SIAM J. SCI. COMPUT.
Vol. 44, No. 3, pp. A1579–A1605

A NONLINEAR ELIMINATION PRECONDITIONED INEXACT NEWTON ALGORITHM*

LULU LIU†, FENG-NAN HWANG‡, LI LUO§, XIAO-CHUAN CAI§, AND DAVID E. KEYES¶

Abstract. A nonlinear elimination preconditioned inexact Newton (NEPIN) algorithm is proposed for problems with localized strong nonlinearities. Due to unbalanced nonlinearities ("nonlinear stiffness"), the traditional inexact Newton method often exhibits a long plateau in the norm of the nonlinear residual or even fails to converge. NEPIN implicitly removes the components causing trouble for the global convergence through a correction based on nonlinear elimination within a subspace that provides a modified direction for the global Newton iteration. Numerical experiments show that NEPIN can be more robust than global inexact Newton algorithms and maintain fast convergence even for challenging problems, such as full potential transonic flows. NEPIN complements several previously studied nonlinear preconditioners with which it compares favorably experimentally on a classic shocked duct flow problem considered herein. NEPIN is shown to be fairly insensitive to mesh resolution and "bad" subproblem identification based on the local Mach number or the local nonlinear residual for transonic flow over a wing.

Key words. nonlinear elimination, inexact Newton method, domain decomposition, nonlinear equations, full potential equation

AMS subject classifications. 65H20, 65N06, 65N22, 65Y05,76H05

DOI. 10.1137/21M1416138

1. Introduction.

Nonlinear preconditioning is a globalization technique for Newton's method applied to systems of algebraic equations with unbalanced nonlinearities. Without a good initial iterate, the norm of the nonlinear residual may plateau for many iterations until a domain of superlinear convergence is found, progress being limited by damping necessitated by a small number of components. Nonlinear preconditioning is motivated by analogy to linear preconditioning. The goal of linear preconditioning is to reduce the condition number or cluster the spectrum of a linear system by (formal) pre- or post-multiplication by a preconditioning matrix, which may be thought of as changing the equations or changing the unknowns, respectively.

Nonlinear preconditioning techniques handle unbalanced nonlinearities by applying a nonlinear transformation on the left- or right-hand side of the original nonlinear function, replacing the nonlinear functions or the unknowns, respectively. Left nonlinear preconditioners lead to a system with the same root as the original system, which is solved by an outer Jacobian-free Newton method [26]. Examples include the ad-

*Submitted to the journal's Methods and Algorithms for Scientific Computing section April 29, 2021; accepted for publication (in revised form) February 28, 2022; published electronically June 21, 2022.
https://doi.org/10.1137/21M1416138

Funding: The work of the first author was supported by the National Natural Science Foundation of China (11901296) and by the Natural Science Foundation of Jiangsu (BK20180450). The work of the third author was supported by the National Natural Science Foundation of China (11701547).

†School of Mathematics and Statistics, Nanjing University of Science and Technology, Nanjing, 210094, China (lulu.liu@njust.edu.cn).

‡Department of Mathematics, National Central University, Jhongli District, Taoyuan City 320317, Taiwan (hwangf@math.ncu.edu.tw).

§Faculty of Science and Technology, University of Macau, Macau, China (liluo@um.edu.mo, xccai@um.edu.mo).

¶Extreme Computing Research Center, King Abdullah University of Science and Technology (KAUST), Thuwal, 23955-6900, Saudi Arabia (david.keyes@kaust.edu.sa).

ditive (and multiplicative) Schwarz preconditioned inexact Newton methods ASPIN (MSPIN) [1, 3, 5, 22, 38] and two-level ASPIN [6, 34] and MSPIN [28, 29, 30], and the restricted nonlinear Schwarz preconditioners RASPEN [12, 16] and SRASPEN [9]. As with linear preconditioning, a left-preconditioned Jacobian is generally not formed explicitly, it being typically much denser than the original; only the matrix-vector multiplication is provided for Krylov subspace methods.

On the other hand, right nonlinear preconditioners are often associated with a nonlinear elimination (NE) [27] procedure, such as that described in [10] and in nonlinearly preconditioned inexact Newton methods [14, 20, 30, 40], which have been applied effectively to such challenging problems as incompressible Navier–Stokes equations at high Reynolds numbers [32], blood flow in branching arteries [33], and two-phase flow in porous media [41]. Many variants also attract increasing attention, such as nonlinear FETI (finite element tearing and interconnecting) [35], nonlinear FETI-DP (FETI-dual primal) [23, 24, 25] and nonlinear BDDC (balancing domain decomposition by constraints) [23, 25].

In this paper, we propose a new left-preconditioned algorithm named NEPIN (nonlinear elimination preconditioned inexact Newton) and demonstrate it for the classical challenging problem of full potential transonic flow. Whereas the right nonlinear preconditioner based on NE referred to as INB-NE [19, 20, 32, 40] provides an improved starting point for the global Newton iteration by eliminating within a subspace, NEPIN applies nonlinear elimination on the left-hand side and provides a modified Newton direction via a subspace correction. It can thus be regarded as a variant of the left-preconditioned ASPIN but without its major drawback. NEPIN overcomes the difficulty of a densified Jacobian that is defined only implicitly, thus maintaining wide scope for favored linear preconditioning techniques in solving for the Newton correction.

In section 2, we present implementation details of the NEPIN algorithm, as well as an illustrative example of two scalar components, on which it is compared to a global inexact Newton method with backtracking (INB). We discuss the relationship between NEPIN and an NE preconditioned version of INB (INB-NE) in section 2.4, between NEPIN and ASPIN in section 2.5, and between NEPIN and RASPEN in section 2.6. In section 3 we consider one-dimensional and two-dimensional problems from transonic aerodynamics, and develop physical insight on the selection of the degrees of freedom to be eliminated. Conclusions and future directions follow in section 4.

2. The NEPIN algorithm.

Consider a nonlinear system of equations $F : \hat{D} \subset R^n \to R^n$, where we seek a vector $x^* \in R^n$ such that

$$F(x^*) = 0. \tag{2.1}$$

Here $F(x) = [F_1, F_2, \ldots, F_n]^T$, $F_i = F_i(x) = F_i(x_1, \ldots, x_n)$, and $x = [x_1, x_2, \ldots, x_n]^T$. Let $J(x)$ be the Jacobian of the nonlinear system $F(x)$. INB, as recalled in Algorithm 2.1, is commonly used for solving such systems. The parameter η_k controls how accurately the Jacobian linear system needs to be solved. In this paper, we choose a fixed value for η_k. Some adaptive choices of η_k can be found in [13]. Parameter $\lambda^{(k)}$ is obtained from a standard backtracking line-search technique [11]. It determines a step size along the inexact Newton direction $d^{(k)}$ such that

$$\hat{f}(x^{(k)} - \lambda^{(k)} d^{(k)}) \leqslant \hat{f}(x^{(k)}) - \alpha \lambda^{(k)} F(x^{(k)})^T J(x^{(k)}) d^{(k)}, \tag{2.2}$$

Algorithm 2.1 INB.

```
Specify the initial guess $x^{(0)}$ and $k = 0$.
while $\|F(x^{(k)})\| > \epsilon_{global-nonlinear-rtol}\|F(x^{(0)})\|$ do
    1. Find the inexact Newton direction $d^{(k)}$ such that
       (2.3)  $\|F(x^{(k)}) - F'(x^{(k)})d^{(k)}\| \leqslant \eta_k\|F(x^{(k)})\|, \quad \eta_k \in (0, 1).$
    2. Determine the damping parameter $\lambda^{(k)} \in (0, 1]$ using a backtracking line
       search [11] along the direction $d^{(k)}$.
    3. The approximate solution at the $k+1$ iteration is given by
       $x^{(k+1)} = x^{(k)} - \lambda^{(k)}d^{(k)},$
       and set $k = k + 1$.
end while
```

where the merit function $\hat{f}(x) = \frac{1}{2}\|F(x)\|^2$, $J(x) = F'(x)$, and α is a small scalar (herein 10^{-4}). As described in [5], the value of $\lambda^{(k)}$ is often determined by the components with the strongest nonlinearities, and too small values for the $\lambda^{(k)}$ lead to a long period of small reduction in norm of the nonlinear residual. Borrowing the idea from NE [27], we introduce a subspace correction phase to handle these "bad" components in the following subsection. In contrast to the modification of the variable $x^{(k)}$ in the NE algorithm, we replace $d^{(k)}$ by a modified Newton direction in (2.2), thus allowing an increased step length.

2.1. The modified Newton direction. Variables are partitioned into two groups and labeled as x_b and x_g, i.e., $x = [x_b, x_g]^T$, and the corresponding nonlinear system $F : R^n \to R^n$ is conformally split as

$$F(x) = F(x_b, x_g) = \begin{bmatrix} F_b(x_b, x_g) \\ F_g(x_b, x_g) \end{bmatrix}, \quad x_b \in R^{n_b},\ x_g \in R^{n_g}, \tag{2.4}$$

where $n_b + n_g = n$ and x_b and x_g are bad and "good" components, respectively.

The NE preconditioned system

$$\mathcal{F}(x) = \mathcal{F}(x_b, x_g) = \begin{bmatrix} T_b(x_b, x_g) \\ F_g(x_b, x_g) \end{bmatrix} = \begin{bmatrix} T_b(x) \\ F_g(x) \end{bmatrix} = 0 \tag{2.5}$$

is obtained by solving

$$F_b(x_b - T_b(x), x_g) = 0 \tag{2.6}$$

for $T_b(x)$. Taking the derivatives of (2.6) with respect to x_b and x_g yields

$$\left(\frac{\partial F_b}{\partial u_b}\right)\left(I_b - \frac{\partial T_b}{\partial x_b}\right) = 0 \tag{2.7}$$

and

$$-\frac{\partial F_b}{\partial u_b}\frac{\partial T_b}{\partial x_g} + \frac{\partial F_b}{\partial x_g} = 0, \tag{2.8}$$

where $u_b = x_b - T_b(x)$ and $I_b \in R^{n_b \times n_b}$ is the identity matrix that has the same dimension as the x_b block. Assuming that $\frac{\partial F_b}{\partial u_b}$ is nonsingular, solving (2.7) and (2.8), and concatenating the results columnwise, we obtain

$$\frac{\partial T_b}{\partial x} = \left(\frac{\partial F_b}{\partial u_b}\right)^{-1} \begin{bmatrix} \frac{\partial F_b}{\partial u_b} & \frac{\partial F_b}{\partial x_g} \end{bmatrix} \tag{2.9}$$

from which it follows that the Jacobian of $\mathcal{F}(x)$ can be written as

$$\mathcal{J}(x) = \begin{bmatrix} \left(\frac{\partial F_b}{\partial u_b}\right)^{-1} & \\ & I_g \end{bmatrix} \begin{bmatrix} \frac{\partial F_b}{\partial u_b} & \frac{\partial F_b}{\partial x_g} \\ \frac{\partial F_g}{\partial x_b} & \frac{\partial F_g}{\partial x_g} \end{bmatrix}, \tag{2.10}$$

where $I_g \in R^{n_g \times n_g}$ is the identity matrix that has the same dimension as the x_g block. In a practical implementation, it is more convenient and cheaper to employ the following approximation,

$$\mathcal{J}(x) \approx \hat{\mathcal{J}}(x) = \begin{bmatrix} J_b^{-1} & \\ & I_g \end{bmatrix} J(u_b, x_g), \qquad J_b = R_b J(u_b, x_g) R_b^T, \tag{2.11}$$

where R_b is an $n_b \times n$ restriction matrix that extracts the bad components. Now we consider the Newton direction equation associated with the preconditioned system, namely,

$$\hat{\mathcal{J}}(x)\hat{d} = \mathcal{F}(x). \tag{2.12}$$

The coefficient matrix corresponds to the block diagonally preconditioned linear Jacobian system as shown in (2.10). Generally speaking, the block matrix

$$\left(\frac{\partial F_b}{\partial u_b}\right)^{-1} \frac{\partial F_b}{\partial x_g}$$

is dense and expensive to compute and store, even when $\frac{\partial F_b}{\partial u_b}$ and $\frac{\partial F_b}{\partial x_g}$ are sparse. To overcome this difficulty, we multiply (2.12) by the inverse of the block diagonal matrix in (2.11), obtaining

$$J(u_b, x_g)\hat{d} = \begin{bmatrix} J_b T_b(x) \\ F_g(x) \end{bmatrix}, \qquad u_b = x_b - T_b(x), \tag{2.13}$$

which is equivalent to (2.12) but invites conventional linear preconditioning techniques of two or more levels, such as restricted additive Schwarz [8], BoomerAMG [17], and others [39]. This differs from the Newton direction that would be obtained directly from (2.4) in the upper portion of the right-hand side, where $F_b(x)$ has been replaced with $J_b T_b(x)$. The extra cost per step is an elimination of the bad degrees of freedom to obtain T_b, and a (generally) sparse matrix-vector multiplication.

