Compute $\mathbf{R}_j^I$, $\mathbf{R}_j^S$, $\partial \mathbf{R}_j^I / \partial \mathbf{u}_j^I$, $\partial \mathbf{R}_j^S / \partial \mathbf{u}_j^I$, $\partial \mathbf{R}_j^I / \partial \mathbf{u}_j^S$, $\left[\mathrm{Diag}(\partial \mathbf{R}_j^s / \partial \mathbf{u}_j^S)\right]^{-1}$
  • Assemble the interface preconditioner, cf. equation (12)
until $|\mathbf{R}(\mathbf{u})| < \varepsilon_{global}$

4 Results

We have developed parallel domain-decomposition codes using MPI (Forum [1994]) to implement the proposed methodologies. We use METIS (Karypis and Kumar) to partition the domain, SuperLU (Demmel et al.) for the sparse solution of the sub-domain problems and PETSc (Balay et al.) for the iterative solution of the interface problem. All computations are performed on a distributed shared memory Origin 2000 machine.

The preconditioner matrix is in general dense and could be computationally expensive to factorize. For problems in which $(dim(\mathbf{u}^I)/max(dim(\mathbf{u}_j^s))) <$

1, i.e., for problems with few sub-domains, we use an LU factorization to obtain $\mathbf{P}$ otherwise we use a Jacobi method[2] to approximate $\mathbf{P}$.

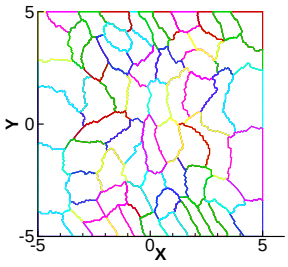

Fig. 1. Domain Partitioning

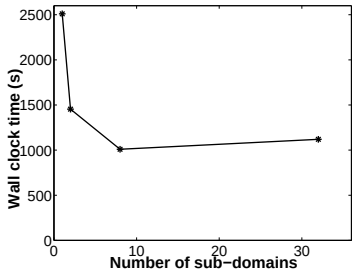

Fig. 2. Single processor efficiency

We consider a steady-state heat conduction problem to demonstrate the proposed algorithms. The nonlinear isotropic heat conduction coefficient κ is defined as: $\kappa(T) = \kappa_0(1 + \gamma\ T)$ where T is the temperature and κ_0 and γ are parameters, the latter of which controls the nonlinearity of the problem. Figure (1) shows the rectangular domain partitioned into 64 sub-domains. We impose zero flux conditions on the north and south boundaries and Dirichlet conditions of $T = 500$ and $T = 0$ on the west and east boundaries respectively. The discretization contains approximately 100,000 elements and 50,000 DOF.

4.1 Single processor efficiency

Figure 2 shows the timing results obtained using the MNK-Schur approach with varying number of sub-domains on a single processor. The problem uses $\kappa_0 = 1$ and $\gamma = 0.01$ and exhibits a balanced nonlinearity. The clock time for the single sub-domain case is obtained using the *dgssv* sparse direct solver of SuperLU (Demmel et al.). As seen from the figure, even on a single processor the DD based MNK-Schur approach performs better than a standard Newton scheme (i.e., the single sub-domain case) equipped with a sparse direct solver.

The timing results obtained for such single processor cases are used as baseline results for evaluating the scalability of the parallel implementations.

4.2 Parallel Scalability

To study the parallel scalability of the MNK-Schur algorithm we analyze the problem described in the previous section with 32 sub-domains and varying number of processors. The MNK-Schur shows near linear scale-up at a 55% to 65% efficiency as shown in figure 3. Note that the ability of our parallel implementation to accommodate multiple sub-domains per processor is

[2] PETSc has several built-in preconditioners that can be chosen at run time in place of the Jacobi method.

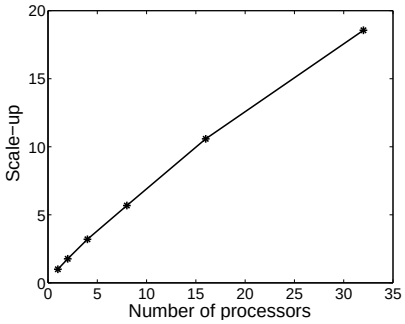

Fig. 3. Scalability of MNK-Schur

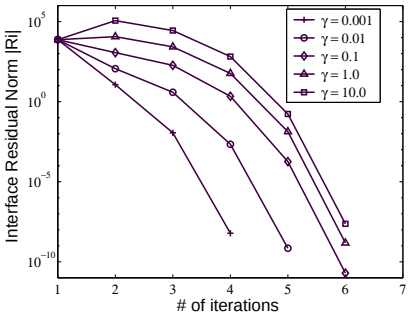

Fig. 4. Nonlinear convergence

demonstrated in these examples as the number of sub-domains is fixed while the number of processors is varied. Figure (4) shows the terminal quadratic convergence of the MNK-Schur scheme for several nonlinear problems.

