NONLINEAR PRECONDITIONING TECHNIQUES FOR FULL-SPACE LAGRANGE-NEWTON SOLUTION OF PDE-CONSTRAINED OPTIMIZATION PROBLEMS*

HAIJIAN YANG†, FENG-NAN HWANG‡, AND XIAO-CHUAN CAI§

Abstract. The full-space Lagrange-Newton algorithm is one of the numerical algorithms for solving problems arising from optimization problems constrained by nonlinear partial differential equations. Newton-type methods enjoy fast convergence when the nonlinearity in the system is well-balanced, however, for some problems, such as the control of incompressible flows, even linear convergence is difficult to achieve and a long stagnation period often appears in the iteration history. In this work, we introduce a nonlinearly preconditioned inexact Newton algorithm for the boundary control of incompressible flows. The system has nine field variables, and each field variable plays a different role in the nonlinearity of the system. The nonlinear preconditioner approximately removes some of the field variables, and as a result, the nonlinearity is balanced and inexact Newton converges much faster when compared to the unpreconditioned inexact Newton method or its two-grid version. Some numerical results are presented to demonstrate the robustness and efficiency of the algorithm.

Key words. PDE-constrained optimizations, flow control, nonlinear elimination preconditioner, inexact Newton method, sequential quadratic programming

1. Introduction. Optimization problems constrained by nonlinear partial differential equations (PDEs) represent a large class of problems with important scientific and engineering applications, such as control of fluid flows, optimal design, and inverse problems, etc [7, 8, 17, 21, 23, 31, 40, 41, 49, 50, 51]. The problems are often nonlinearly difficult to solve, and very large in size, therefore require parallel computing. In this paper, we introduce a nonlinearly preconditioned inexact Newton algorithm, which is also highly parallel when used together with domain decomposition method as the linear Jacobian solver. As an application, we consider some boundary control problems of thermally convective fluid flows. In practice, the instability of thermally convective flows is responsible for certain undesirable convective states in some industrial processes such as laser welding, alloy manufacturing, crystal growth, and so on [38, 39, 44]. The flow control can be useful to reduce the instability [9, 17, 36, 50].

The Lagrange-Newton algorithm is one of the algorithms for solving the PDE-constrained optimization problems. The algorithm is based on the full-space sequential quadratic programming technique [37], which converts the constrained optimization problem to an unconstrained problem by introducing some Lagrangian multipliers and a Lagrangian functional. Then, an inexact Newton method is employed to solve a large sparse nonlinear system of equations derived from the first-order optimality condition. From the algorithmic viewpoint, although the inexact Newton method enjoys fast convergence, the full-space method poses some computational challenges. First, at each Newton iteration, a large sparse saddle-point system needs to be solved, which is often highly ill-conditioned [5]. Second, the convergence of the inexact New-

*The first author was supported in part by the National Natural Science Foundation of China: 11571100, 91530103, and 91330111, the second author was supported in part by the Ministry of Science and Technology, Taiwan, MOST-103-2115-M-008-007, and the third author was supported in part by the National Science Foundation, CCF-1216314.

†College of Mathematics and Econometrics, Hunan University, Changsha, Hunan, 410082, P. R. China (haijianyang@gmail.com)

‡Corresponding author. Department of Mathematics, National Central University, Jhongli District, Taoyuan City 32001, Taiwan (hwangf@math.ncu.edu.tw)

§Department of Computer Science, University of Colorado Boulder, Boulder, CO 80309, USA (cai@cs.colorado.edu)

ton method is often problematic. Numerical pieces of evidence [11, 28, 30] suggest that the slow convergence is often determined by a small subset of equations in the system with the highest nonlinearities. The family of continuation methods, such as the grid-sequencing approach [2, 34] or the parameter continuation approach [3, 32], is quite robust for these difficult nonlinear problems but not efficient; see their applications for PDE-constrained optimization problems [7, 48, 49, 51]. Alternatively, we extend some nonlinear preconditioning techniques [29, 30], previously developed for the system of nonlinear equations, to PDE-constrained optimization problems. The fast convergence of the inexact Newton method can be restored when it is employed in conjunction with the nonlinear preconditioning.

Similar to the linear case, a nonlinear preconditioner can be applied on the right or on the left of a nonlinear function. For the left nonlinear preconditioner, the algorithm reformulates the original nonlinear function implicitly into a more balanced function. The additive Schwarz preconditioned inexact Newton algorithm (ASPIN) [11, 28] belongs to this class, where the nonlinearly preconditioned function is defined based on the additive Schwarz framework and the new system is solved by an inexact Newton algorithm. ASPIN has been successfully applied for some nonlinear system of equations with high local nonlinearities, such as high Reynolds number incompressible flows [28], transonic compressible flows [12], nonlinear elasticity problems [22], and multiphase flows in porous media [43, 45]. A parallel implementation of ASPIN is available in the popular software package, PETSc [4], and it can be used as either an iterative method or as a nonlinear preconditioner [10]. More recently, restricted and multiplicative versions of nonlinear Schwarz preconditioner are introduced in [18] and [35], respectively. On the other hand, the right nonlinear preconditioner works differently. The main role of the right nonlinear preconditioner is to balance the nonlinearities of $F(x)$ by changing x. Since x often lives in a space of high dimension, we partition the space into subspaces and then try to identify the subspace in which x behaves badly. Some possible partition strategies are: pointwise partition, subdomain-based partition [30] and field-based partition [35]. In this paper, we focus only on the field based approach since for the flow control problems some fields have to be eliminated to achieve fast convergence, pointwise and subdomain based partitions do not work well.

The rest of this paper is organized as follows. In Section 2, we briefly review the full-space Lagrange-Newton algorithm for solving general PDE-constrained optimization problems. In Section 3, we introduce the nonlinear elimination approach as a right preconditioner and describe in detail the proposed nonlinearly preconditioned Lagrange-Newton algorithm. In Section 4, we discuss a simple example with unbalanced nonlinearity to provide some insight as to how the right nonlinear elimination preconditioner works. In Section 5, we describe a boundary control problem of thermally convective fluids. Section 6 is devoted to numerical experiments and discussions. Some concluding remarks are given in Section 7.

2. Full-space Lagrange-Newton algorithm. We briefly review the framework of full-space Lagrange-Newton algorithm for solving PDE-constrained optimization problems [7, 8, 40, 41], which serves as the basis of the proposed algorithm, in which an inexact Newton-type method is employed as both the subspace and the global nonlinear solver. In this work, we follow the discretize-then-optimize approach, hence after the discretization, we write the resulting finite-dimensional PDE-

constrained optimization problem as follows.

$$\begin{cases} \text{Find the control } u \in U \text{ and the state } s \in S \\ \text{such that } \min_{(s,u)\in S\times U} \ \mathcal{F}(s,u) \\ \text{subject to} \quad C(s,u) = 0, \in Y, \end{cases} \tag{2.1}$$

where S, U, and Y are normed spaces, $\mathcal{F}$: $S \times U \to \mathbb{R}$ is the objective function, C: $S \times U \to Y$ satisfying $C(s,u) = 0$, which is the nonlinear system of equations arising from the discretization of some law of physics modeled by PDEs. To apply a Lagrange-Newton algorithm for solving (2.1), we consider a Lagrangian functional as follows.

$$\mathcal{L}(s,u,\lambda) \equiv \mathcal{F}(s,u) + \lambda^T C(s,u),$$

where λ is the Lagrangian multiplier, and the corresponding KKT system of nonlinear equations is derived by differentiating the Lagrangian functional with respect to the state, the control, and the Lagrangian multiplier, respectively, as following

$$\mathcal{K}(s,u,\lambda) \equiv \begin{pmatrix} \nabla_s \mathcal{L} \\ \nabla_u \mathcal{L} \\ \nabla_\lambda \mathcal{L} \end{pmatrix} = \begin{pmatrix} \nabla_s \mathcal{F} + \nabla_s C^T \lambda \\ \nabla_u \mathcal{F} + \nabla_u C^T \lambda \\ C(s,u) \end{pmatrix}.$$

Then an inexact Newton-type method with backtracking technique (INB) [16] is applied to find the solution of the first-order optimality condition, i.e.,

$$\mathcal{K}(s,u,\lambda) = 0. \tag{2.2}$$

The standard algorithm consists of the following steps.

Algorithm 1 Inexact Newton method with backtracking (INB)

STEP 0. Choose an appropriate initial guess $p^{(0)}$, and set $k = 0$.
Until convergence do
STEP 1. Inexactly solve the Jacobian system

$$\mathcal{K}'(p^{(k)})\Delta p^{(k)} = -\mathcal{K}(p^{(k)}). \tag{2.3}$$

STEP 2. Update $p^{(k+1)} = p^{(k)} + \alpha^{(k)}\Delta p^{(k)}$, where $\alpha^{(k)} \in (0,1]$ is determined by a linesearch along $\Delta p^{(k)}$.
$k = k + 1$
End

Once the initial guess is picked, the most expensive step is Step 1, however, for many problems, Step 0 is the most difficult step which determines the robustness and overall cost of the algorithm. Below we briefly comment on some existing techniques for Step 1, Step 2, and Step 0. (2.3) is a large sparse saddle point type system, which is indefinite and highly ill-conditioned. A Krylov subspace method is often used together with an effective preconditioner. Developing such preconditioners for saddle point problems is an active research topic in numerical linear algebra; see [5] and references therein for details. Some of the well-studied methods include the overlapping Schwarz preconditioners [40, 41] and the block triangular decomposition methods based on Schur complement [7, 8]. The solution accuracy of (2.3) is determined by the forcing term η_k, which is understood as follows.

$$\|\mathcal{K}'(p^{(k)})(M^{(k)})^{-1}M^{(k)}\Delta p^{(k)} + \mathcal{K}(p^{(k)})\|_2 \le \eta_k \|\mathcal{K}(p^{(k)})\|_2, \tag{2.4}$$

where $M^{(k)}$ is a right preconditioner. If η_k is sufficiently small, the method is reduced to the exact Newton method. To enhance the robustness of INB, one can use adaptive forcing terms as suggested by Eisenstat and Walker [19], together with some globalization technique, such as linesearch or trust region. For example, the standard cubic backtracking algorithm [16, 19] can be used to pick $\alpha^{(k)}$ such that

$$\|\mathcal{K}(p^{(k)} + \alpha^{(k)}\Delta p^{(k)})\|_2 \leqslant (1 - \eta\alpha^{(k)})\|\mathcal{K}(p^{(k)})\|_2, \tag{2.5}$$

where η is the parameter associated with backtracking to make sure that the reduction of the merit function, $\|\mathcal{K}\|_2$, is sufficiently large.