Remark 2.1. In a neighborhood U of the exact solution x^*, we assume that the Jacobian $J(x) = F'(x)$ is continuous, and $J(x^*)$ and $R_b J(x^*) R_b^T$ are invertible. From the Taylor expansion of the function $F_b(x)$,

$$F_b(x) = F_b(x_b, x_g) = F_b(x_b - T_b(x), x_g) + J_b(x_b - T_b(x), x_g)T_b(x) + o(\|T_b(x)\|), \tag{2.14}$$

it follows from (2.6) that

$$F_b(x) = J_b(x_b - T_b(x), x_g)T_b(x) + o(\|T_b(x)\|). \tag{2.15}$$

Following [1, 29], the function $T_b(x)$ is continuous and $T_b(x^*) = 0$. Due to the continuity of $J(x)$ and $T_b(x)$, using (2.13) and (2.15), we deduce that

$$J(x_b - T_b(x), x_g) \to J(x), \quad \begin{bmatrix} J_b T_b(x) \\ F_g(x) \end{bmatrix} \to \begin{bmatrix} F_b(x) \\ F_g(x) \end{bmatrix} = F(x), \quad \text{as } x \to x^*. \tag{2.16}$$

Let d be the Newton direction satisfying

$$J(x)d = F(x), \tag{2.17}$$

which implies the difference between the Newton direction and the modified Newton direction vanishes in the neighborhood of x^*, i.e., $\hat{d} \to d$ as $x \to x^*$.

2.2. The basic algorithm.

Algorithm 2.2 defines the NEPIN method. The key idea is to replace the Newton direction in INB by a modification. We set $x^{(0)}$ as the initial guess. At the kth iteration, the current approximate solution is denoted by $x^{(k)}$ and $x^{(k+1)}$ is the new approximate solution.

Some practical remarks about Algorithm 2.2 follow:

(1) The subsets $S_b^{(0)}$ and $S_g^{(0)}$ are predetermined for the initial partition of the index set $S = \{1, 2, \ldots, n\}$. If we do not determine which components to eliminate beforehand in the phase of the subspace correction, we can simply

Algorithm 2.2 NEPIN.

```
Specify the initial guess $x^{(0)}$ and $k = 0$.
Initialize the partition: $S_b^{(0)}$ and $S_g^{(0)}$.
while $\|F(x^{(k)})\| > \epsilon_{global-nonlinear-rtol}\|F(x^{(0)})\|$ do
    Step 1 (Subspace correction): Start from the initial guess $T_{b,0}^{(k)} = 0$, $i = 1, 2, \ldots, N$,
    find $T_b^{(k)}$ by solving the following subproblems:
    (2.18)    $F_b(x_b^{(k)} - T_b^{(k)}, x_g^{(k)}) = 0$,
    and form the global residual
    (2.19)    $g^{(k)} = \begin{bmatrix} J_b^{(k)} T_b^{(k)} \\ F_g \end{bmatrix}, \quad J_b^{(k)} = R_b J(x_b^{(k)} - T_b^{(k)}, x_g^{(k)}) R_b^T$.
    Step 2: Find the inexact Newton direction $\hat{d}^{(k)}$ by approximately solving
    (2.20)    $J(x_b^{(k)} - T_b^{(k)}, x_g^{(k)})\hat{d}^{(k)} = g^{(k)}$.
    Step 3: Compute the new approximate solution.
    (2.21)    $x^{(k+1)} = x^{(k)} - \lambda^{(k)}\hat{d}^{(k)}$,
    where the step length $\lambda^{(k)}$ is determined by a backtracking line search [11] based
    on the merit function $\hat{f}(x) = \frac{1}{2}\|F(x)\|^2$.
    Set $k = k + 1$ and determine a new partition $S = S_b^{(k)} \bigcup S_g^{(k)}$.
end while
```

implement one-step standard Newton iteration by setting $S_b^{(0)} = \emptyset$ and $S_g^{(0)} = S$.

(2) The choice of bad components affects the quality of the preconditioning and is crucial to success of the algorithm. As the dimension of the subspace problem increases, it exacts more computational cost. However, if enough bad components are not removed, the subsequent global Newton solver may fail.

(3) The global nonlinear problem is iterated upon by INB until

$$\|F(x^{(k)})\| \leqslant \epsilon_{global-nonlinear-rtol}\|F(x^{(0)})\|. \tag{2.22}$$

The Jacobian system in (2.20) is solved by a Krylov subspace method, such as GMRES [36], combined with the right restricted additive Schwarz (RAS) preconditioner. The RAS preconditioner employs a full restriction operator, but ignores the off-process values during interpolation [8]. We define the restriction operator as R_i^δ, where δ is the size of overlap. The linear solver is stopped when

$$\|g^{(k)} - J(x_b^{(k)} - T_b^{(k)}, x_g^{(k)})M_{RAS}^{-1}(M_{RAS}\hat{d}^{(k)})\| \leqslant \epsilon_{global-linear-rtol}\|g^{(k)}\|,$$

where

$$M_{RAS}^{-1} = \sum_{i=1}^{N}(R_i^0)^T J_i^{-1} R_i^\delta, \quad J_i = R_i^\delta J(x_b^{(k)} - T_b^{(k)}, x_g^{(k)})(R_i^\delta)^T. \tag{2.23}$$

(4) At the kth iteration, we define the subspace nonlinear problem according to [33] as follows:

$$G(z) = \begin{bmatrix} F_b(z) \\ R_g z - x_g^{(k)} \end{bmatrix} = 0, \tag{2.24}$$

where R_g is an $n_g \times n$ restriction matrix that extracts the good components. These subspace problems are solved by INB starting from $z^{(0)} = x^{(k)}$ and the stopping condition is set as

$$\|G(z^{(p)})\| \leqslant \epsilon_{sub-nonlinear-rtol}\|G(z^{(0)})\|. \tag{2.25}$$

The nonlinear correction $T^{(k)}$ is obtained from the solution $z^{(p)}$ of the subspace problem. The Jacobian systems of subspace problems are also solved using GMRES and the right RAS preconditioner as the global Jacobian system.

(5) From the Taylor expansion of the function $F(x^{(k)} + p^{(k)})$,

$$F(x^{(k)} + p^{(k)}) = F(x^{(k)}) + F'(x_k)p^{(k)} + o(\|p^{(k)}\|), \tag{2.26}$$

it is easy to see that the Newton direction $p^{(k)}$ is reliable only when the difference between $F(x^{(k)} + p^{(k)})$ and the linear model $F(x^{(k)}) + F'(x_k)p^{(k)}$ is not too large. For problems with the local high nonlinearity, the high-order terms in (2.26) dominate and the Newton direction is not a good downhill direction, leading to slow convergence of INB. The difference between INB and NEPIN lies in different descent directions. The modified Newton direction corresponding to the nonlinearly preconditioned system in Step 2 of Algorithm 2.2 is crucial to accomplishing fast convergence.

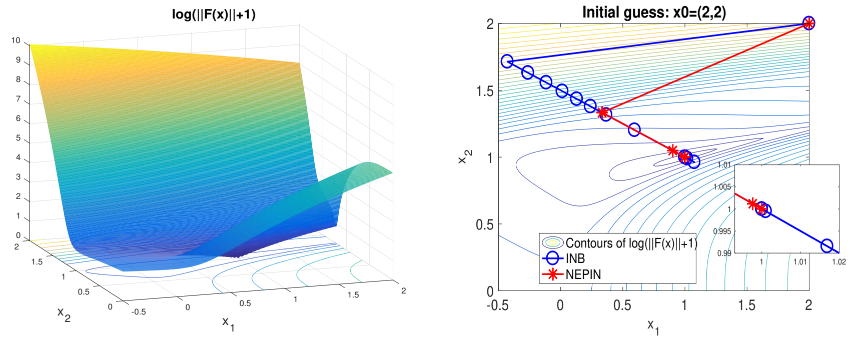

FIG. 1. *Contours of* $\log(\|F(x)\|+1)$ *for* $m=5$ *and the path using INB and NEPIN from the starting point* $x_0=[2,2]^T$*. The blue circles and red stars represent the intermediate solution* $x^{(k)}$ *corresponding to INB and NEPIN, respectively.*

(6) In the linear case, Algorithm 2.2 is the same as the block Jacobi linear preconditioning. If we have the linear system

$$F(x) = Ax - b, \tag{2.27}$$

where

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \tag{2.28}$$

then

$$\mathcal{F}^{\text{NEPIN}}(x) = M^{-1}(Ax-b), \qquad M = \begin{bmatrix} A_{11} & 0 \\ & I_{22} \end{bmatrix}, \tag{2.29}$$

where I_{22} is the identity matrix that has the same dimension as A_{22}.

2.3. A simple illustrative example. Consider the system of two equations in two unknowns with polynomial nonlinearity in the first that can be made arbitrarily steep by increasing the exponent m.

$$F(x_1,x_2) = \begin{bmatrix} F_1(x_1,x_2) \\ F_2(x_1,x_2) \end{bmatrix} = \begin{bmatrix} (x_1 - x_2^3 + 1)^m - x_2^m \\ x_1 + 2x_2 - 3 \end{bmatrix} = 0. \tag{2.30}$$

This is a test problem considered in other contexts [31, 40] to illustrate differences between nonlinear preconditioning techniques. It is easy to verify that $x^* = [1,1]^T$ is a root of this system. As shown in the left plot of Figure 1, the exact solution x^* lies in a narrow valley and the contours $\log(\|F(x)\|+1)$ near x^* tend towards highly eccentric ellipses. By setting

$$F_1(x_1 - T_1(x_1,x_2), x_2) = 0, \tag{2.31}$$

we can solve explicitly for $T_1(x_1,x_2)$, because of the algebraic simplicity of the system, and derive nonlinearly preconditioned systems as follows:

$$\mathcal{F}(x_1,x_2) = \begin{bmatrix} T_1(x_1,x_2) \\ F_2(x_1,x_2) \end{bmatrix} = \begin{bmatrix} x_1 - x_2^3 + 1 - x_2 \\ x_1 + 2x_2 - 3 \end{bmatrix} = 0. \tag{2.32}$$

TABLE 1
The number of nonlinear iterations staring from different points. The relative tolerance for the global Newton iterations is set to $\epsilon_{global-nonlinear-rtol} = 10^{-8}$.

	INB			NEPIN		
Initial guess x_0	$m = 1$	$m = 3$	$m = 5$	$m = 1$	$m = 3$	$m = 5$
$x_0 = [0,0]^T$	5	8	10	5	6	6
$x_0 = [0,2]^T$	5	10	12	5	5	4
$x_0 = [2,0]^T$	5	1	7	5	6	6
$x_0 = [2,2]^T$	5	12	13	5	5	4

Algorithm 2.3 One step of INB-NE.

Given the partitioning $x^{(k)} = [x_b^{(k)}, x_g^{(k)}]^T$.

1. Compute a shifted starting point $\tilde{x}^{(k)} = G(x^{(k)}) = [x_b^{(k)} - T_b^{(k)}, x_g^{(k)}]^T$ by means of a subspace correction such that $F_b(x_b^{(k)} - T_b^{(k)}, x_g^{(k)}) = 0$.
2. Find the inexact Newton direction $d^{(k)}$ such that

$$\|F(\tilde{x}^{(k)}) - F'(\tilde{x}^{(k)})d^{(k)}\| \leqslant \eta_k \|F(\tilde{x}^{(k)})\|. \tag{2.33}$$

3. Update $x^{(k+1)} = \tilde{x}^{(k)} - \lambda^{(k)} d^{(k)}$, where $\lambda^{(k)} \in (0,1]$ is the damping parameter determined by a line search along $d^{(k)}$.

It is noted that preconditioned functions $\mathcal{F}(x_1, x_2)$ are independent of m.

