AIAA AVIATION Forum
June 25-29, 2018, Atlanta, Georgia
2018 Fluid Dynamics Conference

10.2514/6.2018-4162

Immersed Boundary Method as an Inverse Problem

Jianfeng Yan* and Jason E. Hicken†
Rensselaer Polytechnic Institute, Troy, New York, 12180

In this paper we propose a new immersed boundary method (IBM). To enforce Dirichlet boundary conditions, control variables are introduced element boundaries that approximate the exact boundary. The control variables are determined by solving an inverse problem. The proposed method is high-order accurate and agnostic to the discretization. To verify the accuracy of the scheme, convection-diffusion problems are solved with a discontinuous Galerkin finite element discretization. The Karush-Kuhn-Tucker (KKT) system is solved in the full space, and two preconditioners are developed and tested.

I. Introduction

In spite of the rapid development of computational fluid dynamics (CFD) in the past several decades, its application in conceptual design remains limited. This is partly because CFD simulations are relatively expensive. Nevertheless, with the development of increasingly powerful computers and algorithms, we can expect that the computational cost will diminish as a bottleneck. Another reason, which is of more importance in the long term, is mesh generation: automatic mesh generation for complex geometries remains challenging[1], especially generation of high-order meshes. For example, gaps in CAD models are common at intersections of different parts, but mesh generation needs a watertight geometry, which makes human intervention inevitable.

Immersed boundary methods (IBMs) can significantly improve the automation of mesh generation and, hence, enable the use of CFD during conceptual design. Compared to body-fitted methods, which require a mesh that conforms to the problem domain, IBM meshes can be generated almost arbitrarily, irrespective of the problem domain.

Simplified mesh generation comes at the expense of complicating the imposition of boundary conditions. The momentum-force-based implementation of the IBM, which was first introduced by Peskin[2], introduces an additional concentrated force on the domain boundaries to model the action of moving boundaries. Dirac-delta functions, used to model the concentrated force, are then regularized to spread the force field over the neighboring cells. One can show that this approach is equivalent to adding penalty terms in finite element methods[3]. Variations in this class involve the choice of regularization applied to the delta functions and the user specified magnitude of the concentrated force[4–6]. While these methods have proven useful for low and moderate Reynolds number flows, it is challenging to extend this type of method to higher Reynolds numbers, since the regularization of the delta function deteriorates the local accuracy in resolving the boundary layer.

The interpolation-based approach, also known as the sharp immersed interface method, was proposed to overcome the local accuracy problem described above while achieving second or higher-order accuracy[7]. In this approach the precise value of the solution can be implicitly imposed on the immersed boundary through an interpolation scheme; therefore, ad-hoc parameters are avoided. Furthermore, desirable properties can be achieved by carefully selecting the stencil and tuning the interpolation coefficients; examples of such properties include accuracy[8], stability[9], and improved conditioning[10]. However, since most methods of this type are based on finite difference discretizations, they are restricted to grids with (possibly mapped) quadrilateral elements in two-dimensions (2D) and hexahedral elements in three-dimensions (3D), which are not always well-suited for mesh adaptation. Furthermore, the interpolation schemes are typically constructed based on a one-dimensional analysis of a specific problem, so the desired properties (e.g. stability) may not generalize to multiple dimensions and other physical problems.

Immersed boundary conditions can be imposed in finite-element using a Lagrange-multiplier formulation in which a Lagrange multiplier is introduced to impose the boundary condition as a constraint. Although this approach can be applied to various physical problems, the proper choice of the Lagrange multiplier space is challenging, since the multiplier function space and the solution space should satisfy the so-called inf-sup condition[11]. Otherwise, severe oscillations along the Dirichlet boundary, known as boundary locking[12, 13], may occur. Unfortunately, proving that

*Ph.D student, Department of Mechanical, Aerospace and Nuclear Engineering, AIAA Student Member
†Assistant Professor, Department of Mechanical, Aerospace and Nuclear Engineering, AIAA Member

the inf-sup condition is satisfied *a priori* is difficult, if not impossible. To the best of our knowledge, few high-order accurate results are reported using this approach.

In this work we propose a new approach that is high-order accurate and agnostic to the discretization. It is similar to the momentum-force-based methods. However, instead of adding a concentrated force along the exact Dirichlet boundary, we introduce control variables that are distributed along an approximate boundary that consists merely of element faces. The significance of doing this is that it allows us to eliminate the delta function and, consequently, its problematic regularization. An inverse problem is solved to find the control variables that best satisfy the boundary conditions. Compared to the Lagrange-multiplier approach, no special function space is required. The approach requires the solution of a PDE-constrained optimization problem, which can be computationally expensive in general. Fortunately, the structure of the problem can be exploited to develop effective preconditioners[14].