Unfortunately, the damping factor $\alpha^{(k)}$ can be extremely small for problems with unbalanced nonlinearity, no matter how accurately or inaccurately one solves the Jacobian problem (2.3). As a result, this may cause the failure of linesearch, i.e., no appropriate scaling factor can be found so that (2.5) is satisfied. Using a high-Reynolds number lid-driven cavity flow as an example, Tuminaro et al. [47] explained that this is likely because the angle between the Newton direction and the steepest descent direction is close to being orthogonal. In such a situation, the Newton direction becomes only a weakly descent direction; thus INB converges slowly or simply stagnates. Linear preconditioner $M^{(k)}$ in (2.4) does not change this angle, but appropriate nonlinear preconditioning may improve the angle.

Step 0 is often the hardest step in INB, a family of the continuation methods provides an alternative, in which using the solution of some "easier" problem as an initial guess for the target problem. As shown in Algorithm 2, a special case of the grid-sequence approaches will be considered for comparison purposes with the proposed nonlinear preconditioned method in Section 6.

Algorithm 2 Two-grid Lagrange-Newton algorithm

Suppose there are two grids covering the computational domain. If we denote Ω_H as the coarse grid, Ω_h as the fine grid, and I_H^h as the coarse to fine interpolation, then the algorithm can be described as follows.

Solve $\mathcal{K}^H(p^H) = 0$ on the coarse grid, by INB using a zero initial guess.
Interpolate the solution from the coarse grid to the fine grid $p_0^h = I_H^h p^H$.
Solve $\mathcal{K}^h(p^h) = 0$ on the fine grid, by INB using p_0^h as the initial guess.

In this work, instead of using the globalization techniques mentioned above, we focus on the development of a nonlinear preconditioning technique to enhance the robustness of INB based on the nonlinear elimination method. As pointed out by [27], the linear preconditioning may speed up the solution algorithm for the Jacobian system, but it does not help improving the quality of the search direction. On the other hand, nonlinear preconditioning may provide a better search direction.

3. Nonlinear elimination preconditioned Newton algorithm for multi-field problems. Consider the nonlinear KKT problem (2.2) in which the variable p consists of several fields, s, u, λ etc. We partition these field variables into two groups and label them as p_g and p_b, i.e., $p = (p_b, p_g)^T$ and we assume the corresponding equations are also partitioned as

$$\mathcal{K}(p) = \mathcal{K}(p_b, p_g) = \begin{bmatrix} \mathcal{K}_b(p_b, p_g) \\ \mathcal{K}_g(p_b, p_g) \end{bmatrix}, \tag{3.1}$$

where p_b and p_g are called the "bad" and "good" components, respectively. For any given good vector, $y_g \in \mathbb{R}^n$, we define its extension to the bad region, $\mathcal{T}_b(y_g) \in \mathbb{R}^m$,

as the solution of the following subspace correction equation,

$$\mathcal{K}_b(\mathcal{T}_b(y_g), y_g) = 0, \tag{3.2}$$

by using INB or other nonlinear iterative methods. Using the subspace correction operator, we introduce a function $\mathcal{G}(x)$ as follows

$$p = \mathcal{G}(y) = \begin{bmatrix} \mathcal{T}_b(y_g) \\ y_g \end{bmatrix}.$$

This function keeps the good component of x, and replaces the bad component by a new component. In practice, this is usually done approximately. In other words, (3.2) is not solved exactly. The reason is not to save the computational time, but to avoid the sharp change from the good component to the bad component. This operation is referred to as nonlinear elimination since y_b is eliminated. Now, consider the right preconditioned system

$$\mathcal{W}(y) \equiv \mathcal{K}(\mathcal{G}(y)) = 0,$$

where

$$p = \mathcal{G}(y)$$

acts as a nonlinear preconditioner, and the Jacobian matrix takes the form

$$\frac{\partial \mathcal{W}}{\partial y} = \frac{\partial \mathcal{K}}{\partial \mathcal{G}} \frac{\partial \mathcal{G}}{\partial y}.$$

Then, INB used for solving $\mathcal{W}(y) = 0$ is given by

$$y^{(k+1)} = y^{(k)} + \alpha \Delta y^{(k)} \tag{3.3}$$

where $\Delta y^{(k)}$ is the solution of the following Jacobian system

$$\frac{\partial \mathcal{W}(y^{(k)})}{\partial y} \Delta y^{(k)} = -\mathcal{W}(y^{(k)}),$$

or

$$\frac{\partial \mathcal{K}(p^{(k)})}{\partial \mathcal{G}} \frac{\partial \mathcal{G}(y^{(k)})}{\partial y} \Delta y = -\mathcal{K}(\mathcal{G}(y^{(k)})), \tag{3.4}$$

Let $\Delta p^{(k)} = \frac{\partial \mathcal{G}(y^{(k)})}{\partial y} \Delta y^{(k)}$ and apply $\mathcal{G}$ to both sides of (3.3) to obtain

$$\begin{aligned} \mathcal{G}(y^{(k+1)}) &= \mathcal{G}(y^{(k)} + \alpha \Delta y) \\ &\approx \mathcal{G}(y^{(k)}) + \alpha \frac{\partial \mathcal{G}(y^{(k)})}{\partial y} \Delta y^{(k)} \end{aligned}$$

Here, the first-order Talyor's expansion around $y^{(k)}$ is applied. Hence,

$$p^{(k+1)} = p^{(k)} + \alpha \Delta p^{(k)}, \tag{3.5}$$

where

$$\frac{\partial \mathcal{K}(p^{(k)})}{\partial \mathcal{G}} \Delta p^{(k)} = -\mathcal{K}(\mathcal{G}(y^{(k)})), \tag{3.6}$$

In general, applying INB directly to solve the system $\mathcal{W}(y) = 0$ in the preconditioned solution space for the y-variable ((3.3) and (3.4)) is not equivalent to solve the system $\mathcal{K}(p) = 0$ for the solution space for the p-variable ((3.5) and (3.6)), since the operator $\mathcal{G}$ and the solution update (3.3) do not commute. Their equivalence can be established by using the linearization of $\mathcal{G}$. Note that in practice, we do not work with the y-variable directly, since it has some potential drawbacks. For example, the operator $\mathcal{G}(y)$ is implicitly defined so that the Jacobian of $\mathcal{G}(y)$ is difficult to obtain. Also designing an effective preconditioner for (3.4) is quite challenging. Hence, in this paper, we focus on INB for $\mathcal{K}(p) = 0$ in conjunction with nonlinear elimination preconditioning and the details of the algorithm are in Algorithm 3.

Algorithm 3 Nonlinear Elimination Preconditioned Inexact Newton algorithm (INB-NE)

Given $p^{(0)}$.
Evaluate $\mathcal{K}(p^{(0)})$ and $\|\mathcal{K}(p^{(0)})\|$, and set $k = 1$.
Until convergence do
  Subspace correction:
    Evaluate $p^{(k)} = \mathcal{G}(p^{(k-1)})$.
  Global update:
    Inexactly solve $\mathcal{K}'(p^{(k)})\Delta p^{(k)} = -\mathcal{K}(p^{(k)})$
    Update $p^{(k+1)} = p^{(k)} + \alpha^{(k)}\Delta p^{(k)}$, where $\alpha^{(k)} \in (0, 1]$.
    Set $k = k + 1$
End

Remarks:

1. The algorithm does not include a step to detect the bad components. The user needs to identify these components. Such mechanism is often problem dependent. Using the knowledge of physics to the problem is one of the possible approaches. For example, for the transonic flow problems, the bad components are the components within or near the region where the shock occurs. In such a situation, the numerical Mach number on the grid points serves as a good indicator for these components [29, 30]. On the other hand, the subdomain wise or pointwise residual norm can also be used for judging whether the nonlinearity of the system is balanced or not. If a small number of components contributes a large percentage of the total nonlinear residual norm, they are classified as the bad components [13, 26]. In the numerical experiment section, we show a slowness analysis as a way to find the bad component.
2. The algorithm needs several control parameters. For example, the outer loop needs stopping conditions, the subspace correction step also needs several stopping conditions, and the Jacobian solve requires some stopping conditions. A careful selection of these control parameters is very important for the fast convergence of the algorithm. It is important to note that the stopping condition for the correction equation should not be too small; in other words, over-correction may slow down the outer iteration.
3. Linear preconditioner is required in the Jacobian solve of the subspace correction, and also the global update step.
4. This algorithm is closely related to the nonlinear elimination algorithm presented in [33]. The main difference is that the NE algorithm of [33] iterates in

the subspace defined by the "good" components, but our algorithm iterates in the whole space, and NE is only a preconditioner.

5. In this paper, we are interested in the splitting of $x = (x_b, x_g)$ based on a field partition, which is different from the domain based partition [13] in several ways. In certain applications, such as flow control problems, the subdomain based partition does not speed up the convergence, because, an entire field needs to be eliminated. An advantage of the field based partition is that the load balancing problem, which is difficult to deal with in the subdomain based partition, disappears since all processors receive an equal number of equations from the bad field(s), and the good field(s).
6. In the subdomain based partition, there is often a jump in the residual function near the subdomain boundary. One has to choose carefully the stopping conditions for the subdomain solver to make sure that the jump is not too large or use a large enough overlap to move the jump away from the interior of the subdomain. In the field based partition, such problem does not seem to be an issue.

4. A simple example with unbalanced nonlinearity. To illustrate how the nonlinear elimination speeds up the Newton iteration, we consider a simple example with unbalanced nonlinearity.

$$r(x_1, x_2) \equiv \begin{bmatrix} r_1(x_1, x_2) \\ r_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} (x_1 - x_2^3 + 1)^m - x_2^m \\ x_1 + 2x_2 - 3 \end{bmatrix} = 0,$$

where $m = 1, 3, 5$. $x^* = (1, 1)^T$ is the exact solution. Four different choices of an initial vector are tested, $x^{(0)} = (0, 0)^T$, $(0, 2)^T$, $(2, 0)^T$, and $(2, 2)^T$. The distances between the initial guesses and the exact solution are all the same. Table 4.1 summarizes the iteration counts for the INB and INB-NE. In INB-NE, x_1 is chosen as the "bad" component, which is eliminated using the first equation, $r_1(x_1, x_2) = 0$, in every outer Newton iteration. Then the preconditioned problem is defined as

$$t(x_1, x_2) \equiv \begin{bmatrix} t_1(x_1, x_2) \\ t_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} (x_1 - \bar{x}_2^3 + 1)^m - \bar{x}_2^m \\ x_2 - \bar{x}_2 \end{bmatrix} = 0, \tag{4.1}$$

provided that $\bar{x}_2$ is given, and the subspace problem in the elimination step is solved by INB. The absolute stopping condition for the outer and subspace Newton is 10^{-7}. Some observations from Table 4.1 are made as follows.