4.3 Comparison of the two algorithms

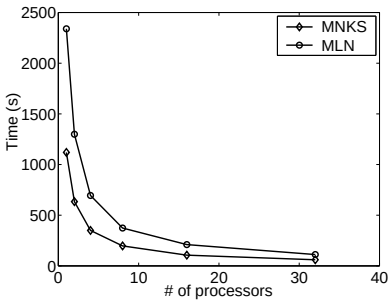

Fig. 5. MLN-Schur vs. MNK-Schur

To compare our MNK-Schur and MLN-Schur algorithms we analyze the 32 sub-domain case of the previous example problem. In figure 5 we plot the wall-clock time of the MLN-Schur and MNK-Schwarz schemes for varying number of processors. We see that the MLN-Schur scheme is less efficient than the MNK-Schur scheme irrespective of the number of processors employed. This difference is attributed to the full resolution of the sub-domain residual equations in the MLN-Schur scheme. Hence, the MLN-Schur requires more computations per interface Newton iteration.

However, for problems with unbalanced nonlinearities, fewer interface Newton iterations may be required when using the MLN-Schur method.

5 Conclusion

We have introduced two non-overlapping DD schemes based on a two-level Newton approach. The MNK-Schur scheme combines the advantages of the MLN-Schur and NK-Schur schemes to provide a general approach that efficiently solves problems with balanced and unbalanced nonlinearities. The DD implementation shows good scalability. By assigning multiple sub-domains to each processor we obtain a scheme that is efficient on a single processor

and one that is amenable to load balancing in parallel implementations. The implementations have been designed to accommodate unstructured meshes, nonsymmetric matrices and a variety of iterative solvers and preconditioners.

References

N. Aluru and J. White. A MLN method for mixed-energy domain sim. of MEMS. *Journal of MEMS systems*, 8(3):299–308, 1999.

M. Bächtold, J. Korvink, J. Funk, and H. Baltes. New convergence scheme for self-consistent electromech. analysis of iMEMS. In *IEEE Inter. Electron Devices Meeting*, 1995.

S. Balay, W. Gropp, L. McInnes, and B. Smith. PETSc, v. 2.1.3 code and documentation. URL `http://www-unix.mcs.anl.gov/petsc/`.

P. Bjørstad, J. Koster, and P. Krzyżanowski. DD solvers for large scale industrial finite element problems. In *PARA 2000*, volume 1947 of *LNCS*. Springer, 2001.

X.-C. Cai and D. Keyes. Nonlinearly preconditioned inexact Newton algorithms. *SIAM J. Sci. Comput.*, 24:183–200, 2002.

X.-C. Cai, D. E. Keyes, and D. P. Young. A nonlinear additive Schwarz preconditioned inexact Newton method for shocked duct flow. In *Proc. D.D. Methods-13*, 2001.

J. Demmel, J. Gilbert, and X. Li. SuperLU v. 2.0 code and documentation. URL `http://crd.lbl.gov/~xiaoye/SuperLU/`.

C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, and D. Rixen. FETI-DP: a dual-primal unified FETI method–part I. *IJNME*, 50:1523–1544, 2001.

M. P. I. Forum. MPI: A message-passing interface standard. *Inter. J. Supercomputing Apps.*, 8(3/4), 1994.

G. Karypis and V. Kumar. METIS: v. 4.0 code and documentation. URL `http://www-users.cs.umn.edu/~karypis/metis`.

D. Keyes. DD methods for PDEs. *SIAM J. Sci. Statistic. Comput.*, 13:967–993, 1992.

D. Keyes. Aerodynamic applications of NKS solvers. In *Proceedings of the 14th Conf. Numer. Methods in Fluid Dynamics*, pages 1–20, Berlin, 1995.

J. Kim, N. Aluru, and D. Tortorelli. Improved MLN solvers for fully-coupled multi-physics problems. *IJNME*, 58, 2003.

D. Knoll and D. Keyes. Jacobian-free Newton-Krylov methods: A survey of approaches and applications. *Journal of Comp. Physics*, 2002.

Y. Saad and M. H. Schultz. GMRES: An algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.

B. Smith, P. Bjørstad, and W. Gropp. *D.D, Parallel multilevel methods for elliptic PDEs*. Cambridge University Press, 1996.