In numerical tests, we choose four different initial guesses the same distance away from the exact solution x^*, i.e., $x_0 = [0,0]^T$, $x_0 = [0,2]^T$, $x_0 = [2,0]^T$, $x_0 = [2,2]^T$. Corresponding to $m = 1, 3, 5$, Table 1 shows the number of Newton iterations for solving INB and NEPIN, respectively. In contrast to INB, NEPIN is not very sensitive to initial estimates and the parameter m. The right plot of Figure 1 shows the respective paths of INB and NEPIN starting from the initial guess $x_0 = [2,2]^T$. For INB, the Newton direction at the first iteration is nearly parallel to the contours on the steep hillside and then it takes 8 iterations to approach the bottom of the narrow valley. However, it requires just 2 iterations for NEPIN, with its modified first step, to go down the steep hill and enter the neighborhood of the exact solution x^*.

2.4. Comparison with INB-NE. Right preconditioning changes the coordinates of the solution, rather than of the residual, as in the understanding of "right" in linear preconditioning; however, the switch from left to right is not as straightforward as specifying where to apply a linear transformation in the nonlinear case. Our quick review of the INB-NE algorithm follows [40].

Algorithm 2.3 gives details of one step of the INB-NE algorithm. Observe the following:

(1) Step 1 of Algorithm 2.3 requires the same NE as in Step 1 of NEPIN (2.18). For INB-NE, it is used to generate a new starting point before each global Newton direction instead of replacing components of the right-hand side for the Jacobian systems.
(2) Figure 2 plots the convergence histories for the example of section 2.3 zoomed in to a consistent scale for the last few Newton steps for all four starting guesses and both methods, together with their full residual convergence his-

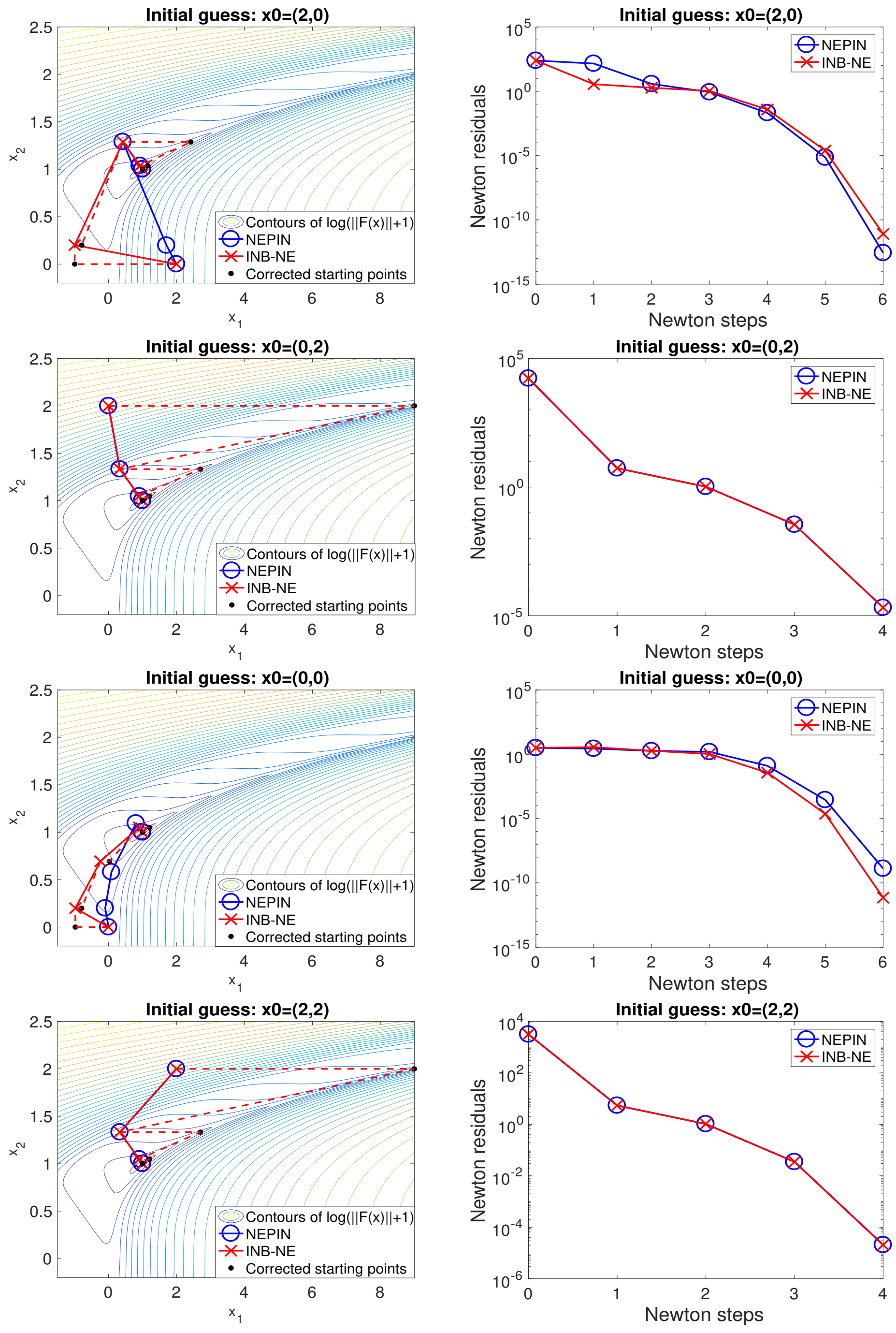

FIG. 2. *Comparison between NEPIN and INB-NE for the example with* $m = 5$ *in subsection 2.3. The left column shows the final stages of convergence using NEPIN and INB-NE, and the blue circles and red x-marks represent the intermediate solution* $x^{(k)}$ *corresponding to NEPIN and INB-NE, respectively. For INB-NE, the black dot represents the starting point after the subspace correction phrase. The right column shows the norm of the Newton residuals.*

Algorithm 2.4 One-step of ASPIN.

```
Given $x^{(k)}$.
1. Solve local problems $F_{\Omega_i}(x^{(k)} - g_i^{(k)}) = 0, \quad i = 1, \dots, N$ for $g_i^{(k)}$, starting from the initial guess $g_i^{(k)} = 0$.
2. Let $\mathcal{F}^{\text{ASPIN}}(x^{(k)}) = \sum_{i=1}^N g_i^{(k)}$ and find the inexact Newton direction $d^{(k)}$ such that
   (2.34)   $\|\mathcal{F}^{\text{ASPIN}}(x^{(k)}) - \hat{\mathcal{J}}^{\text{ASPIN}}(x^{(k)})d^{(k)}\| \leqslant \eta_k \|\mathcal{F}^{\text{ASPIN}}(x^{(k)})\|.$
3. Update $x^{(k+1)} = x^{(k)} - \lambda^{(k)} d^{(k)}$, where $\lambda^{(k)} \in (0, 1]$ is the damping parameter determined by a line search based on the merit function $f(x) = \frac{1}{2}\|\mathcal{F}^{\text{ASPIN}}(x)\|^2$.
```

tories. NEPIN and INB-NE require essentially the same number of Newton iterations for convergence starting from the same point. For a fixed number of iterations, NEPIN reduces the norm of the nonlinear residual slightly better in the first case, INB-NE in the third case; they perform essentially the same from the second and fourth starting points. The zigzag dashed red lines show how INB-NE shifts the starting vector for each Newton step to a point lower in the valley of the merit function.

(3) The computational costs are roughly the same for each Newton iteration using NEPIN and INB-NE since each of the three steps of elimination, global linear solve, and line-search update have comparable cost.

2.5. Comparison with ASPIN. We consider the ASPIN version [5] of left nonlinear preconditioning introduced for decompositions into subdomains, i.e., $\Omega = \bigcup_{i=1}^N \Omega_i$. In the ASPIN algorithm, a Jacobian-free inexact Newton algorithm is used to solve the nonlinearly preconditioned system

$$\mathcal{F}^{\text{ASPIN}}(x) = \sum_{i=1}^N g_i(x) = 0, \tag{2.35}$$

where $g_i(x)$ is obtained by solving the local nonlinear system

$$F_{\Omega_i}(x - g_i(x)) = 0, \quad i = 1, 2, \dots, N. \tag{2.36}$$

Following [1, 5, 38], the exact Jacobian of $\mathcal{F}^{\text{ASPIN}}$ can be written as

$$\mathcal{J}^{\text{ASPIN}}(x) = \sum_{i=1}^N R_i^T [R_i J(x - g_i(x)) R_i^T]^{-1} R_i J(x - g_i(x)), \tag{2.37}$$

where R_i is the restriction operator corresponding to the subdomain Ω_i. It is recommended in [5] to use the following approximation:

$$\hat{\mathcal{J}}^{\text{ASPIN}}(x) = \sum_{i=1}^N R_i^T [R_i J(x) R_i^T]^{-1} R_i J(x). \tag{2.38}$$

The approximate Jacobian $\hat{\mathcal{J}}_{\text{ASPIN}}$ is never formed explicitly, but we instead provide matrix-vector multiplications in Krylov subspace iterative methods.

Algorithm 2.4 gives details of one step of the ASPIN algorithm. Observe the following:

(1) The domain partition of ASPIN generally does not change during Newton iterations, but the set of bad components may be different for NEPIN.

(2) Step 2 of Algorithm 2.4 requires the Newton direction corresponding to the nonlinearly preconditioned system as in Step 2 of NEPIN. However, the line search for NEPIN and ASPIN is based on different merit functions as shown in step 3 of Algorithm 2.4 and NEPIN.

(3) For ASPIN, we consider the partition into two nonoverlapping subdomains, i.e., $\Omega = \Omega_b \bigcup \Omega_g$, $x^{(k)} = [x_b^{(k)}, x_g^{(k)}]$, $g_b^{(k)} = [T_b^{(k)}, \mathbf{0}]^T$, and $g_g^{(k)} = [\mathbf{0}, T_g^{(k)}]^T$. The variables and equations can be written in the form

$$F_b(x_b^{(k)} - T_b^{(k)}, x_g^{(k)}) = 0, \qquad F_b(x_b^{(k)}, x_g^{(k)} - T_g^{(k)}) = 0, \tag{2.39}$$

and it is easy to get the nonlinearly preconditioned function

$$\mathcal{F}^{\text{ASPIN}}(x^{(k)}) = \begin{bmatrix} T_b^{(k)} \\ T_g^{(k)} \end{bmatrix} \tag{2.40}$$

and the approximate Jacobian

$$\hat{\mathcal{J}}^{\text{ASPIN}}(x^{(k)}) = \begin{bmatrix} J_b^{-1} & \\ & J_g^{-1} \end{bmatrix} J(x^{(k)}), \tag{2.41}$$

where $J_b^{(k)} = R_b J(x^{(k)}) R_b^T$ and $J_g^{(k)} = R_g J(x^{(k)}) R_g^T$. In step 2 of Algorithm 2.4, we can rewrite the Jacobian system into the equivalent form

$$J(x^{(k)}) d^{(k)} = \begin{bmatrix} J_b T_b^{(k)} \\ J_g T_g^{(k)} \end{bmatrix} \tag{2.42}$$

as in Step 2 of of NEPIN (2.20), which allows the application of linear preconditioning techniques.