- As m increases, the first equation becomes more nonlinear than the other. If the problem is solved with INB without NE preconditioning, the number of nonlinear iterations increases and is quite sensitive to the choice of the initial guess.
- It is clear that INB-NE improves the convergence of INB, and also reduces its sensitivity to the choice of the initial guess.

To understand why INB-NE works, we plot the iteration path history of INB-NE and that of INB for the purpose of comparison in Fig. 4.1. For INB, basically speaking, the iteration follows the Newton search direction and converges slowly toward the exact solution. The convergence is especially slow in the region with steep gradient or within the valley. On the other hand, although INB-NE follows a similar path as INB (if the subspace correction phase is ignored), the subspace correction step in INB-NE can take the intermediate solution away from the slow region and then bring it back to track via the global update. Unlike INB, which wastes so many steps in the slow region, INB-NE skips several unnecessary steps.

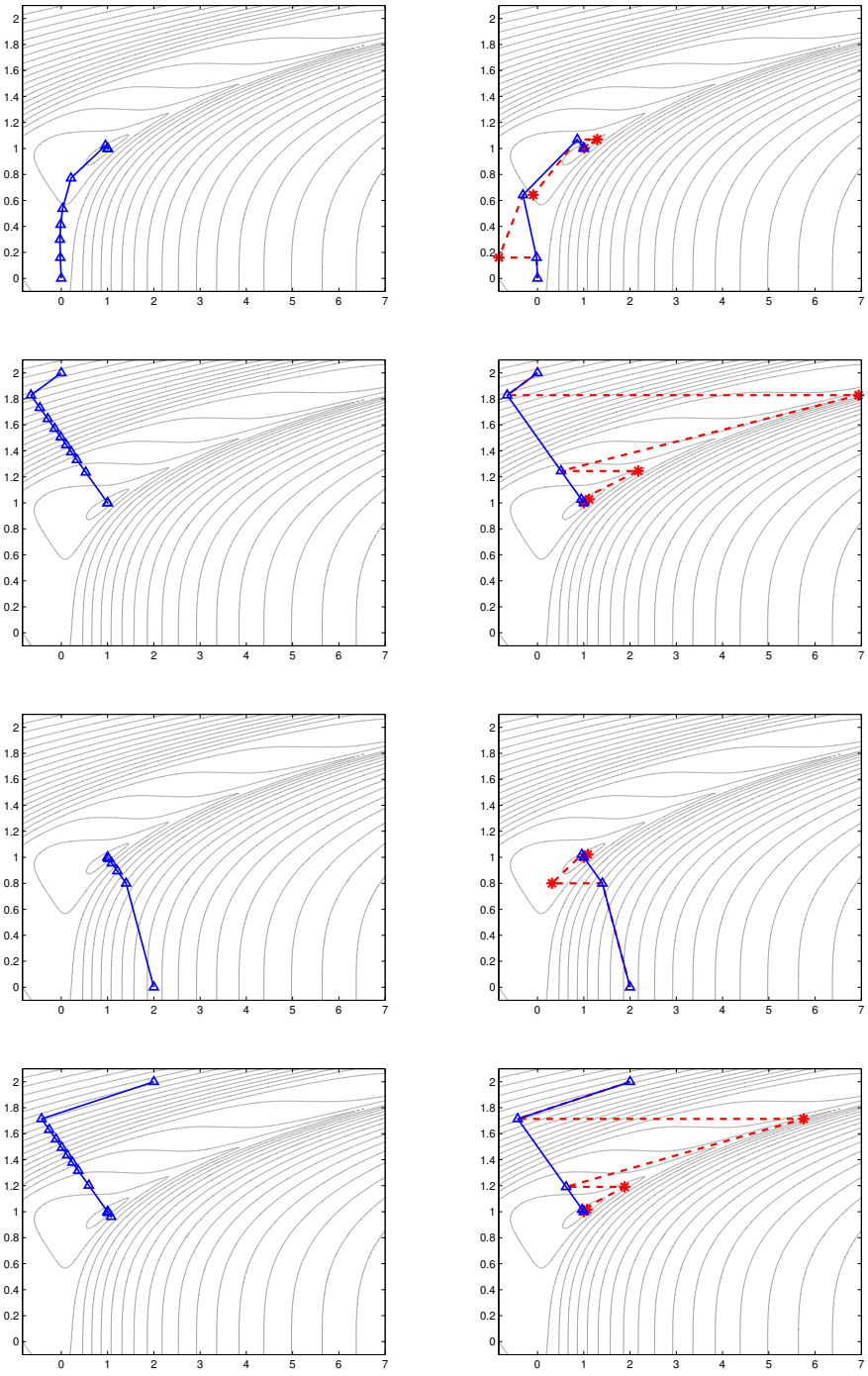

FIG. 4.1. *A contour plot of* $log(0.5\|r(x)\|_2^2 + 1)$ *and a path history comparison of INB (left column) and INB-NE (right column) for the case of* $m = 5$. *Note that the exact solution is located within the valley. For INB, the blue triangle represents the intermediate solution,* p^{k+1} *in Algorithm 1. For INB-NE, the red star represents the solution after the subspace correction phrase is performed in Algorithm 3 and the blue triangle represents the solution after the global updates in Algorithm 3. Each row corresponds a different initial guess as shown in Table 4.1.*

TABLE 4.1
A summary of the number of nonlinear iterations for INB and INB-NE by using four different initial guesses.

	INB			INB-NE		
$x^{(0)}$	m=1	m=3	m=5	m=1	m=3	m=5
$(0,0)^T$	5	8	10	5	6	6
$(0,2)^T$	5	11	12	5	6	6
$(2,0)^T$	5	1	7	5	1	5
$(2,2)^T$	5	12	13	5	5	6

We mention the right nonlinear preconditioning can also be interpreted as a *nonlinear* change-of-basis for the solution space. For example being considered, let $\beta = \{\mathbf{e}_1, \mathbf{e}_2\}$, where $\mathbf{e}_1 = (1,0)^T$ and $\mathbf{e}_2 = (0,1)^T$ be the standard basis for $\mathbb{R}^2$. The subspace correction operator is then derived from (4.1) as

$$\mathcal{G}(x_1, x_2) = (-x_2^3 + 1) \begin{bmatrix} -1 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 1 \\ 1 \end{bmatrix} = (-x_2^3 + 1)\mathbf{e}_1' + x_2\mathbf{e}_2',$$

$\beta' = \{\mathbf{e}_1', \mathbf{e}_2'\}$, where $\mathbf{e}_1' = (-1,0)^T$ and $\mathbf{e}_2' = (1,1)^T$ is a new basis. This relation changes the original Cartesian coordinates into the β'-coordinate system. In the right figure of Fig. 4.2, we replot the contour of $log(0.5\|r(x)\|_2^2 + 1)$ and the outer iterates but now in the transformed space, which is compared with the ones in the original space. What we can see is that the contours of the transformed space become straight lines. This is very different from the linear change of basis, in which the contours are preserved with rotation. As a result, it is much easier for INB to find the solution in the transformed space than in the original space. This is probably why INB with right NE preconditioning works so well. Note that for this simple example, the nonlinear right preconditioned system $r(\mathcal{G}(x_1, x_2)) = 0$ can be formed explicitly and hence the Jacobian can be evaluated exactly. Also, it is easy to check that the iteration actually terminates in a single step in the transformed space.

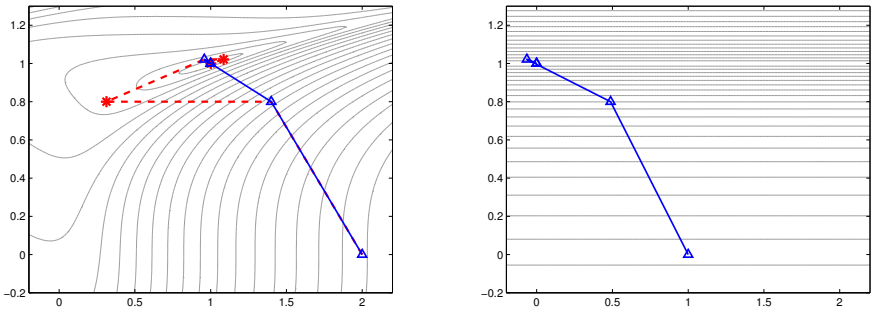

FIG. 4.2. *A comparison of a path history of INB-NE and the contour plot of* $log(0.5\|r(x)\|_2^2+1)$ *on the standard Cartesian coordinates (left) and new transformed coordinates relative to* β' *(right).*

5. Boundary control of thermally convective flows. Flow control problems can be described in many different forms such as flow matching, vorticity minimization, viscous drag minimization, avoiding hot spots, stabilization enhancement, mixing maximization, and so on [1, 6, 20, 23]. The control variables can be defined

on the boundary or inside the flow domain and can be the velocity, external force, temperature, etc. In this study, we focus on the minimization of vorticity in viscous incompressible thermally convective flows by controlling the temperature of the surrounding medium on the wall. Let Ω be a 2D bounded computational domain with the boundary segment, $\Gamma = \Gamma_p \cup \Gamma_c$. Here, Γ_p is the uncontrolled part of the boundary and Γ_c is the control part of the boundary. Then this flow control problem can be mathematically stated as finding the control variable g and the state variables (u, v, ω, T) that minimize the objective functional $\mathcal{L}$

$$\mathcal{L} = \frac{1}{2}\int_{\Omega} \omega^2 \, d\Omega + \frac{\gamma}{2}\int_{\Gamma_c} g^2 \, d\Gamma, \tag{5.1}$$

subject to the constraints

$$\begin{cases} -\Delta u - \dfrac{\partial \omega}{\partial y} = 0, \\ -\Delta v + \dfrac{\partial \omega}{\partial x} = 0, \\ -\dfrac{1}{Re}\Delta \omega + u\dfrac{\partial \omega}{\partial x} + v\dfrac{\partial \omega}{\partial y} - \dfrac{Gr}{Re^2}\dfrac{\partial T}{\partial x} = 0, \\ -\dfrac{1}{RePr}\Delta T + u\dfrac{\partial T}{\partial x} + v\dfrac{\partial T}{\partial y} = 0, \end{cases} \tag{5.2}$$

where u and v are the two components of the velocity field, $\omega = -\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}$ is the vorticity, T is the temperature. g is the temperature of the surrounding medium to the wall that controls the flow field via the Robin-type boundary condition, $\frac{\partial T}{\partial n} = g - T$, and γ is the weight of the regularization. In addition, Re is the Reynolds number, Gr is the Grashof number, and Pr is the Prandtl number. To close the PDE-constrained optimization system (5.1) and (5.2), some appropriate boundary conditions are required. The constraint (5.2) along with the corresponding boundary conditions is the so-called simulation or forward problem. Generally speaking, the optimization problem is more difficult to solve than the simulation problem, since the nonlinearities are more unbalanced, and an initial guess that is good for all fields is much harder to find.

To numerically solve the boundary control problem of thermally convective flows, we employ the discretize-then-optimize approach [1, 21, 23], where the objective functional and the PDE constraints are first approximated numerically before applying an optimization method to the resulting finite-dimensional nonlinear optimization problem. In our implementation, a standard second-order central finite difference scheme (both for the Laplacian operators and the first-order partial derivatives) is used to discretize (5.2) on the domain. A set of uniform grids with the grid size of h is used. Omitting the lengthy details of the discretization, we simply write the discrete optimization problem as follows:

$$\begin{cases} \min \quad \mathcal{L} = \frac{1}{2}\sum\limits_{p\in P} \omega_p^2 \, h^2 + \frac{\gamma}{2}\sum\limits_{p\in P_b} g_p^2 \, h, \\ \text{s.t.} \quad F(u_p, v_p, \omega_p, T_p, g_p) = (F_p^{(u)}, F_p^{(v)}, F_p^{(\omega)}, F_p^{(T)}) = 0, \ p \in P, \end{cases} \tag{5.3}$$

where P is the index set of all grid points, and P_b is the index set of all boundary grid points.

The first-order optimality condition says that the solution of (5.3) is the critical

point of the Lagrangian functional defined as

$$\begin{aligned}
&\mathcal{L}(u, v, \omega, T, g, \lambda_u, \lambda_v, \lambda_\omega, \lambda_T) \\
&= \frac{1}{2}\sum_{p\in P}\omega_p^2 h^2 + \frac{\gamma}{2}\sum_{p\in P_b} g_p^2 h \\
&+ \sum_{p\in P_b}(\lambda_u F_p^{(u)} + \lambda_v F_p^{(v)} + \lambda_\omega F_p^{(\omega)} + \lambda_T F_p^{(T)})h \\
&+ \sum_{p\in P\setminus P_b}(\lambda_u F_p^{(u)} + \lambda_v F_p^{(v)} + \lambda_\omega F_p^{(\omega)} + \lambda_T F_p^{(T)})h^2,
\end{aligned} \tag{5.4}$$

where λ_u is the Lagrangian multiplier or the adjoint variable with respect to the state variable u, and the others are defined in a similar way. Thus, it satisfies the Karush-Kuhn-Tucker (KKT) condition

$$\begin{aligned}
&\mathcal{K}(u, v, \omega, T, g, \lambda_u, \lambda_v, \lambda_\omega, \lambda_T) \\
&= \nabla\mathcal{L}(u, v, \omega, T, g, \lambda_u, \lambda_v, \lambda_\omega, \lambda_T) \\
&= (\mathcal{K}_{u_p}, \mathcal{K}_{v_p}, \mathcal{K}_{\omega_p}, \mathcal{K}_{T_p}, \mathcal{K}_{g_p}, \mathcal{K}_{\lambda_{u_p}}, \mathcal{K}_{\lambda_{v_p}}, \mathcal{K}_{\lambda_{\omega_p}}, \mathcal{K}_{\lambda_{T_p}}) = 0
\end{aligned} \tag{5.5}$$

The optimality system (5.5) is a large, sparse nonlinear, and multi-fields system, which consists of the following three types of nonlinear or linear systems:

- The state system is nonlinear and derived from the original Navier-Stokes equations

$$\left(\mathcal{K}_{u_p}, \mathcal{K}_{v_p}, \mathcal{K}_{\omega_p}, \mathcal{K}_{T_p}\right) = \left(F_p^{(u)}, F_p^{(v)}, F_p^{(\omega)}, F_p^{(T)}\right) = 0. \tag{5.6}$$

- The control system, which relates the control and an adjoint variable, is a linear system

$$\mathcal{K}_{g_p} = \gamma g_p + \lambda_T \frac{\partial F_p^{(T)}}{\partial g_p} = 0. \tag{5.7}$$

- The adjoint system, which is the result of differentiation with respect to state variables, is itself a nonlinear system

$$\begin{cases}
\mathcal{K}_{\lambda_{u_p}} = \displaystyle\sum_{i\in I}\lambda_{u_p} s_i \frac{\partial F_i^{(u)}}{\partial u_p} = 0, \\
\mathcal{K}_{\lambda_{v_p}} = \displaystyle\sum_{i\in I}\lambda_{v_p} s_i \frac{\partial F_i^{(v)}}{\partial v_p} = 0, \\
\mathcal{K}_{\lambda_{\omega_p}} = \omega_p h^2 + \displaystyle\sum_{i\in I}\lambda_{\omega_p} s_i \frac{\partial F_i^{(\omega)}}{\partial \omega_p} = 0, \\
\mathcal{K}_{\lambda_{T_p}} = \displaystyle\sum_{i\in I}\lambda_{T_p} s_i \frac{\partial F_i^{(T)}}{\partial T_p} = 0,
\end{cases} \tag{5.8}$$

where the index set I is a set of the standard five-point central difference stencil points associate with the grid point p, and

$$s_p = \begin{cases} h, & p \in P_b, \\ h^2, & \text{others.} \end{cases}$$

6. Numerical results and discussions. In this section, we report a series of numerical experiments for understanding the performance of the proposed algorithms. The software is developed using PETSc, [4]) library. In our implementation, the KKT conditions (5.5) are analytically formulated, and the associated Jacobian matrices are approximately constructed by using a multicolored central finite difference method [15]. For the nonlinear elimination methods, INB is employed for solving the nonlinear subspace problem. 10^{-10} (10^{-6}) is chosen as the absolute (relative) stopping condition for all nonlinear solves. To solve the global and subspace Jacobian systems, we use the one-level restricted additive Schwarz (RAS) preconditioned GMRES [14, 42] with an absolute (relative) tolerance 10^{-10} (10^{-6}). A zero initial guess is used for Newton for all test cases. We refer the methods with/without nonlinear elimination preconditioning as LNKSz-NE and LNKSz, respectively. Generally speaking, the performance of the RAS preconditioner in conjunction with a Krylov subspace method depends on several parameters, including the size of the overlap, the quality of the subdomain solve, and the subdomain partitioning, etc. Our numerical experiments suggest the number of linear iterations decreases when the overlapping size increases, which is similar to the scalar elliptic problems, but for the flow control problem, a moderate overlapping size is often needed to obtain the optimal compute time. Hence, the overlapping size for the RAS preconditioner for the global and local problems is set to be 6. Other options for RAS are set as follows. A checkerboard partition is used to obtain the subdomains, and a sparse LU decomposition is used as the subdomain solve. Other studies of RAS for flow control problems can be found in [40, 41, 49, 51].

We consider two flow control problems: a high-pressure chemical vapor deposition (CVD) reactor problem [25, 31] and a backward-facing step (BFS) flow problem [24, 31, 40]. All control problems are to seek the minimization of the objective function (5.1) with $\gamma = 0.01$. In the study, $Re = 1$ and $Pr = 0.72$ [31] are fixed. Note that typical values for Pr are around 0.7 to 0.8 for air and many other gasses. We investigate the influence of different values of Grashof number, which controls the system's nonlinearity and in particular, we are interested in the cases that LNKSz converges slowly or stagnates.

6.1. The CVD reactor problem. CVD is an important technique to deposit a layer or layers of a substance on a thin film. Mathematically, the CVD reactor problem is to find the state variables, (u, v, ω, T), and the control variable g, respectively, such that the minimization of

$$\mathcal{J} = \frac{1}{2}\int_{\Omega} |\omega|^2 \, d\Omega + \frac{\gamma}{2}\int_{\Gamma_2 \cup \Gamma_4} |g|^2 \, d\Gamma$$

is achieved subjecting to the constraints (5.2) with the following boundary conditions.

$$\begin{cases} \mathbf{v} = (0,0) \text{ and } T = 1, & \text{on} \quad \Gamma_1 \cup C_1 \cup C_2, \\ \mathbf{v} = (0,0) \text{ and } \dfrac{\partial T}{\partial n} = g - T, & \text{on} \quad \Gamma_2 \cup \Gamma_4, \\ \mathbf{v} = (0, -4(x-1/3)(2/3-x)) \text{ and } T = 0, & \text{on} \quad \Gamma_{3,m}, \\ \mathbf{v} = (0, 2x(1/3-x)) \text{ and } \dfrac{\partial T}{\partial n} = 0, & \text{on} \quad \Gamma_{3,l} \cup C_4, \\ \mathbf{v} = (0, 2(x-2/3)(1-x)) \text{ and } \dfrac{\partial T}{\partial n} = 0, & \text{on} \quad \Gamma_{3,r} \cup C_3, \end{cases}$$

where the geometry of the computational domain is shown in Fig. 6.1. The goal is to minimize the vorticity by controlling the boundary temperature near the side walls

in order to obtain a flow field with smaller circulation, which implies better vertical transport. The flow is assumed to be fully developed at the outflow boundaries, $\Gamma_{3,l}$ and $\Gamma_{3,r}$.