2.6. Comparison with RASPEN. We briefly recall the restricted additive Schwarz preconditioned exact Newton (RASPEN) version [12] of left nonlinear preconditioning introduced for decompositions into subdomains, i.e., $\Omega = \bigcup_{i=1}^N \Omega_i$. Let R_i and $\tilde{R}_i$ be the restriction operators based on the overlapping and nonoverlapping decomposition, respectively. In the RASPEN algorithm, a Jacobian-free inexact Newton algorithm is used to solve the nonlinearly preconditioned system

$$\mathcal{F}^{\text{RASPEN}}(x) = \sum_{i=1}^N \tilde{E}_i g_i(x) = 0, \qquad \tilde{E}_i = \tilde{R}_i^T \tilde{R}_i \in R^{n \times n}, \tag{2.43}$$

where $g_i(x)$ is obtained by solving the local nonlinear system

$$F_{\Omega_i}(x - g_i(x)) = 0, \quad i = 1, 2, \ldots, N. \tag{2.44}$$

Following [12], the exact Jacobian of $\mathcal{F}^{\text{RASPEN}}$ can be written as

$$\mathcal{J}^{\text{RASPEN}}(x) = \sum_{i=1}^N \tilde{R}_i^T [R_i J(x - g_i(x)) R_i^T]^{-1} R_i J(x - g_i(x)). \tag{2.45}$$

Algorithm 2.5 gives details of one step of the RASPEN algorithm. Observe the following:

Algorithm 2.5 One-step of RASPEN.

```
Given $x^{(k)}$.
1. Solve local problems $F_{\Omega_i}(x^{(k)} - g_i^{(k)}) = 0, \quad i = 1, \dots, N$, for $g_i^{(k)}$, starting from the initial guess $g_i^{(k)} = 0$.
2. Let $\mathcal{F}^{\mathrm{RASPEN}}(x^{(k)}) = \sum_{i=1}^{N} \tilde{E}_i g_i^{(k)}$ and find the inexact Newton direction $d^{(k)}$ such that
   (2.46)  $\|\mathcal{F}^{\mathrm{RASPEN}}(x^{(k)}) - \mathcal{J}^{\mathrm{RASPEN}}(x^{(k)}) d^{(k)}\| \leqslant \eta_k \|\mathcal{F}^{\mathrm{RASPEN}}(x^{(k)})\|.$
3. Update $x^{(k+1)} = x^{(k)} - \lambda^{(k)} d^{(k)}$, where $\lambda^{(k)} \in (0, 1]$ is the damping parameter determined by a line search based on the merit function $f(x) = \frac{1}{2}\|\mathcal{F}^{\mathrm{RASPEN}}(x)\|^2$.
```

(1) As with ASPIN, the domain partition of RASPEN generally does not change during Newton iterations, but the set of bad components may be different for NEPIN.

(2) Step 2 of Algorithm 2.5 requires the Newton direction corresponding to the nonlinearly preconditioned system as in Step 2 of NEPIN. However, the line search for NEPIN and RASPEN is based on different merit functions as shown in step 3 of Algorithm 2.5 and NEPIN.

3. Experimental results.

In this section, we apply the NEPIN algorithm to two model boundary value problems dealing with flows ranging from subsonic to transonic regimes. In the first case, we consider a compressible flow through a converging-diverging nozzle. In the second, NEPIN is tested in a two-dimensional transonic flow passing over an airfoil. In each case, a shock with feature size much smaller than the domain and not present in the initial condition must be located and then sharpened up. All the numerical experiments are done in the portable extensible toolkit for scientific computation (PETSc) [2].

3.1. A shocked duct flow.

Consider a compressible flow passing through a straight duct with variable cross-section area [7, 19]. The quasi-one-dimensional model problem is governed by

$$(A(x)\rho(u)\phi_x)_x = 0, \quad 0 < x < 2, \tag{3.1}$$

$$\phi(0) = 0, \ \phi(2) = \phi_R, \tag{3.2}$$

for velocity potential function ϕ, with $u = \phi_x$ being the cross-section area-averaged flow velocity along the x-direction. The duct area $A(x)$ is given by the simple parabola

$$A(x) = 0.4 + 0.6(x-1)^2, \tag{3.3}$$

and the duct has a narrow throat around $x = 1$, as shown in Figure 3 (left).

The dimensionless density function in (3.1) is given by

$$\rho(u) = (c^2)^{\frac{1}{\gamma-1}} = \left(1 + \frac{\gamma-1}{2}(1-u^2)\right)^{\frac{1}{\gamma-1}}, \tag{3.4}$$

where c is the sound speed and we set the ratio of specific heat $\gamma = 1.4$ (that of air in the standard atmosphere) for all tests. The local Mach number is defined as

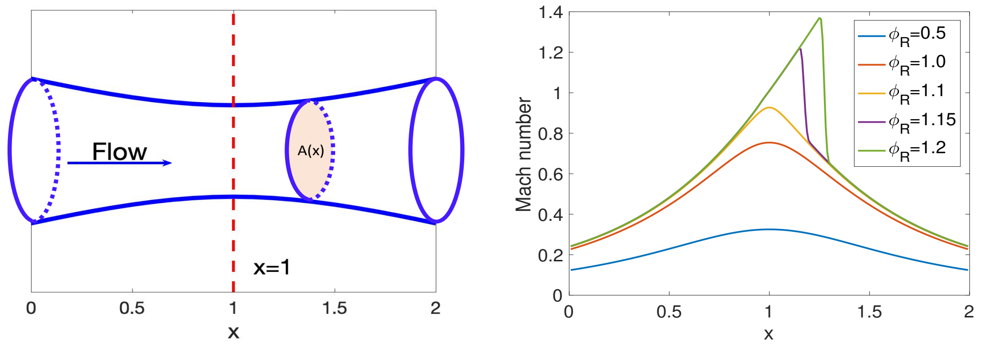

FIG. 3. *The left figure shows the flow passing through a duct with variable cross-section area. The right figure shows the Mach distribution corresponding to different right boundary values ϕ_R in (3.2) on the uniform grid with the grid size $h = \frac{1}{128}$.*

$M = |u|/c$. Figure 3 (right) shows the Mach distribution of the solution corresponding to different values ϕ_R for the right boundary condition. As the value of ϕ_R is increased, the Mach number in the throat becomes larger, and finally a shock sets up by the time ϕ_R reaches 1.15.

Using a standard finite difference method, our goal is to obtain a discrete solution of the governing equation (3.1) and (3.2) on a uniform mesh

$$0 = x_0 < x_1 < \cdots < x_N = 2,$$

where the grid size is $h = \frac{2}{N}$. As described in [7, 19], the discretized nonlinear equations at the interior grid points are given by

$$A_{i+\frac{1}{2}}\tilde{\rho}_{i+\frac{1}{2}}(\phi_{i+1} - \phi_i) - A_{i-\frac{1}{2}}\tilde{\rho}_{i-\frac{1}{2}}(\phi_i - \phi_{i-1}) = 0, \qquad i = 1, 2, \ldots, n-1, \tag{3.5}$$

where $\tilde{\rho}_j$ is the so-called first-order density upwind biasing, i.e.,

$$\tilde{\rho}_{j+\frac{1}{2}} = \rho_{j+\frac{1}{2}} - \mu_j(\rho_{j+\frac{1}{2}} - \rho_{j-\frac{1}{2}}). \tag{3.6}$$

Here, μ_j is a switching function given by

$$\mu_j = \max_{j-2\leqslant i\leqslant j+2} \max\left\{0, 1 - \frac{\hat{M}_c^2}{M_i^2}\right\},$$

where $\hat{M}_c$ is a given cutoff Mach number and M_i is the Mach number at the point x_i. In this paper, we choose a rather tight Mach cutoff $\hat{M}_c = 0.95$ and the Mach number, using (3.4), is

$$M_i = \frac{|u_i|}{\sqrt{1 + \frac{\gamma-1}{2}(1 - u_i^2)}}. \tag{3.7}$$

The value of μ_j in the transonic region depends on the ratio of $\hat{M}_c$ and M_i over the five-point range $i = j-2, j-1, j, j+1, j+2$. In the subsonic regions, $\mu_j = 0$, i.e., no upwinding is applied.

For all tests, the initial iterate is a simple linear interpolation of the boundary conditions, i.e.,

$$\phi^0(x) = \frac{1}{2}x\phi_R, \quad x \in [0, 2].$$

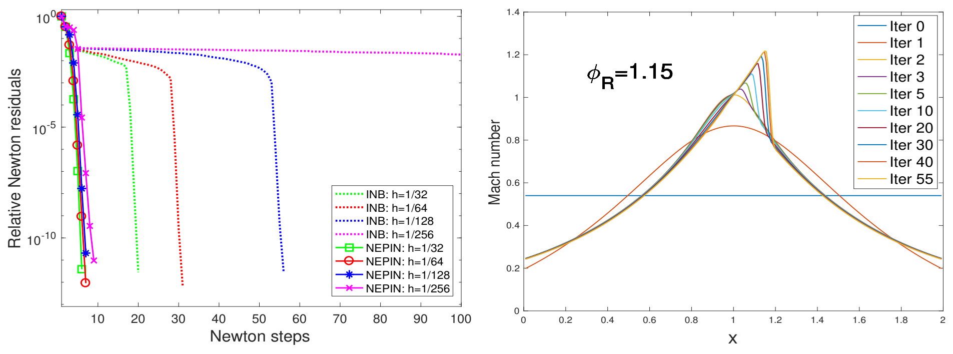

FIG. 4. *For $\phi_R = 1.15$, the left figure shows the history of the Newton residual using INB and NEPIN on problems of different resolution with the mesh size $h = \frac{1}{32}, \frac{1}{64}, \frac{1}{128}, \frac{1}{256}$; the right figure shows the convergence history of Mach number curves for INB when $h = 1/128$.*

TABLE 2
Comparison of the number of nonlinear iterations corresponding to various ϕ_R for INB and NEPIN. "-" indicates that the number of nonlinear iterations is not available due to failure of the line search.

	Its. INB			Its. NEPIN		
$h =$	1/64	1/128	1/256	1/64	1/128	1/256
$\phi_R = 1.10$	6	6	7	5	5	5
$\phi_R = 1.15$	30	55	156	6	6	8
$\phi_R = 1.18$	35	71	-	6	6	7

For NEPIN, we set the initial partition as $S_b^{(0)} = \emptyset$ and $S_g^{(0)} = S$, and define bad components as unknowns whose local Mach numbers exceed a given value, here $M_j > 0.45$, an evolving contiguous range around the shock as it develops. The Jacobian systems for both global and subspace problems are solved by GMRES(30) with RAS preconditioning of overlap 2. The termination tolerances are set as $\epsilon_{global-nonlinear-rtol} = 10^{-10}$, $\epsilon_{global-linear-rtol} = 10^{-3}$, $\epsilon_{sub-nonlinear-rtol} = 10^{-2}$, and $\epsilon_{sub-linear-rtol} = 10^{-3}$. The numerical tests use 4 subdomains.

We first fix $\phi_R = 1.15$ and vary the mesh resolution. The left plot of Figure 4 shows the history of the Newton residual using INB and NEPIN on problems of mesh width 1/32, 1/64, 1/128, and 1/256. It is observed that the convergence of the INB method degenerates as the mesh becomes finer. In effect, the shock "walks" one mesh point per Newton step towards its converged location due to the damping required to stabilize its progress. However, the convergence of the NEPIN method is almost mesh invariant and therefore overwhelmingly superior in the high resolution limit. The right plot of Figure 4 shows the convergence history of Mach number curves on a fixed mesh with $h = 1/128$. We note that it takes only 3 iterations for INB to establish the weak shock but another 48 or so iterations to find the approximately correct location and the correct shock strength, after which it achieves the desired superlinear convergence rate.

We then vary the values of ϕ_R on the right boundary. Table 2 compares the number of nonlinear iterations corresponding to the different ϕ_R for INB and NEPIN. The INB method requires more Newton iterations as the value of ϕ_R is increased on a fixed mesh, and it fails due to failure of the linear search for $\phi_R = 1.18$ with the grid size $h = 1/256$. In contrast, the number of NEPIN iterations is nearly parameter independent.