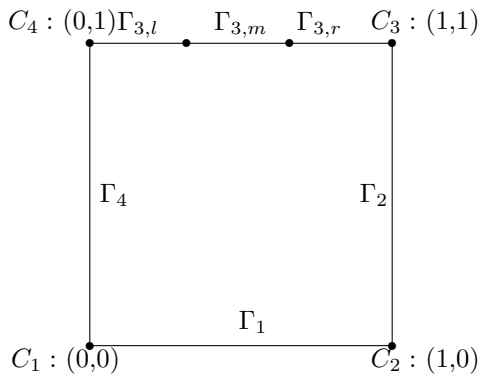

FIG. 6.1. *The CVD reactor problem. The computational domain* $\Omega = (0,1) \times (0,1)$. $\Gamma_1 = \{(x,y) : 0 < x < 1, y = 0\}$, $\Gamma_2 = \{(x,y) : x = 1, 0 < y < 1\}$, $\Gamma_3 = \{(x,y) : 0 < x < 1, y = 1\}$, $\Gamma_4 = \{(x,y) : x = 0, 0 < y < 1\}$. $\Gamma_{3,l} = \{(x,y) \in \Gamma_3 : 0 < y < 1/3\}$ *is the left outflow boundary,* $\Gamma_{3,r} = \{(x,y) \in \Gamma_3 : 2/3 < y < 1\}$ *is the right outflow boundary, and* $\Gamma_{3,m} = \{(x,y) \in \Gamma_3 : 1/3 \leq y \leq 2/3\}$ *is the inflow boundary.*

We first perform a grid convergence test to assure that the numerical solution is correct by using different grids ranging from 128×128 to 512×512. Fig. 6.2 shows the v-component of the computed velocity profiles along the horizontal line $y = 0.5$ for different grids. The convergence of the computed solutions is observed as the grid is refined for each Grashof number. The finest grid solution is used as a reference solution, and a near quadratic convergence is achieved. In Fig. 6.3, we compare the solution plots of the simulation problem and the corresponding control problem via the temperature contours for $Gr = 10^4$, 5×10^4, and 10^5, respectively. From these figures, we find that, under the mechanism of the temperature control, the change rate of the temperature in the x-direction becomes much smaller. Also shown in Fig. 6.4, the magnitude of the vorticity measured by the 2-norm is reduced more than one order of magnitude when the flow is properly controlled for different values of Grashof number. Fig. 6.5 shows a comparison of the uncontrolled and controlled vorticity contour plots for the case of $Gr = 10^5$. Two major vortices with large magnitude are completely eliminated when the temperature on the boundary is properly controlled. The stronger vorticity is localized in the top corners for the controlled case.

6.2. A slow convergence analysis of INB. We investigate how the classical LNKSz method performs for solving the CVD reactor problem with different Grashof numbers and grid sizes when no nonlinear preconditioning is applied. A summary of the nonlinear iteration counts for each case is reported in Table 6.1. For small Gr, the convergence of Newton is quite fast, and the rate is independent of the grid sizes. On the other hand, when the grid size is fixed, the number of Newton iterations increases as Gr increases until some critical Gr is reached, where the classical Newton method begins to stagnate.

In the left figure of Fig. 6.6, we show the LNKSz convergence history of the non-

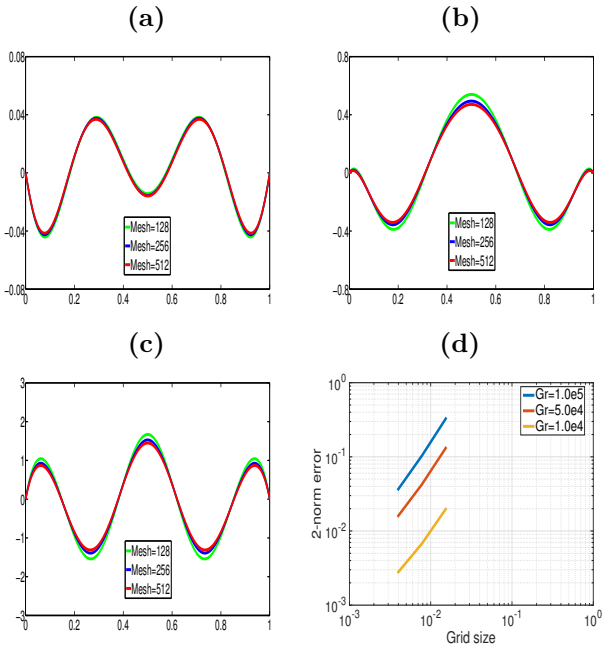

FIG. 6.2. *The CVD reactor problem. A comparison of the v-component of the velocity along the horizontal line $y = 0.5$ with different grid sizes for three values of the Grashof numbers.* (**a**) $Gr = 10^4$; (**b**) $Gr = 5 \times 10^4$; (**c**) $Gr = 10^5$. *In addition,* (**d**) *The plot of the 2-norm error of the v-component versus the grid size in the logarithmic scale for different Gr. The orders of the convergence for $Gr = 10^4$, 5×10^4, and 10^5 are 1.75, 1.78, and 1.78, respectively.*

linear residual norm with different Gr. The method exhibits a typical convergence behavior; the iteration stagnates longer and longer as Gr increases. When the approximate solution moves into the ball of convergence, then quadratic convergence is suddenly achieved. The plateau in the left figure of Fig. 6.6 is an indication that some nonlinear preconditioning is necessary, but it does not tell which part of the nonlinear problem is responsible for the slowness of the convergence. To further analyze the slow convergence, we look at the residual vectors more carefully – component by component and field by field. The right figure of Fig. 6.6 shows a field by field plot of the residual function, and it is clear that the temperature component dominates the residual. Therefore, it is chosen as the component, together with its Lagrangian multiplier, to be eliminated. Fig. 6.7 shows the optimal control variable, i.e., the temperature along the boundary $\Gamma_2 \cup \Gamma_4$. As Gr increases, the gradient becomes sharper, which makes the resulting KKT condition (5.5) more nonlinear, as a result, Newton is having difficulty to converge.

Note that the optimal T is nearly linear everywhere in the computational domain, except for the two corners near the cold wall; see the second row of Fig. 6.3. To further understand the slow convergence of LNKSz, we plot some of the intermediate solutions for T in Fig. 6.8 at few selected Newton iterations as well as some of the temperature profiles along the vertical line at $x = 0.5$. For this case, $Gr = 2 \times 10^4$ is set, and a 64×64 grid is used. LNKSz takes 17 iterations to converge. During the stagnation period, LNKSz can roughly catch certain features of the solution such as $\partial T/\partial x = 0$ and $\partial T/\partial y$ is constant in the y-direction almost everywhere. But the value of $\partial T/\partial y$ is incorrect. From the bottom-right figure of Fig. 6.8, we observe that

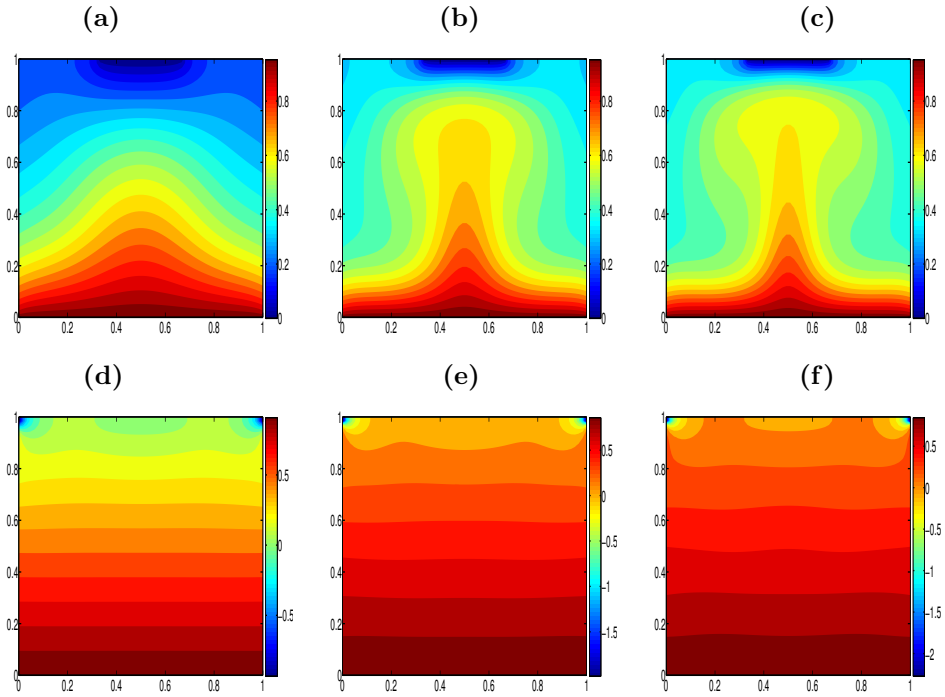

FIG. 6.3. *The CVD reactor problem. The temperature contour plots for different value of Gr. The Reynolds number is fixed to Re = 1 and the grid is 64×64.* (**a**) *Simulation problem at* $Gr = 10^4$*;* (**b**) *Simulation problem at* $Gr = 5 \times 10^4$*;* (**c**) *Simulation problem at* $Gr = 10^5$*;* (**d**) *Control problem at* $Gr = 10^4$*;* (**e**) *Control problem at* $Gr = 5 \times 10^4$*;* (**f**) *Control problem at* $Gr = 10^5$.

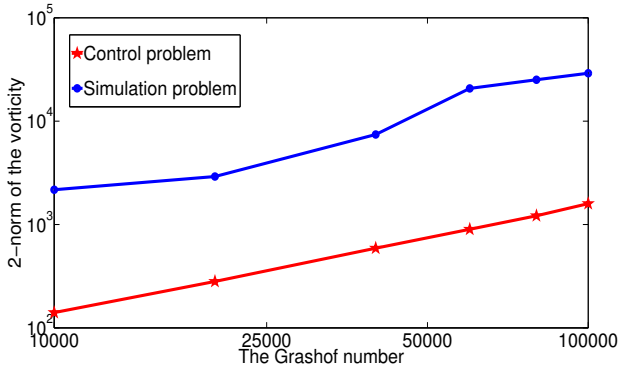

FIG. 6.4. *The CVD reactor problem. The 2-norm of vorticity for the control and simulation problems with different Grashof numbers on a* 64×64 *grid. In the figure, "Simulation problem" denotes the uncontrolled flow and "Control problem" denotes the controlled flow.*

LNKSz gradually correct $\partial T/\partial y$ by increasing its value at a very slow rate; moreover, this correction is global in the sense that all points in the computational domain are involved.

Note that the convergence behavior of INB for the flow control problems is quite different from the problems with local nonlinearity, such as boundary layers, shock

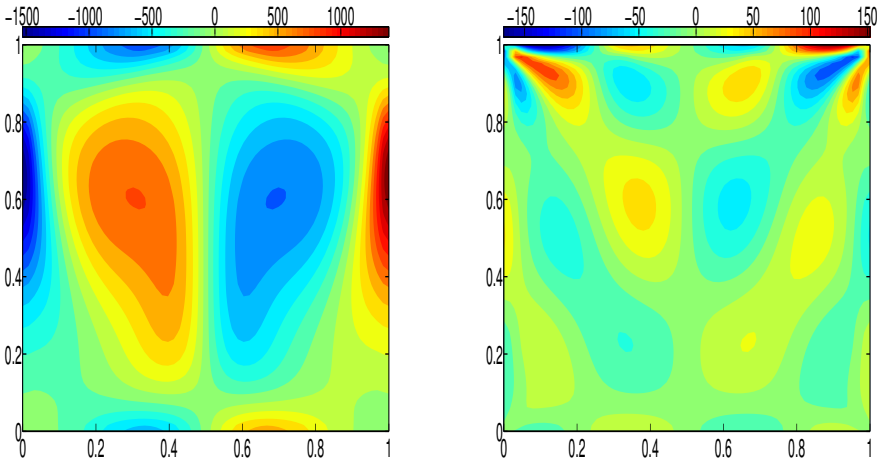

FIG. 6.5. *The CVD reactor problem. A comparison of the vorticity contour plots for* $Gr = 10^5$. *Left: the uncontrolled flow and right: the controlled flow.*

TABLE 6.1

The number of nonlinear iterations for the classical LNKSz method for the CVD reactor problem for different Grashof numbers and grid sizes. In the table, "Newton" denotes the number of inexact Newton iterations and "–" means the failure of the Newton iteration.

CVD reactor problem				
Gr	64×64	128×128	256×256	512×512
10^4	3	3	3	3
2×10^4	17	11	6	4
4×10^4	41	25	25	23
6×10^4	–	104	48	––
8×10^4	–	–	–	–
10^5	–	–	–	–

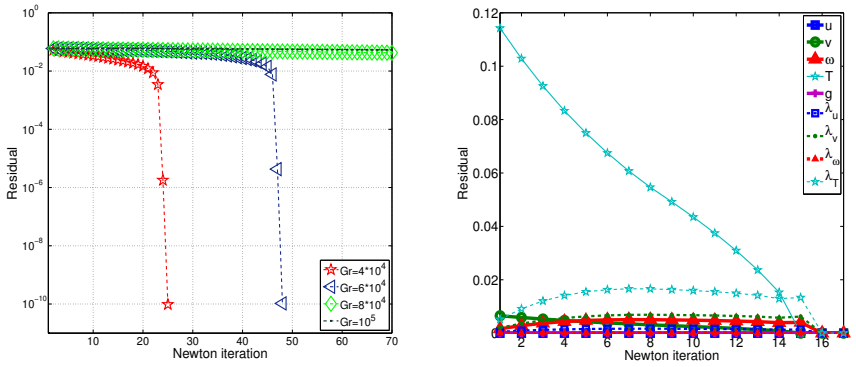

FIG. 6.6. *The CVD reactor problem. Left: The convergence history plots of the global nonlinear residual norms for different Grashof numbers* $Gr = 4 \times 10^4$, 6×10^4, 8×10^4, *and* 10^5 *on a* 256×256 *grid. For* $Gr = 8 \times 10^4$ *and* 10^5, *LNKSz does not converge after 70 Newton iterations. Right: The convergence history plots of the nonlinear residual norms for all nine components obtained by LNKSz, which requires the total number of 17 nonlinear iterations to converge. The calculation is carried out for* $Gr = 2 \times 10^4$ *on a* 64×64 *grid.*

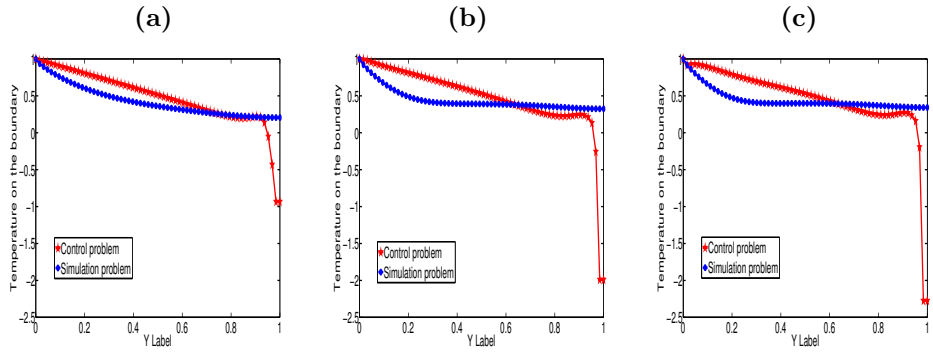

FIG. 6.7. *The CVD reactor problem. Computed temperature on the boundary $x = 0$ (i.e., Γ_4) with different Grashof numbers on a 64×64 grid. $Re = 1$ and $Pr = 0.72$.* (**a**) $Gr = 10^4$; (**b**) $Gr = 5 \times 10^4$; (**c**) $Gr = 10^5$.

wave or corner singularity. For example, for the transonic flow problem [12, 30], the number of INB iterations typically increases as the grid is refined. When the shock location is highly resolved, the nonlinear problem is harder to solve. In such case, the high nonlinearity is "local". But for the control problem, sometimes, the coarse problem is more difficult to solve than the fine grid problem for some value of Gr. This is an indication that some component is "globally" bad on the whole domain. In such a situation, nonlinear elimination based on the field partition is more appropriate than the one based on the subdomain partition, which is more suitable to problems with local high nonlinearities.

It is important to identity the right field to eliminate, otherwise, the convergence of the outer Newton iteration is not improved much. An iteration count comparison of two different elimination choices (the field variables ω or T in Table 6.2) confirms this. If we select the *correct* components, i.e., T and λ_T in this case, the algorithm becomes very robust and efficient. Only 4 or 5 global Newton iterations is needed and the number of linear iterations increases slightly as Gr increases. If the fields corresponding to ω and λ_ω are eliminated, the convergence deteriorated drastically as Gr is increased.

TABLE 6.2

A comparison of two different elimination strategies for the CVD reactor problem on a 128×128 grid. In the table, "Newton" denotes the number of the global Newton iterations, "GMRES" denotes the average number of GMRES iterations per Newton iteration, and "Time" the total computing time in seconds. "Global" denotes the performance of LNKSz–NE in the global update phase and "Local" denotes the performance of LNKSz-NE in the subspace correction phase. "–" means the failure of the Newton iteration.

	Eliminate ω and λ_ω strategy LNKSz-NE: Global (Local)			Eliminate T and λ_T strategy LNKSz-NE: Global (Local)		
Gr	Newton	GMRES	Time	Newton	GMRES	Time
10^4	4 (2)	50.2 (11.5)	10.7 (1.6)	4 (2)	50.0 (16.0)	10.6 (1.9)
2×10^4	11 (9)	269.8 (16.0)	107.7 (9.3)	4 (5)	61.2 (18.0)	12.6 (5.4)
4×10^4	23 (21)	459.7 (15.6)	360.9 (21.5)	4 (5)	61.2 (18.0)	12.6 (5.4)
6×10^4	93 (91)	971.0 (15.6)	3453.1 (101.4)	5 (6)	74.6 (18.0)	17.5 (6.1)
8×10^4	–			5 (6)	76.8 (18.5)	18.6 (6.5)
10^5	–			5 (7)	83.8 (18.7)	19.5 (7.4)

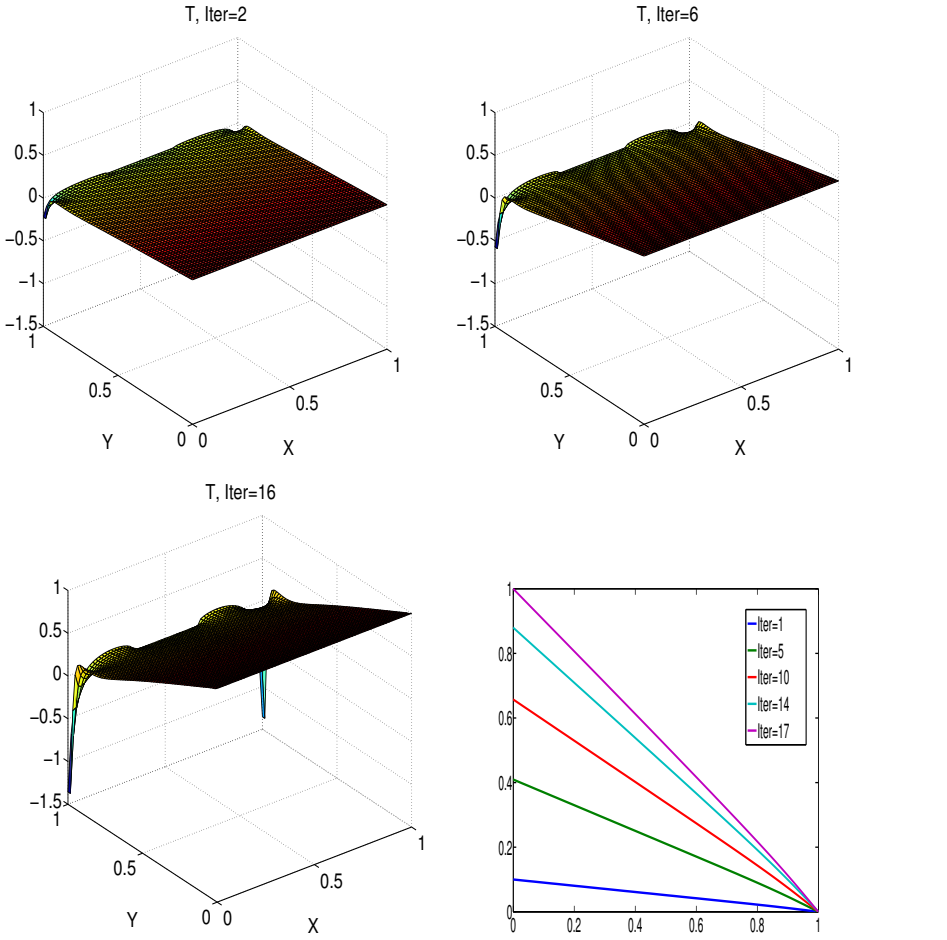

FIG. 6.8. *The CVD reactor problem. Computed intermediate solution for the temperature at the 2nd (top-left), 6th (top-right), and 16th (bottom-left) Newton iteration for the Grashof number, $Gr = 2 \times 10^4$ on a 64×64 grid. The the temperature profile curves along the vertical line at $x = 0.5$ for few selected Newton iterations (bottom-right).*