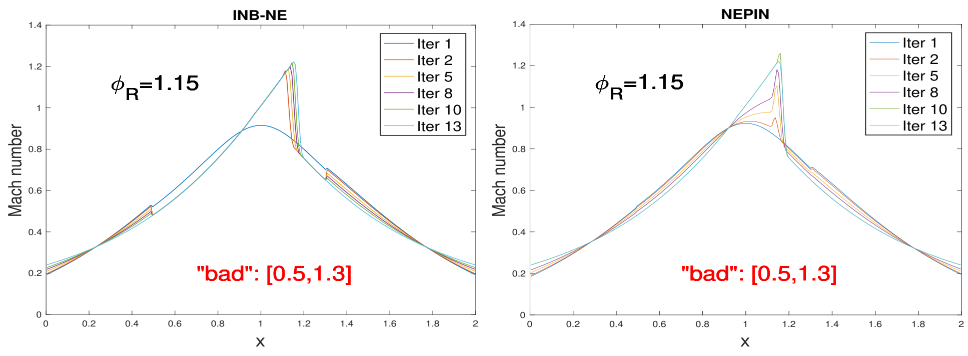

FIG. 5. *For $\phi_R = 1.15$ and $h = 1/128$, the figure shows the convergence history of Mach number curves for INB-NE and NEPIN, respectively. We choose unknowns within the interval* $[0.5, 1.3]$ *as bad components and $\epsilon_{sub-nonlinear-rtol} = 10^{-2}$ for INB-NE and NEPIN.*

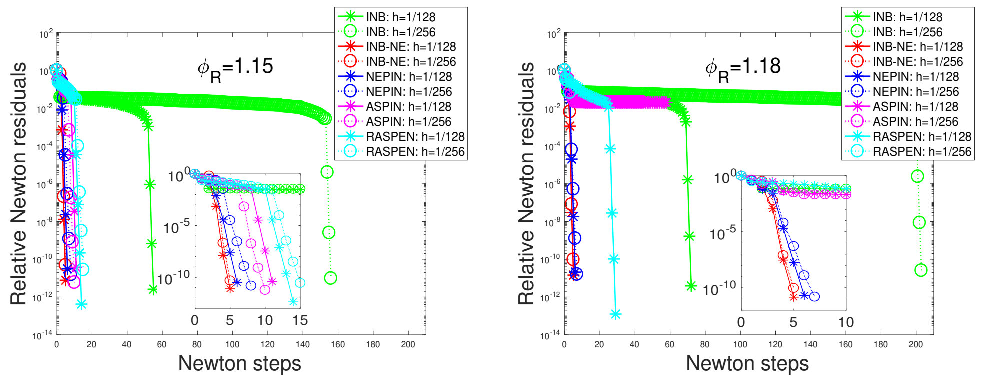

FIG. 6. *For $\phi_R = 1.15$ and $\phi_R = 1.18$, the figure shows the convergence history of the Newton residual using INB, INB-NE, NEPIN, ASPIN, and RASPEN on problems of different resolutions with the grid size $h = 1/128$ and $1/256$, respectively. For ASPIN and RASPEN, we run the test on eight subdomains and set overlaps $5h$ and $10h$ when the grid size $h = 1/128$ and $h = 1/256$, respectively.*

In order to make a fair comparison between INB-NE and NEPIN, we fix the set of bad components (namely, the unknowns within the interval $[0.5, 1.3]$) instead of a dynamic partition of good and bad components based on local Mach numbers. For $\phi_R = 1.15$ and the grid size $h = 1/128$, INB-NE and NEPIN converge within 15 and 14 Newton iterations, respectively. Figure 5 shows the convergence history of Mach number curves for INB-NE and NEPIN corresponding to the fixed sets S_b and S_g, respectively. It is seen that some jumps in the Mach number curves are introduced around the interface $x = 0.5$ and $x = 1.3$ for both INB-NE and NEPIN methods, and NEPIN overshoots before converging.

Finally, in Figure 6, we compare the convergence history of the Newton residual using INB, INB-NE, NEPIN, ASPIN, and RASPEN on problems of different resolutions with the values $\phi_R = 1.15, 1.18$ and the different grid sizes $h = 1/128, 1/256$, respectively. For INB-NE and NEPIN, we still set the initial partition as $S_b^{(0)} = \emptyset$ and $S_g^{(0)} = S$, and choose the bad components whose local Mach numbers are larger than 0.45. We run the tests using ASPIN and RASPEN with eight subdomains, and the

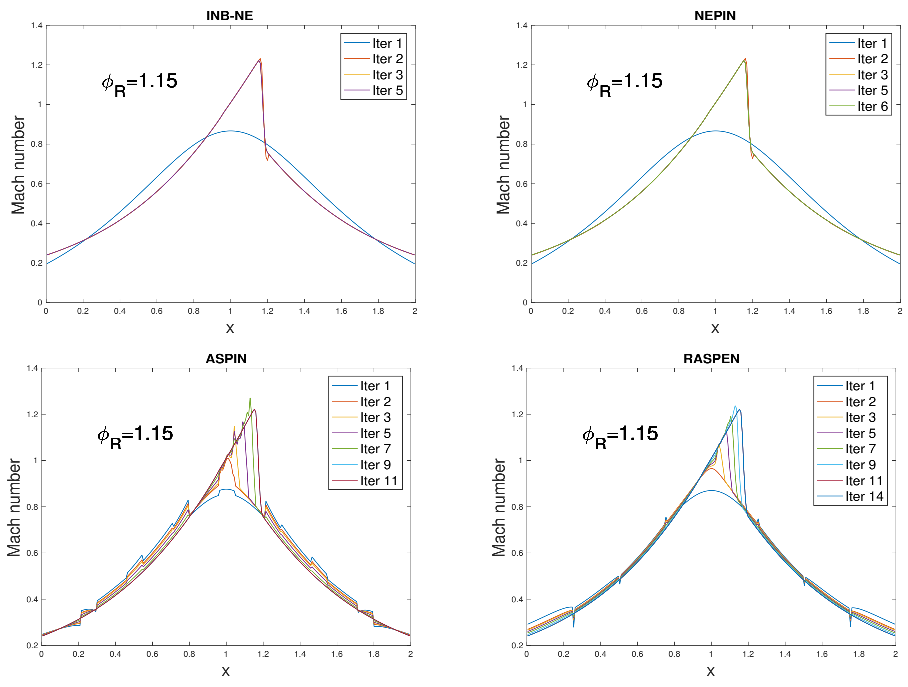

Fig. 7. *For $\phi_R = 1.15$ and $h = 1/128$, the figure shows the convergence history of Mach number curves for INB-NE, NEPIN, ASPIN, and RASPEN using eight processors. For INB-NE and NEPIN, the bad components are identified by the local Mach number ($M_j > 0.45$). We run the tests using ASPIN and RASPEN with eight subdomains, and the overlaps are set to $5h$.*

overlaps are set to $5h$ and $10h$ when the grid size $h = 1/128$ and $h = 1/256$, respectively. For ASPIN and RASPEN, local problems converge if a 10^{-2} relative reduction of the initial residual is satisfied. Compared with INB, as shown in Figure 6, INB-NE, NEPIN, ASPIN, and RASPEN are superior in terms of the number of Newton iterations when $\phi_R = 1.15$. In contrast, ASPIN and RASPEN fail to converge when $\phi_R = 1.18$ and $h = 1/256$, but the other three methods are still convergent. Figure 7 shows the convergence history of Mach number curves for INB-NE, NEPIN, ASPIN, and RASPEN corresponding to $\phi_R = 1.15$ and $h = 1/128$. For ASPIN and RASPEN, we observe some sharp jumps near the interfaces of subdomains and the oscillations in the interval $[1, 1.25]$. Obviously, the bad components are not included in a single overlapping subdomain and it is not easy to solve the subproblem associated with the interval $[1, 1.25]$. In particular, the failure of the corresponding local solver leads to divergence of outer Newton iterations for ASPIN and RASPEN (see Figure 6), when $\phi_R = 1.18$ and $h = 1/256$.

3.2. Transonic full potential flow. We next consider a two-dimensional transonic flow over the upper half of a standard NACA 0012 airfoil at zero angle of attack, the most standard of test cases for aerodynamics codes. The lowest fidelity and easiest to reproduce physical model that is sufficiently difficult to illustrate the advantage of nonlinear preconditioning is transonic full potential flow. This model, which can be derived from the Euler equations, assumes that the flow is inviscid, isentropic, and irrotational [4, 37]. The full potential equation in conservation form with a single unknown function Φ is formulated as:

$$\nabla \cdot (\rho(\Phi)\nabla\Phi) = 0, \tag{3.8}$$

where Φ is the velocity potential and $\nabla\Phi = [u,v]^T$ is the velocity field. The dimensionless density ρ is here given by

$$\rho(\Phi) = \rho_\infty \left(1 + \frac{\gamma-1}{2} M_\infty^2 \left(1 - \frac{\|\nabla\Phi\|_2^2}{q_\infty^2}\right)\right)^{\frac{1}{\gamma-1}}, \tag{3.9}$$

where γ is the ratio of specific heats, and ρ_∞, M_∞, and q_∞ represent the density, the Mach number, and the speed referred to a uniform freestream (at ∞), respectively. Here we set $\gamma = 1.4$, $\rho_\infty = 1.0$, $M_\infty = 0.8$, and $q_\infty = 1.0$ for all tests.

The problem is defined on a square domain $\Omega = [0,1] \times [0,1]$ with the airfoil occupying the middle third of the domain, with the following boundary conditions [4]:

- Along the left inflow boundary ($x = 0$, $0 < y < 1$): $\Phi(x,y) = 0$.
- Along the right exit boundary ($x = 1$, $0 < y < 1$): $\Phi(x,y) = q_\infty$.
- Along the top freestream boundary ($y = 1$, $0 < x < 1$): $\Phi(x,y) = x$.
- Along the bottom boundary, we impose for symmetry the no-penetration condition $\frac{\partial\Phi}{\partial y} = 0$ (y=0, $0 < x \leqslant \frac{1}{3}$, and $\frac{2}{3} \leqslant x < 1$) and a transpiration boundary condition $\frac{\partial\Phi}{\partial y} = \frac{\partial\Phi}{\partial x} f'(3x-1)$ ($y = 0$, $\frac{1}{3} < x < \frac{2}{3}$) that forces the flow to sense the tapered airfoil geometry, while allowing the computation to occur on a Cartesian mesh, for simplicity. The function $f(z)$ used for the NACA 0012 airfoil geometry is given by

$$\begin{aligned} f(z) = {} & 0.17814(\sqrt{z} - z) + 0.10128(z(1-z)) - 0.10968z^2(1-z) \\ & + 0.06090z^3(1-z), \quad z \in (0,1), \end{aligned}$$

and it is scaled into the interval $(\frac{1}{3}, \frac{2}{3})$ through $z = 3x - 1$.

The square domain Ω is partitioned into uniform rectangular cells. A finite difference scheme [21] is used to discretize the two-dimensional full potential flow problem, leading to large algebraic systems with respect to the unknown potential at the grid points:

$$\begin{aligned} & \frac{\rho_{i+\frac{1}{2},j}(\Phi_{i+1,j} - \Phi_{ij}) - \rho_{i-\frac{1}{2},j}(\Phi_{ij} - \Phi_{i-1,j})}{\Delta x^2} \\ & \qquad + \frac{\rho_{i,j+\frac{1}{2}}(\Phi_{i,j+1} - \Phi_{ij}) - \rho_{i,j-\frac{1}{2}}(\Phi_{ij} - \Phi_{i,j-1})}{\Delta y^2} = 0, \end{aligned} \tag{3.10}$$

where $i = 1, 2, \ldots, N_x$ and $j = 1, 2 \ldots, N_y$. Following [4, 18, 42], we introduce a first-order density upwinding scheme in order to capture the shock in the solution for transonic cases, and it plays an essential role in the success of the inexact Newton method, in terms of the convergence and finding the correct location and strength of the shock. For example, the density coefficient $\rho_{i+\frac{1}{2},j}$ in (3.10) is replaced by

$$\hat{\rho}_{i+\frac{1}{2},j} = \rho_{i+\frac{1}{2},j} - \mu_{ij}(\rho_{i+\frac{1}{2},j} - \rho_{i-\frac{1}{2},j}) \triangleq \rho^-_{i+\frac{1}{2},j}, \quad V^x_{i+\frac{1}{2},j} > 0, \tag{3.11}$$

$$\hat{\rho}_{i+\frac{1}{2},j} = \rho_{i+\frac{1}{2},j} - \mu_{i+1,j}(\rho_{i+\frac{1}{2},j} - \rho_{i+\frac{3}{2},j}) \triangleq \rho^+_{i+\frac{1}{2},j}, \quad V^x_{i+\frac{1}{2},j} < 0, \tag{3.12}$$

where $V^x_{i+\frac{1}{2},j}$ is the velocity component in the x-direction and μ_{ij} is the switching function to control the convergence of Newton's method. Combining (3.11) and (3.12), we write a unified formula to compute the density,

$$\hat{\rho}_{i+\frac{1}{2},j} = \frac{1}{2}\left(\rho^-_{i+\frac{1}{2},j} + \rho^+_{i+\frac{1}{2},j}\right) + \frac{1}{2}\mathrm{sgn}(V^x_{i+\frac{1}{2},j})\left(\rho^-_{i+\frac{1}{2},j} - \rho^+_{i+\frac{1}{2},j}\right), \tag{3.13}$$

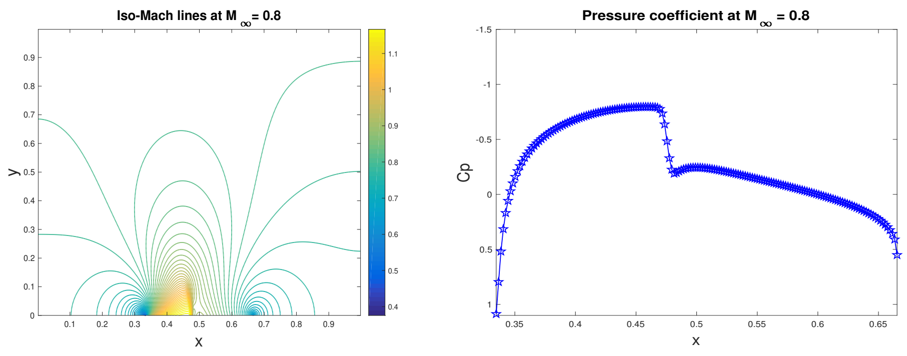

FIG. 8. *Mach number countours (left) and the pressure coefficient C_p curve (right) obtained by the final solution at $M_\infty = 0.8$ on the uniform mesh* 512×512.

where $\mathrm{sgn}(x)$ is the sign function. In a practical implementation, the sign function is replaced by the hyperbolic tangent $\tanh(kx)$ (k a positive integer) to render the upwinding differentiable. Here we choose the second-level switching function, i.e., $\mu_{ij}^{(2)} = \max\{0, 1 - \hat{M}_c^2/M_{s,t}^2\}$, where $i-2 \le s \le i+2$ and $j-2 \le t \le j+2$. In this paper, we set $\hat{M}_c^2 = 0.95$, and $M_{s,t}$ is the local Mach number at (x_s, y_t) given by

$$M_{s,t} = \frac{M_\infty q_{s,t}}{\sqrt{q_\infty^2 + \frac{\gamma-1}{2} M_\infty^2 (q_\infty^2 - q_{s,t}^2)}}, \tag{3.14}$$

where $q_{s,t} = \|\nabla\Phi\|_2$ is the local flow speed at (x_s, y_t). More details can be found in [4, 15, 20].

Both global systems and subspace nonlinear problems are solved by INB techniques. Global Jacobian systems are solved by GMRES(30) with right overlapping restricted additive Schwarz preconditioners, where each individual block is solved by the direct LU decomposition and the overlap is set to 2. Our tests set the initial guess to be a simple interpolation of the farfield boundary condition, i.e., $\Phi(x, y) = x$, for both INB and NEPIN. We set the tolerance parameters as $\epsilon_{global-nonlinear-rtol} = 10^{-10}$, $\epsilon_{global-linear-rtol} = 10^{-3}$, $\epsilon_{sub-linear-rtol} = 10^{-2}$. Figure 8 shows the Mach contours and the pressure coefficient C_p at the final solution at $M_\infty = 0.8$ on the uniform mesh 512×512, where C_p is calculated using

$$C_p = \frac{2}{\gamma M_\infty^2}\left(\left(1 + \frac{\gamma-1}{2} M_\infty^2 (1 - q^2)\right)^{\frac{\gamma}{\gamma-1}} - 1\right).$$

In the NEPIN algorithm, we initialize the partition as $S_b^{(0)} = \emptyset$ and $S_g^{(0)} = S$. For the following iterations, we choose the cutoff Mach number M_c, and the bad components are those whose local Mach numbers satisfy $M_{s,t} > M_c$, with the same balance of considerations of cost per iteration versus robustness of the overall iteration as in the duct flow example of the previous subsection. Table 3 shows a comparison of the cutoff Mach number M_c using NEPIN for various mesh sizes. As M_c varies from 0.88 to 0.90, there is a substantial increase in total execution time and the number of subspace Newton iterations in most cases. Based on Table 3, $M_c = 0.82$ appears to be a suitable balance in terms of the overall execution time. Figure 9 illustrates how the bad region evolves for $M_c = 0.82$ using NEPIN on the uniform mesh 512×512.

TABLE 3

A comparison of the cutoff Mach number M_c using NEPIN on the 256×256 and 512×512 meshes. We set $\epsilon_{global-nonlinear-rtol} = 10^{-10}$, $\epsilon_{global-linear-rtol} = 10^{-3}$, and $\epsilon_{sub-linear-rtol} = 10^{-2}$ as the stopping conditions, respectively.

Mesh size	NEPIN							
	$M_c =$	0.80	0.82	0.83	0.84	0.85	0.88	0.90
	$\epsilon_{sub-nonlinear-rtol} = 10^{-1}$							
256×256	Global Newton steps	8	7	7	7	7	7	9
	GMRES per global Newton	23.1	22.3	22.7	23.6	21.6	22.6	23.4
	Subspace Newton steps	14	13	13	14	12	13	15
	Total execution time (s)	3.92	3.41	3.40	3.49	3.20	3.26	3.91
512×512	Global Newton steps	9	10	10	10	11	10	14
	GMRES per global Newton	38.7	40.2	40.2	40.2	39.7	35.6	36.6
	Subspace Newton steps	28	28	25	25	26	44	69
	Total execution time (s)	27.80	27.54	25.77	25.51	26.90	34.25	49.65
	$\epsilon_{sub-nonlinear-rtol} = 10^{-2}$							
256×256	Global Newton steps	7	7	7	7	7	7	9
	GMRES per global Newton	22.9	22.9	22.1	23.7	23.3	23.7	24
	Subspace Newton steps	17	15	15	15	15	16	17
	Total execution time (s)	4.05	3.65	3.61	3.61	3.57	3.60	4.12
512×512	Global Newton steps	9	9	10	11	12	14	11
	GMRES per global Newton	39.7	40	41.2	41	41.9	40.3	38.4
	Subspace Newton steps	37	35	39	36	37	35	55
	Total execution time (s)	32.97	30.02	33.31	32.64	34.20	34.19	39.65
	$\epsilon_{sub-nonlinear-rtol} = 10^{-3}$							
256×256	Global Newton steps	6	6	6	6	7	7	8
	GMRES per global Newton	24.3	23.7	22.3	23.3	23.4	24	23.1
	Subspace Newton steps	19	18	18	18	17	17	22
	Total execution time (s)	4.09	3.79	3.72	3.69	3.81	3.72	4.37
512×512	Global Newton steps	9	9	10	11	12	14	11
	GMRES per global Newton	40.9	40.7	41.2	41.7	42	41.6	36.6
	Subspace Newton steps	43	41	41	42	42	43	62
	Total execution time (s)	37.01	33.40	34.31	35.90	36.74	38.27	42.80

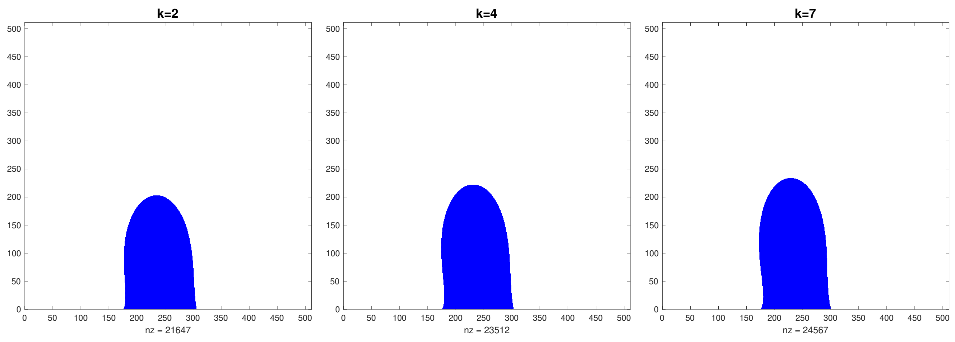

FIG. 9. *For $M_\infty = 0.8$ and $M_c = 0.82$, the figure shows the evolution of the distribution of the bad components ($M_{s,t} > M_c$) using NEPIN on a uniform 512×512 mesh, corresponding to the second, the fourth, and the seventh global Newton iterations. "nz" represents the number of bad components, less than 10% of the total.*

It is observed from numerical experiments that the number of bad components does not change after the seventh nonlinear iteration in this case. Moreover, we note that the numbers of elements in $S_b^{(k)}$ account for less than 10% of the total number of unknowns, and therefore the subspace solvers require relatively low computational overhead.

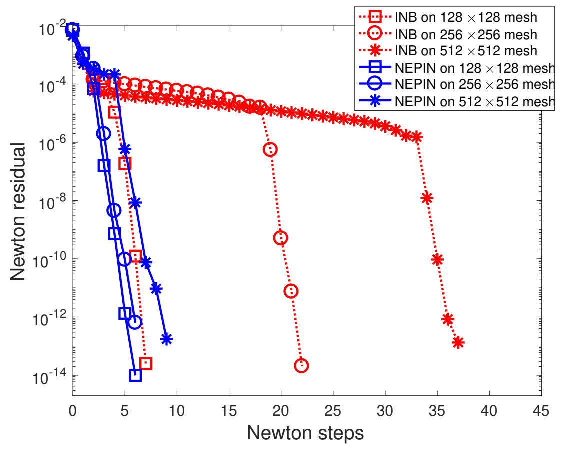

FIG. 10. *For $M_\infty = 0.8$ and $M_c = 0.82$, the figure shows the history of the Newton residual using INB and NEPIN on 128×128, 256×256, and 512×512 meshes. For NEPIN, we set the tolerance $\epsilon_{sub-nonlinear-rtol} = 10^{-2}$ for subspace nonlinear problems.*

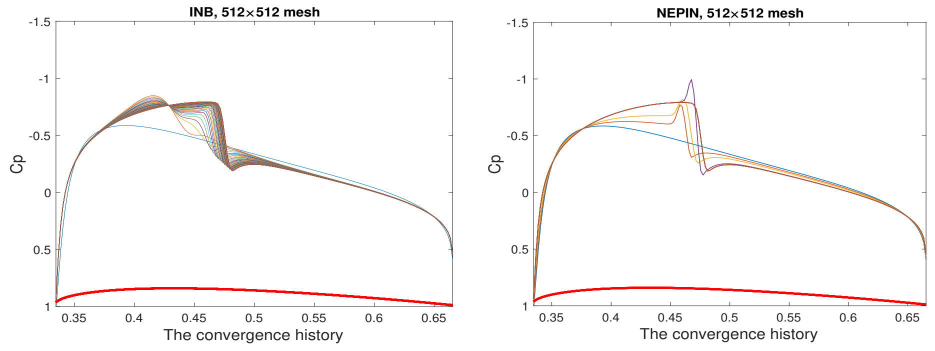

FIG. 11. *For $M_\infty = 0.8$ and $M_c = 0.82$, the figure shows the convergence history of the pressure coefficient C_p curve on the uniform mesh 512×512, as well as a vertically exaggerated NACA 0012 curve (red) at the bottom.*

For $M_\infty = 0.8$ and $M_c = 0.82$, we run the test using INB and NEPIN for uniform meshes of sizes 128×128, 256×256, and 512×512. Here we set the tolerance $\epsilon_{sub-nonlinear-rtol} = 10^{-2}$ for subspace nonlinear problem in the NEPIN algorithm. Figure 10 shows the history of the Newton residual $\|F(\Phi^{(k)})\|$ for two different methods on problems of different resolutions. In contrast with the long plateau period required by global INB, NEPIN converges within at most 9 iterations and is very effective in reducing the total number of global Newton iterations. Figure 11 shows the convergence history of the pressure coefficient C_p curve using INB and NEPIN on a 512×512 mesh, respectively. We note that it takes many iterations for INB to establish the shape of the shock and move it to the exact location. INB suffers from a period of slow evolution before the linear convergence is interrupted in Figure 10. According to the right plot of Figure 11, however, it takes only 2 iterations for NEPIN to set up the neighborhood of the shock with an overshoot, and then it forms the approximately exact shock without any overshoots after the fifth iteration, which corresponds to the superlinear convergence rate in Figure 10. We remark that the red INB convergence histories of Figure 10 motivate another common strategy

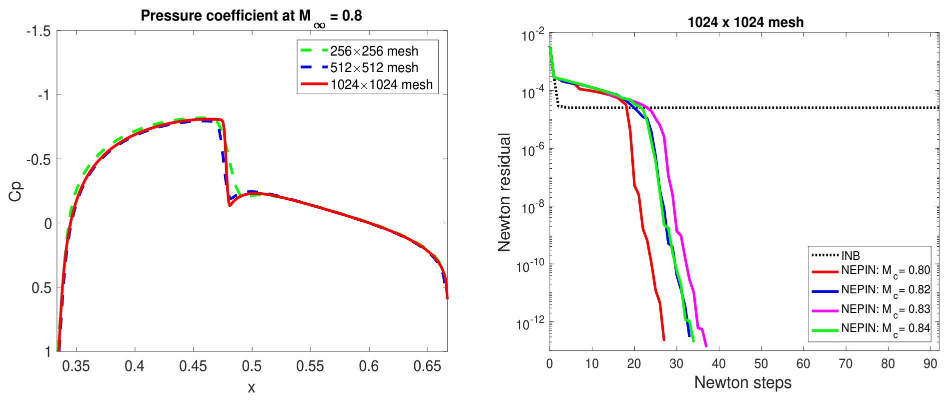

FIG. 12. *For $M_\infty = 0.8$, the left figure shows the pressure coefficient C_p curve for different mesh sizes; the right shows the history of the Newton residual using INB and NEPIN with different cutoff Mach numbers M_c on the 1024×1024 mesh, and the INB algorithm is terminated because of the failure of the backtracking line search after 92 iterations.* $\epsilon_{sub-nonlinear-rtol} = 10^{-3}$.

for problems with such mesh dependences: Grid sequencing, in which a solution obtained on a coarse mesh is used to initialize the iterations on the next finer mesh until asymptotically only a "root-polishing" iteration or two is required. The same motivation is drawn from the blue NEPIN curves, though the payoff is less drastic.