6.3. A comparison with a two-grid method. In this section, we compare LNKSz-NE with a two-grid version of the LNKSz method as described in Algorithm 2. The numerical experiments are carried out on two fixed grids, 256×256 and 512×512 with the Grashof number varies from 10^4 to 10^5. For the two-grid method, the size of the coarse grid H is taken as $4h$. Recall that Table 6.1 reveals that for LNKSz, the number of Newton iterations increases dramatically as the Grashof number increases due to the high nonlinearity of the system. For the two-grid method, the nonlinear iterations become acceptable and, compared with the one-grid LNKSz method, the total computing time is much smaller. As a result, the two-grid method performs so much better. However, with the further increase of the Grashof number, neither LNKSz nor the two-grid method is convergent. Next we test the proposed LNKSz-NE. In Table 6.3, we show that the convergence is dramatically improved when LNKSz is applied with NE preconditioning, interestingly, the number of linear iterations is also substantially reduced. Compared with the two-grid method, LNKSz-

NE is much more effective, in terms of the total number of outer Newton iterations and the total compute time. Here, we only consider a two-grid LNKSz method. Alternatively, the full-approximation scheme (FAS) (or nonlinear multigrid (NLMG) from Hackbusch) [46], which has a correction term on the right-hand side of the coarse problem, might be able to push the interpolant into the domain of convergence on the fine grid.

TABLE 6.3

A comparison of LNKSz and LNKSz-NE for the CVD reactor problem. In the table, "Newton" denotes the number of inexact Newton iterations, "GMRES" denotes the average number of GMRES iterations per Newton iteration, and "Time" the total computing time in seconds. "LNKSz-NE: Global" denotes the performance of LNKSz-NE in the global iteration and "LNKSz-NE: Local" denotes the performance of LNKSz-NE in the subspace iteration. "–" means divergence of the Newton iteration on the coarse grid.

Gr	LNKSz–NE: Global (Local)			Two-grid method: Fine (coarse)		
	Newton	GMRES	Time	Newton	GMRES	Time
	256×256 grid and $N_p = 64$					
10^4	4 (2)	115.2 (28)	18.1 (3.1)	3 (4)	108.0 (75.3)	13.3 (3.3)
2×10^4	4 (4)	141.7 (29.7)	21.2 (6.1)	3 (17)	127.6 (341.4)	10.0 (35.1)
4×10^4	5 (6)	162.2 (32.0)	36.6 (9.5)	3 (41)	160.0 (481.3)	25.5 (107.1)
6×10^4	6 (6)	213.3 (34.0)	54.7 (10.1)	4 (183)	162.0 (864.5)	47.9 (801.5)
8×10^4	6 (6)	248.5 (34.0)	65.4 (10.1)	–		
10^5	6 (6)	339.3 (35.3)	91.3 (11.3)	–		
	512×512 grid and $N_p = 256$					
10^4	4 (1)	470.2 (78.0)	87.8 (4.0)	3 (4)	312.3 (152.7)	42.1 (8.9)
2×10^4	5 (4)	433.6 (56.5)	100.5 (11.6)	3 (11)	390.6 (395.7)	46.1 (36.5)
4×10^4	5 (5)	554.8 (64.0)	135.2 (16.4)	3 (25)	522.3 (591.6)	66.7 (106.4)
6×10^4	6 (6)	613.6 (65.0)	172.9 (19.6)	4 (104)	481.7 (922.0)	96.7 (616.1)
8×10^4	6 (6)	679.0 (67.0)	204.1 (19.8)	–		
10^5	6 (6)	662.6 (69.0)	186.2 (20.2)	–		

6.4. The backward-facing step flow problem. To illustrate the applicability of the LNKS-NE algorithm, we consider a BFS flow problem, which is stated as follows. Find (u, v, ω, T, g) such that

$$\min \mathcal{J} = \frac{1}{2} \int_{\Omega} |\omega|^2 \, d\Omega + \frac{\gamma}{2} \int_{\Gamma_1 \cup \Gamma_3} |g|^2 \, d\Gamma$$

subject to the constraints (5.2) with the following boundary conditions.

$$\begin{cases} \mathbf{v} = (0,0) \text{ and } \dfrac{\partial T}{\partial n} = g - T, & \text{on} \quad \Gamma_1 \cup \Gamma_3, \\ \mathbf{v} = (y(1-y), 0) \text{ and } \dfrac{\partial T}{\partial n} = 0, & \text{on} \quad \Gamma_2 \cup C_2 \cup C_3, \\ \mathbf{v} = (8(1-y)(y - \dfrac{1}{2}), 0) \text{ and } T = 0, & \text{on} \quad \Gamma_{4,i} \cup C_4, \\ \mathbf{v} = (0,0) \text{ and } T = 1, & \text{on} \quad (\Gamma_4 - \Gamma_{4,i}) \cup C_1, \end{cases}$$

The geometrical configuration for the BFS problem is shown in Fig. 6.9, and the temperature control is applied on Γ_1 and Γ_3. Fig. 6.10 shows the vorticity contour plots for the simulation and controlled cases. The circulation in the controlled case is clearly smaller than the one without control.. For $Gr = 10^4$, Fig. 6.11 presents a comparison of the magnitude of the vorticity in the 2-norm for the uncontrolled and controlled problems. We vary the Grashof number from 10^4 to 10^5, and fix the Prandtl number to $Pr = 0.72$. From Fig. 6.11, we see that, as the Grashof number

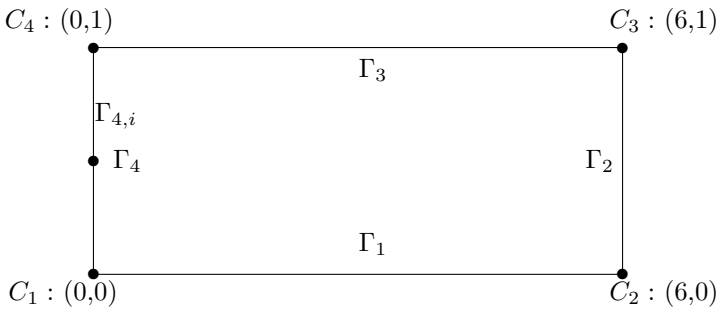

FIG. 6.9. *The BFS flow problem.* $\Omega = (0,6) \times (0,1)$. $\Gamma_1 = \{(x,y) : 0 < x < 6, y = 0\}$, $\Gamma_2 = \{(x,y) : x = 6, 0 < y < 1\}$, $\Gamma_3 = \{(x,y) : 0 < x < 1, y = 1\}$, $\Gamma_4 = \{(x,y) : x = 0, 0 < y < 1\}$, *and* $\Gamma_{4,i} = \{(x,y) \in \Gamma_4 : 0.5 \leq y < 1\}$. $\Gamma_{4,i}$ *(*Γ_2*) is the inlet (outlet) boundary, respectively.*

increases, the the magnitude of the vorticity also grows for both the simulation and the control problems. But the order of magnitude of the vorticity for the control problem is smaller than of the simulation problem.

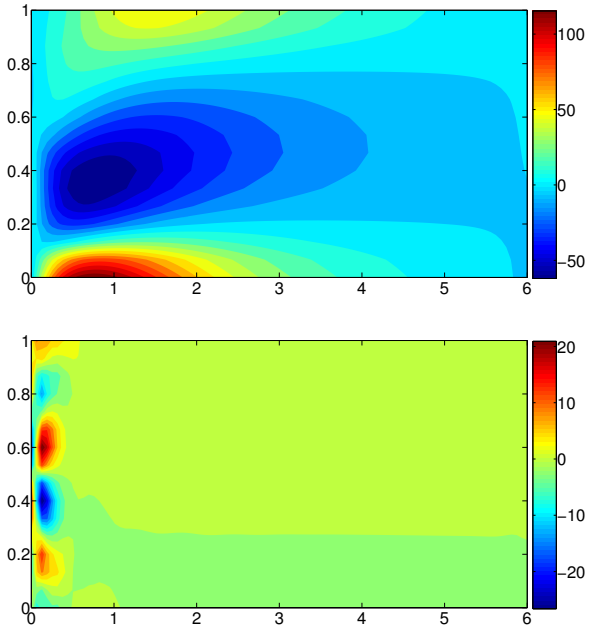

FIG. 6.10. *The BFS flow problem. The vorticity* ω *contour plots* $Gr = 10^4$ *on a* 96×16 *grid. The top figure is the uncontrolled solution and the bottom figure is the corresponding controlled solution.*

Fig. 6.12 compares the convergence histories of LNKSz and LNKSz-NE with different Grashof numbers on a 384×64 grid. Again, it is clear that LNKSz-NE is more robust and efficient than LNKSz.

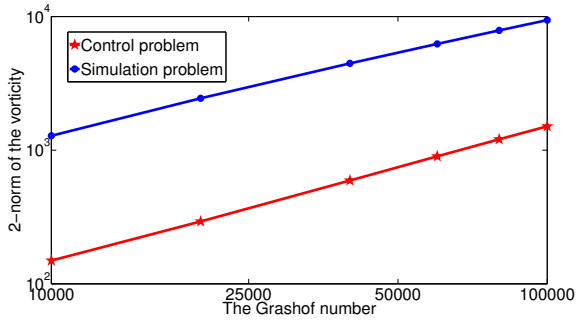

FIG. 6.11. *The BFS flow problem. The 2-norm of vorticity for the control and simulation problems with different Grashof numbers. The Reynolds number is fixed to* $Re = 1$ *and the grid is* 96×16.