As observed from the left panel of Figure 12, the shock is less smeared as the mesh is refined up to 1024×1024. To obtain a better resolution of the shock, we solve the problem on the 1024×1024 mesh and the nonlinear iteration of the subspace nonlinear problem is stopped if it has a 10^{-3} relative reduction of the initial residual or the maximum of 20 nonlinear iterations is reached. The right panel of Figure 12 shows the the history of the Newton residual using INB and NEPIN with $M_c = 0.80, 0.82, 0.83, 0.84$, respectively. The INB algorithm is terminated because of the failure of the backtracking line search after 92 iterations, but the NEPIN algorithm converges within 40 iterations. The failure of subproblems happens on the 1024×1024 mesh because the line search fails or the maximum number of nonlinear iterations is reached, which results in an extremely inexact Newton direction. Interestingly, NEPIN suffers from the stagnation of nonlinear residual norms and fails to converge within 100 iterations when the cutoff Mach number $M_c = 0.81$ on the 1024×1024 mesh, illustrating the fickleness of the choice of the subproblems requiring elimination.

We also show some parallel scaling results on a Cray X40 with dual socket compute nodes based on 16-core Intel Haswell processors running at 2.3 GHz, each node having 128 GB of DDR4 memory running at 2.3 GHz. We vary the number of processors and present the numerical results using INB and NEPIN with different tolerances for the subspace nonlinear problems. For the 512×512 mesh, modest strong scalability for INB and NEPIN within the PETSc framework are displayed in Table 4, using 4, 16, 64, 256, up to 1024 processors. Varying the number of subdomains results in changes of the inexact Newton direction on a fixed mesh. As the number of processors increases, the number of global Newton iterations does not change much, but the number of global linear iterations per global Newton increases, which is attributed to the lack of intersubdomain communication for a one-level domain decomposition. For the fixed subdomain partition, Table 4 tracks the number of global Newton iterations using NEPIN, which is not sensitive as we change $\epsilon_{sub-nonlinear-rtol}$. It seems that the loose tolerance, e.g., 10^{-1}, is sufficient for NEPIN on a 512×512 mesh, and it can

Table 4

Different subdomain partitions with the same fine mesh 512×512, $M_\infty = 0.8$, and $M_c = 0.82$. We set $\epsilon_{global-nonlinear-rtol} = 10^{-10}$ and $\epsilon_{global-linear-rtol} = 10^{-3}$ as the stopping conditions for the global Newton and GMRES iterations, respectively. N_p–indicates the number of processors, which is the same as the number of subdomains.

N_p		INB	NEPIN 10^{-1}	$\epsilon_{sub-nonlinear-rtol} =$ 10^{-2}	10^{-3}
$4 = 2 \times 2$	No. of global Newton steps	37	10	9	9
	No. of GMRES per global Newton	19.9	17.9	17.6	18.0
	No. of subspace Newton steps.	-	28	36	41
	Execution time (s) of subspace solvers	-	118.34	151.22	172.39
	Total execution time (s)	372.36	246.60	266.78	288.35
$16 = 4 \times 4$	No. of global Newton steps	37	10	9	9
	No. of GMRES per global Newton	27.5	24.1	25.1	25.4
	No. of subspace Newton steps.	-	26	35	40
	Execution time (s) of subspace solvers	-	41.30	55.15	63.23
	Total execution time (s)	100.15	76.68	87.31	95.48
$64 = 8 \times 8$	No. of global Newton steps	37	10	9	9
	No. of GMRES per global Newton	44.9	40.2	40.0	40.7
	No. of subspace Newton steps.	-	28	35	41
	Execution time (s) of subspace solvers	-	15.15	18.85	22.18
	Total execution time (s)	35.91	27.49	29.97	33.35
$256 = 16 \times 16$	No. of global Newton steps	38	10	10	10
	No. of GMRES per global Newton	71.6	57.4	58.5	58.4
	No. of subspace Newton steps.	-	25	37	41
	Execution time (s) of subspace solvers	-	4.15	6.12	6.71
	Total execution time (s)	12.95	8.08	10.17	10.67
$1024 = 32 \times 32$	No. of global Newton steps	37	10	10	10
	No. of GMRES per global Newton	96.0	75.2	84.7	87.5
	No. of subspace Newton steps.	-	27	38	41
	Execution time (s) of subspace solvers	-	1.67	2.26	2.47
	Total execution time (s)	4.71	3.12	3.79	4.03

save at least 23% time compared with INB. In addition, we note that the execution time of subspace solvers accounts for half of the total execution time. For subspace nonlinear problems, however, we actually do not exploit any additional techniques (e.g., mesh sequencing) to accelerate convergence of Newton iterations. For the current implementation, we do not improve the performance of subspace solvers in terms of load balancing. As a result, computing costs in subdomains near the airfoil are much higher than in other subdomains. Therefore, NEPIN is expected to be more efficient when dynamic runtime systems remapping tasks to underutilized processors are employed, as is routinely done with tile algorithms in direct factorization methods of linear algebra, for example. This is a general focus of future work for most nonlinear preconditioning methods.

Figure 13 shows strong scaling behavior using INB and NEPIN on the 512×512 and 1024×1024 meshes, respectively. In terms of the execution time, both INB and NEPIN scale well for up to 1024 processors on the 512×512 mesh. For the 1024×1024 mesh, we do not show any results for INB because of the failure of the backtracking line search on this mesh in Figure 13, and it only shows the performance of strong scalability for NEPIN up to 4096 processors, corresponding to $M_c = 0.80$ and $M_c = 0.82$, respectively. The failure of subproblems causes drastic changes in the number of global Newton iterations as the number of processors increases, but this happens only in the case of the 1024×1024 mesh. Therefore, it is not surprising that the scaling behavior degrades on the 1024×1024 mesh compared with the coarser mesh.

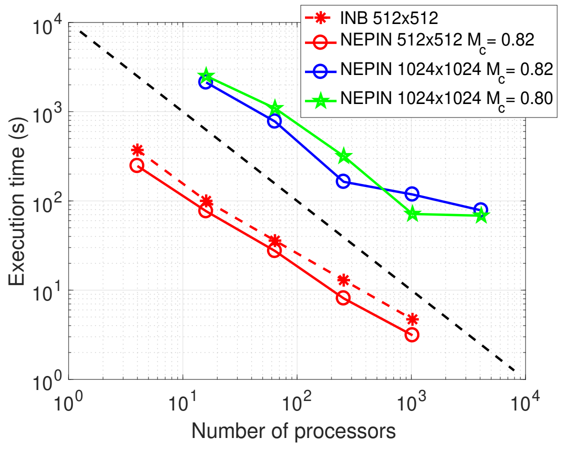

FIG. 13. *Strong scaling for the transonic full potential flow problem on the* 512×512 *and* 1024×1024 *meshes. Execution time for INB is not shown since it fails to converge on the* 1024×1024 *mesh.* $\epsilon_{sub-nonlinear-rtol} = 10^{-1}$.

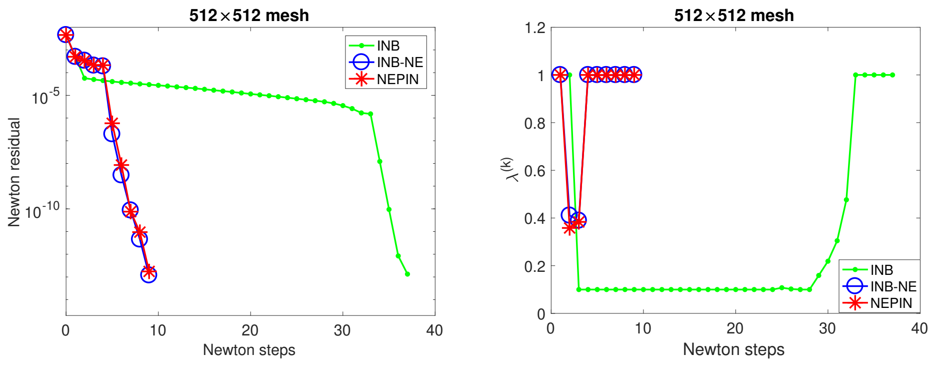

FIG. 14. *For* $M_\infty = 0.8$ *and* $M_c = 0.82$*, the figure shows the history of the Newton residual and the damping parameter* $\lambda^{(k)}$ *using INB, INB-NE, and NEPIN on the* 512×512 *mesh. For INB-NE and NEPIN, we set the tolerance* $\epsilon_{sub-nonlinear-rtol} = 10^{-2}$ *for subspace nonlinear problems.*

For $M_\infty = 0.8$ and $M_c = 0.82$, Figure 14 compares the history of Newton residuals and the damping parameter $\lambda^{(k)}$ on the uniform 512×512 mesh using INB, INB-NE, and NEPIN. For the left plot, INB suffers from the stagnation because of the localized strong nonlinearities. In contrast, both INB-NE and NEPIN are effective in reducing the number of Newton iterations. From the right plot of Figure 14, the damping parameter $\lambda^{(k)}$ is forced down to the lower bound 0.1 (the default value in PETSc) in INB, corresponding to the long plateau in the norm of the nonlinear residual. However, INB-NE and NEPIN perform the full Newton step on most iterations, so fast convergence is attached. We mention that reducing the number of global Newton steps reduces global synchronization frequency and overall communication volume. This more than compensates, in general, for a small increase in local computational work to perform the elimination of the bad components, and it is a trade-off that algorithm designers increasingly want to make on the path to exascale.

Finally, we consider a different elimination approach based on the residual suggested in [14, 33]. Let $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ represent the set of mesh points, and

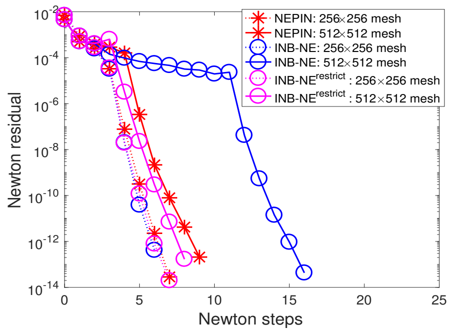

FIG. 15. *For $M_\infty = 0.8$, the figure shows the history of the Newton residual using NEPIN, INB-NE, and INB-NE^{restrict} on 256×256 and 512×512 meshes. For three methods, we set the tolerance $\epsilon_{sub-nonlinear-rtol} = 10^{-2}$ for subspace nonlinear problems, $\rho_{res} = 10^{-3}$, $r = 0.008$, and $\varepsilon = 0.002$. The bad components for the three methods are identified with residual thresholding as shown in (3.15).*

$F_i(x)$ be the nonlinear residual component corresponding to the point $\mathbf{p}_i$. The index set corresponding to the bad components are defined as

$$S_b = \{i \in S \mid \exists\, j,\ \mathrm{dist}(\mathbf{p}_i, \mathbf{p}_j) \leqslant r \text{ and } |F_j(x)| > \rho_{res}\|F(x)\|_\infty\}, \tag{3.15}$$

where $\mathrm{dist}(\mathbf{p}_i, \mathbf{p}_j)$ is the Euclidean distance between $\mathbf{p}_i$ and $\mathbf{p}_j$. The restricted version of INB-NE, INB-NE$^{\text{restrict}}$, keeps the corrected update only in the interior of the bad region in order to avoid sharp jumps in the residual function, and the corresponding restricted index set of bad components is defined as

$$S_b^\varepsilon = \{i \in S \mid \exists\, j,\ \mathrm{dist}(\mathbf{p}_i, \mathbf{p}_j) \leqslant r - \varepsilon \text{ and } |F_j(x)| > \rho_{res}\|F(x)\|_\infty\}. \tag{3.16}$$

In numerical tests, the parameters are set as $\rho_{res} = 10^{-3}$, $r = 0.008$, and $\varepsilon = 0.002$, and we initialize the partition as $S_b^{(0)} = \emptyset$ and $S_g^{(0)} = S$. We restrict the size of subproblems to be less than 15% of the total degrees of freedom, otherwise INB is implemented for the current iteration. For $M_\infty = 0.8$, Figure 15 shows the history of the Newton residual using NEPIN, INB-NE, and INB-NE$^{\text{restrict}}$, respectively. All three algorithms converge within at most seven iterations on the 256×256 mesh. In contrast, it is observed that INB-NE requires nearly twice as much as the number of nonlinear iterations using NEPIN or INB-NE$^{\text{restrict}}$ on the 512×512 mesh. Figure 16

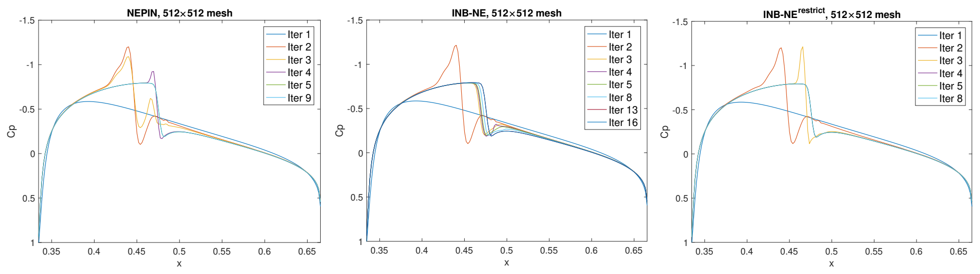

FIG. 16. *For $M_\infty = 0.8$, the figure shows the history of the pressure coefficient C_p using NEPIN, INB-NE, and INB-NE^{restrict}, respectively. We set the parameters as $\rho_{res} = 10^{-3}$, $r = 0.008$, and $\varepsilon = 0.002$.*

shows the corresponding history of the pressure coefficient C_p using NEPIN, INB-NE, and INB-NE$^{\text{restrict}}$ on the 512×512 mesh. From the plots, it takes more than 10 iterations for INB-NE to establish the shape of the shock and move it to the exact location, which corresponds to the slow evolution before the superlinear convergence rate is achieved in Figure 15.

4. Conclusions and future directions.

A new algorithm for problems with unbalanced nonlinearities, namely, a left-preconditioned NE method, NEPIN, a companion to INB-NE, is presented. Numerical results illustrate that NEPIN is effective in improving the performance of global Newton iterations. NEPIN requires additional computational cost for the solution of subspace nonlinear problems in inner iterations. Furthermore, identification of the degrees of freedom to be eliminated is problem dependent and the algorithm can be sensitive to this heuristic. The partition into bad and good degrees of freedom can be learned over a production history of a given application. To improve the trade-off between the extra overhead and the fast robust convergence, the adaptive nonlinear preconditioning framework of [31] may be useful in turning NEPIN on and iterations on and off in the course of the outer Newton iterations, in particular, in turning it off as the domain of convergence of the global Newton iteration is detected. NE in a subspace is a sequential step that produces a work imbalance in bulk-synchronous implementations of implicit PDE solvers. In commonly encountered examples, its size is small compared to that of the global Newton steps that it saves. However, dynamic runtime systems using directed acyclic graphs to control data dependences can be employed in future implementations with more highly heterogeneous imbalances of load.

Acknowledgments. The authors are grateful for the use of a Cray XC40 ("Shaheen II") operated by the Supercomputing Laboratory of the King Abdullah University of Science and Technology.

REFERENCES

[1] H.-B. An, *On convergence of the additive Schwarz preconditioned inexact Newton method*, SIAM J. Numer. Anal., 43 (2005), pp. 1850–1871.

[2] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, D. A., V. Eijkhout, W. D. Gropp, K. D., D. Kaushik, M. G. Knepley, M. D. A., L. C. McInnes, M. R. T., M. T., K. Rupp, S. P., B. F. Smith, S. Zampini, H. Zhang, and H. Zhang, *PETSc*, http://www.mcs.anl.gov/petsc (2021).

[3] P. R. Brune, M. G. Knepley, B. F. Smith, and X. Tu, *Composing scalable nonlinear algebraic solvers*, SIAM Rev., 57 (2015), pp. 535–565.

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

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

[6] X.-C. Cai, D. E. Keyes, and L. Marcinkowski, *Non-linear additive Schwarz preconditioners and application in computational fluid dynamics*, Int. J. Numer. Methods Fluids, 40 (2002), pp. 1463–1470.

[7] X.-C. Cai, D. E. Keyes, and D. P. Young, *A nonlinear additive Schwarz preconditioned inexact Newton method for shocked duct flow*, in Domain Decomposition Methods in Science and Engineering: Proceedings of the 13th International Conference on Domain Decomposition Methods, International Center for Numerical Methods in Engineering, Barcelona, 2002.

[8] X.-C. Cai and M. Sarkis, *A restricted additive Schwarz preconditioner for general sparse linear systems*, SIAM J. Sci. Comput., 21 (1999), pp. 792–797.

[9] F. Chaouqui, M. J. Gander, P. Kumbhar, and T. Vanzan, *Linear and nonlinear substructured Restricted Additive Schwarz iterations and preconditioning*, Numer. Algorithms, to appear.

[10] P. Cresta, O. Allix, C. Rey, and S. Guinard, *Nonlinear localization strategies for domain decomposition methods: Application to post-buckling analyses*, Comput. Methods Appl. Mech. Eng., 196 (2007), pp. 1436–1446.

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

[12] V. Dolean, M. J. Gander, W. Kheriji, F. Kwok, and R. Masson, *Nonlinear preconditioning: How to use a nonlinear Schwarz method to precondition Newton's method*, SIAM J. Sci. Comput., 38 (2016), pp. 3357–3380.

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

[14] S. Gong and X.-C. Cai, *A nonlinear elimination preconditioned inexact Newton method for heterogeneous hyperelasticity*, SIAM J. Sci. Comput., 41 (2019), pp. S390–S408.

[15] W. G. Habashi and M. M. Hafez, *Finite element solutions of transonic flow problems*, AIAA J., 20 (1982), pp. 1368–1376.

[16] A. Heinlein and M. Lanser, *Additive and hybrid nonlinear two-level Schwarz methods and energy minimizing coarse spaces for unstructured grids*, SIAM J. Sci. Comput., 42 (2020), pp. A2461–A2488.

[17] V. E. Henson and U. M. Yang, *BoomerAMG: A parallel algebraic multigrid solver and preconditioner*, Appl. Numer. Math., 41 (2002), pp. 155–177.

[18] T. L. Holst, *Transonic flow computations using nonlinear potential methods*, Progr. Aerosp. Sci., 36 (2000), pp. 1–61.

[19] F.-N. Hwang, H.-L. Lin, and X.-C. Cai, *Two-level nonlinear elimination based preconditioners for inexact Newton methods with application in shocked duct flow calculation*, Electron. Trans. Numer. Anal., 37 (2010), pp. 239–251.

[20] F.-N. Hwang, Y.-C. Su, and X.-C. Cai, *A parallel adaptive nonlinear elimination preconditioned inexact Newton method for transonic full potential equation*, Comput. & Fluids, 110 (2015), pp. 96–107.

[21] A. Jameson, *Transonic potential flow calculations using conservation form*, in Proceedings of the 2nd AIAA Computational Fluid Dynamics Conference, AIAA, New York, 1975, pp. 148–155.

[22] D. E. Keyes et al., *Multiphysics simulations: Challenges and opportunities*, Int. J. High Perform. Comput. Appl., 27 (2013), pp. 4–83.

[23] A. Klawonn, M. Lanser, and O. Rheinbach, *Nonlinear FETI-DP and BDDC methods*, SIAM J. Sci. Comput., 36 (2014), pp. A737–A765.

[24] A. Klawonn, M. Lanser, and O. Rheinbach, *Toward extremely scalable nonlinear domain decomposition methods for elliptic partial differential equations*, SIAM J. Sci. Comput., 37 (2015), pp. C667–C696.

[25] A. Klawonn, M. Lanser, O. Rheinbach, and M. Uran, *Nonlinear FETI-DP and BDDC methods: A unified framework and parallel results*, SIAM J. Sci. Comput., 39 (2017), pp. C417–C451.

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

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

[28] L. Liu and D. E. Keyes, *Field-split preconditioned inexact Newton algorithms*, SIAM J. Sci. Comput., 37 (2015), pp. 1388–1409.

[29] L. Liu and D. E. Keyes, *Convergence analysis for the multiplicative Schwarz preconditioned inexact Newton algorithm*, SIAM J. Numer. Anal., 54 (2016), pp. 3145–3166.

[30] L. Liu and D. E. Keyes, *Approximate error bounds on solutions of nonlinearly preconditioned PDEs*, SIAM J. Sci. Comput., 43 (2021), pp. A2526–A2554.

[31] L. Liu, D. E. Keyes, and R. Krause, *A note on adaptive nonlinear preconditioning techniques*, SIAM J. Sci. Comput., 40 (2018), pp. A1171–A1186.

[32] L. Luo, X.-C. Cai, Z. Yan, L. Xu, and D. E. Keyes, *A multilayer nonlinear elimination preconditioned inexact Newton method for steady-state incompressible flow problems in three dimensions*, SIAM J. Sci. Comput., 42 (2020), pp. B1404–B1428.

[33] L. Luo, W.-S. Shiu, R.-L. Chen, and X.-C. Cai, *A nonlinear elimination preconditioned inexact Newton method for blood flow problems in human artery with stenosis*, J. Comput. Phys., 399 (2019), 108926.

[34] L. Marcinkowski and X.-C. Cai, *Parallel Performance of Some Two-level ASPIN Algorithms*, in Domain Decomposition Methods in Science and Engineering, Springer, Berlin, 2005, pp. 639–646.

[35] J. Pebrel, C. Rey, and P. Gosselet, *A nonlinear dual-domain decomposition method: Application to structural problems with damage*, Int. J. Multiscale Comput. Eng., 6 (2008), pp. 251–262.

[36] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[37] S. Shitrit, D. Sidilkover, and A. Gelfgat, *An algebraic multigrid solver for transonic flow problems*, J. Comput. Phys., 230 (2011), pp. 1707–1729.

[38] J. O. Skogestad, E. Keilegavlen, and J. M. Nordbotten, *Domain decomposition strategies for nonlinear flow problems in porous media*, J. Comput. Phys., 234 (2013), pp. 439–451.

[39] A. J. Wathen, *Preconditioning*, Acta Numer., 24 (2015), pp. 329–376.

[40] H. Yang, F.-N. Hwang, and X.-C. Cai, *Nonlinear preconditioning techniques for full-space Lagrange–Newton solution of PDE-constrained optimization problems*, SIAM J. Sci. Comput., 38 (2016), pp. A2756–A2778.

[41] H. Yang, C. Yang, and S. Sun, *Active-set reduced-space methods with nonlinear elimination for two-phase flow problems in porous media*, SIAM J. Sci. Comput., 38 (2016), pp. B593–B618.

[42] D. P. Young, R. G. Melvin, M. B. Bieterman, F. T. Johnson, S. S. Samant, and J. E. Bussoletti, *A locally refined rectangular grid finite element method: Application to computational fluid dynamics and computational physics*, J. Comput. Phys., 92 (1991), pp. 1–66.