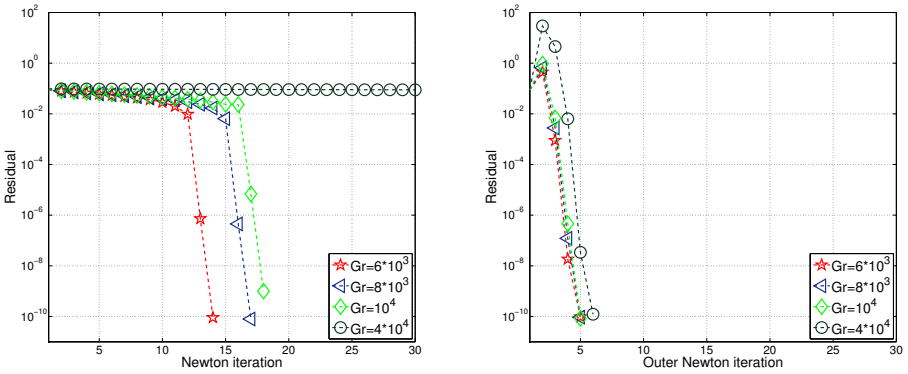

FIG. 6.12. *Nonlinear residual histories of LNKS (left) and LNKS-NE (right) for the BFS flow problem with different Grashof numbers* $Gr = 6 \times 10^3$, 8×10^3, 10^4, *and* 4×10^4 *on a* 384×64 *grid.*

7. Concluding remarks. In the paper, we developed a nonlinearly preconditioned full-space Lagrange-Newton method for solving PDE-constrained optimization problems. The proposed right preconditioner is based on the nonlinear elimination of certain field variables whose residual norms dominate the global residual during the stagnation period of the outer inexact Newton iterations. The new approach consists of two major ingredients: a subspace correction and a global update by the outer Newton iterations. We validated our implementation and evaluated the performance of the proposed full-space method on two benchmark problems, including a chemical vapor deposition reactor problem and a backward-facing step flow problem. Our numerical experiments showed that the number of outer Newton iterations could be drastically reduced when certain "bad" field variables were correctly identified and effectively eliminated by the subspace correction. We restricted our discussion on flow control problems in 2D, but we believe the general approach should be applicable to other PDE-constrained optimization problems and problems in 3D.

REFERENCES

[1] V. AKCELIK, G. BIROS, O. GHATTAS, J. HILL, D. KEYES, AND B. VAN BLOEMEN WAANDERS,

Parallel algorithms for PDE-constrained optimization, in Parallel Processing for Scientific Computing, M. A. Heroux, P. Raghavan, and H. D. Simon, eds., SIAM, Philadelphia, 2006.

[2] E. L. Allgower and K. Georg, *Continuation and path following*, Acta Numer., 2 (1992), pp. 1–64.

[3] V. F. De Almeida and J. J. Derby, *Construction of solution curves for large two-dimensional problems of steady-state flows of incompressible fluids*, SIAM J. Sci. Comput., 22 (2000), pp. 285–311.

[4] S. Balay, S. Abhyankar, F. A. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M.G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, and H. Zhang, *PETSc Web page*. http://www.mcs.anl.gov/petsc, 2015.

[5] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[6] T. R. Bewley, *Flow control: new challenges for a new renaissance*, Prog. Aerosp. Sci., 37 (2001), pp. 21–58.

[7] G. Biros and O. Ghattas, *Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization. Part I: The Krylov-Schur solver*, SIAM J. Sci. Comput., 27 (2005), pp. 687–713.

[8] ———, *Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization. Part II: The Lagrange-Newton solver and its application to optimal control of steady viscous flows*, SIAM J. Sci. Comput., 27 (2005), pp. 714–739.

[9] E. Bodenschatz, W. Pesch, and G. Ahlers, *Recent developments in Rayleigh-Bénard convection*, Annu. Rev. Fluid Mech., 32 (2000), pp. 709–778.

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

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

[12] X.-C. Cai, D. E. Keyes, and D. P. Young, *A nonlinear additive Schwarz preconditioned inexact Newton method for shocked duct flow*, in Proceedings of the 13th International Conference on Domain Decomposition Methods, 2001.

[13] X.-C. Cai and X. Li, *Inexact Newton methods with restricted additive Schwarz based nonlinear elimination for problems with high local nonlinearity*, SIAM J. Sci. Comput., 33 (2011), pp. 746–762.

[14] 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.

[15] T. F. Coleman and J. J. Moré, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM J. Numer. Anal., 20 (1983), pp. 187–209.

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

[17] M. Desai and K. Ito, *Optimal control of Navier-Stokes equations*, SIAM J. Optim., 32 (1994), pp. 1428–1446.

[18] V. Dolean, M. J. Gander, W. Kheriji, F. Kwok, and R. Massin, *Nonlinear preconditioning: How to use a nonlinear Schwarz to precondition Newton's method*, Preprint, (2015).

[19] S. C. Eisenstat and H. F. Walker, *Globally convergent inexact Newton method*, SIAM J. Optim., 4 (1994), pp. 393–422.

[20] A. V. Fursikov, *Optimal Control of Distributed Systems, Theory and Applications*, American Mathematical Society, 2000.

[21] O. Ghattas and J.-H. Bark, *Optimal control of two- and three-dimensional incompressible Navier-Stokes flows*, J. Comput. Phys., 136 (1997), pp. 231–244.

[22] C. Gross and R. Krause, *On the globalization of ASPIN employing trust-region control strategies-convergence analysis and numerical examples*, Tech. Rep. 2011-03, Institute of Computational Science, Universita Della Svizzera Italian, 2011.

[23] M Gunzburger, *Perspectives in Flow Control and Optimization*, SIAM, Philadelphia, 2003.

[24] M. Gunzburger, L. Hou, and T. Svobodny, *The approximation of boundary control problems for fluid flows with an application to control by heating and cooling*, Comput. Fluids, 22 (1993), pp. 239–251.

[25] M. Gunzburger and S. Manservisi, *The velocity tracking problem for Navier-Stokes flow with boundary control*, SIAM J. Numer. Anal., 39 (2000), pp. 594–634.

[26] J. Huang, C. Yang, and X.-C. Cai, *A nonlinearly preconditioned inexact Newton algorithm for steady lattice Boltzmann equations*, SIAM J. Sci. Comput., 38 (2016), pp. A1701–A1724.

[27] F.-N. Hwang and X.-C. Cai, *Improving robustness and parallel scalability of Newton method*

through nonlinear preconditioning, in Lecture Notes in Computational Science and Engineering, R. Kornhuber, R. H. W. Hoppe, D. E. Keyes, J. Periaux, and Pironneau O., eds., Springer-Verlag, Heidelberg, 2004, pp. 201–208.

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

[29] 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.

[30] 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.

[31] K. Ito and S. S. Ravindran, *Optimal control of thermally convected flows*, SIAM J. Sci. Comput., 19 (1998), pp. 1847–1869.

[32] D. A. Knoll and D. E. Keyes, *Jacobian-free Newton-Krylov methods: a survey of approaches and applications*, J. Comput. Phys., 193 (2004), pp. 357–497.

[33] 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.

[34] W. Layton, H. K. Lee, and j. Peterson, *Numerical solution of the stationary Navier-Stokes equations using a multilevel finite element method*, SIAM J. Sci. Comput., 20 (1998), pp. 1–12.

[35] L. Liu and D. E. Keyes, *Field-split preconditioned inexact Newton algorithms*, SIAM J. Sci. Comput., 37 (2015), pp. A1388–A1409.

[36] M. C. Navarro and H. Herrero, *Effects of optimal control over thermoconvective patterns*, Phys. Rev. E, 75 (2007), p. 067203.

[37] J. Nocedal and S. Wright, *Numerical Optimization*, Springer-Verlag, New York, 2nd ed., 2006.

[38] S. B. G. M. O'Brien, *On marangoni drying: Nonlinear kinematic waves in a thin film*, J. Fluid Mech., 254 (1993), pp. 649–670.

[39] N. Postacioglu, P. Kapadia, and J. Dowden, *A theoretical model of thermocapillary flows in laser welding*, J. Phys. D. Appl. Phys., 24 (1991), p. 15.

[40] E. Prudencio, R. Byrd, and X.-C. Cai, *Parallel full space SQP Lagrange-Newton-Krylov-Schwarz algorithms for PDE-constrained optimization problems*, SIAM J. Sci. Comput., 27 (2006), pp. 1305–1328.

[41] E. Prudencio and X.-C. Cai, *Parallel multilevel restricted Schwarz preconditioners with pollution removing for PDE-constrained optimization*, SIAM J. Sci. Comput., 29 (2007), pp. 964–985.

[42] Y. Saad, *Iterative Methods for Sparse Linear Systems,*, SIAM, Philadelphia, 2nd ed., 2004.

[43] 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.

[44] J. E. Spinelli, L. Ferreira, and A. Garcia, *Influence of melt convection on the columnar to equiaxed transition and microstructure of downward unsteady-state directionally solidified snpb alloys*, J. Alloys. Compd., 384 (2004), pp. 217–226.

[45] S. Sun, D. E. Keyes, and L. Liu, *Fully implicit two-phase reservoir simulation with the additive Schwarz preconditioned inexact Newton method.*, in SPE Reservoir Characterization and Simulation Conference and Exhibition, Society of Petroleum Engineers, 2013.

[46] U. Trottenberg, C.W. Oosterlee, and A. Schuller, *Multigrid*, Academic press, 2000.

[47] R. S. Tuminaro, H. F. Walker, and J. N. Shadid, *On backtracking failure in Newton-GMRES methods with a demonstration for the Navier-Stokes equations*, J. Comput. Phys., 180 (2002), pp. 549–558.

[48] H. Yang and X.-C. Cai, *Parallel two-grid semismooth Newton-Krylov-Schwarz method for nonlinear complementarity problems*, J. Sci. Comput., 47 (2011), pp. 258–280.

[49] ———, *Scalable parallel algorithms for boundary control of thermally convective flows*, in 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012, pp. 1387–1396.

[50] ———, *Parallel fully implicit two-grid Lagrange-Newton-Krylov-Schwarz methods for distributed control of unsteady incompressible flows*, Int. J. Numer. Meth. Fluids, 73 (2013), pp. 1–21.

[51] H. Yang, E. Prudencio, and X.-C. Cai, *Fully implicit Lagrange-Newton-Krylov-Schwarz algorithms for boundary control of unsteady incompressible flows*, Int. J. Numer. Meth. Engng., 91 (2012), pp. 644–665.