The remainder of the paper is organized as follows. In Section II we introduce the framework of the proposed method; in Section III we perform a mesh convergence study to investigate the accuracy of the proposed method; in Section IV we discuss preconditioners for the iterative solution of the inverse problem.

II. Proposed method

In this section we first introduce the formulation of the inverse problem used to impose Dirichlet boundary conditions for a generic boundary value problem (BVP). We then provide the first-order optimality conditions and the corresponding Karush-Kuhn-Tucker (KKT) system.

Let $\Omega \subset \mathbb{R}^2$ be a two dimensional domain. We consider the following BVP:

$$\begin{aligned} R(u) &= 0, \qquad && \forall x \in \Omega, \\ u &= u_D, && \forall x \in \partial\Omega, \end{aligned} \tag{1}$$

where $R(u)$ denotes a (possibly nonlinear) partial differential operator, Ω is the problem domain, and u_D is the boundary value on the Dirichlet boundary $\partial\Omega$. In the optimal control community, the solution u is also called the state variable. To simplify the presentation, we focus exclusively on Dirichlet boundary conditions, but the proposed method can easily be generalized to include, for example, Neumann boundary conditions.

A. Problem formulation

As with other immersed boundary methods, the physical domain Ω is extended into a larger computational domain $\tilde{\Omega} \supseteq \Omega$ such that the computational boundary $\partial\tilde{\Omega}$ is an approximation of the physical boundary $\partial\Omega$. This is illustrated in Figure 1, in which Ω is the shaded area, and the computational domain is the domain consisting of all the triangles.

Rather than discretizing the BVP (1) on Ω directly, we instead propose the following discretized inverse problem on $\tilde{\Omega}$:

$$\begin{aligned} \min_{u_h, c_h} \quad & J_h(u_h, c_h) = \|u_h - u_D\|^2_{\partial\Omega} + \|c_h - u_D\|^2_{\partial\tilde{\Omega}} \\ \text{s.t.} \quad & R_h(u_h, c_h) = 0 \quad \text{on } \tilde{\Omega}. \end{aligned} \tag{2}$$

Here, u_h is the discrete solution defined on the computational domain $\tilde{\Omega}$, and c_h is the control variable defined on the computational boundary $\partial\tilde{\Omega}$. The control variable is the Dirichlet boundary value in the discretization of the BVP, denoted by R_h, which itself serves as the equality constraint in the above optimization problem. Note that R_h can be based on any type of discretization that uses weakly imposed boundary conditions, including finite difference, finite volume or finite element. Discretizations that use strongly imposed boundary conditions have not been considered in this work, but we believe the general approach could be adapted to such methods.

There are two terms in the objective J_h, and both involve the (squared) norm of a difference defined on a boundary. For example, since we use a discontinuous Galerkin (DG) method to discretize R_h in this work, a straightforward choice for the objective is

$$J_h(u_h, c_h) = \int_{\partial\Omega} (u_h - u_D)^2 \mathrm{d}\Gamma + \int_{\partial\tilde{\Omega}} (c_h - u_h)^2 \mathrm{d}\Gamma.$$

The first term above, referred to as the misfit in the inverse-problem literature, evaluates the discrepancy between the discrete solution u_h and the boundary value u_D along the true boundary Γ_B. The second term is a Tikhonov-type regularization used to stabilize the inverse problem.

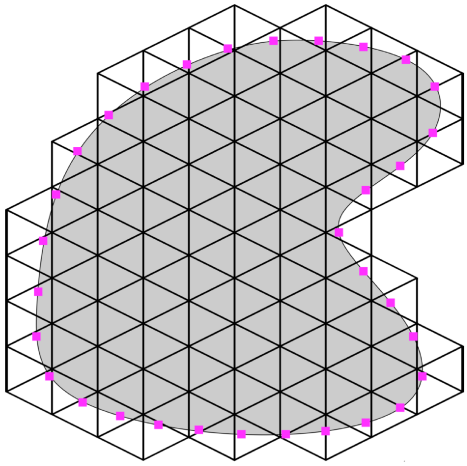


Fig. 1 Example illustrating the physical domain Ω (the shaded domain) and the computational domain $\tilde{\Omega}$ (the domain consisting of all triangles).

Our regularization is different from the typical Tikhonov regularization in the sense that the discrete solution u_h is used as the target value for the control variable c_h. The effectiveness of this choice is due to *a priori* knowledge of the control variable from the equality constraint $R_h = 0$. Recall that c_h is the boundary value of u_h weakly imposed on $\partial\tilde{\Omega}$. Therefore, assuming the discretization R_h produces a state solution of $p+1$ order of accuracy, and the mesh element size is h, then $\|u_h - c_h\|_{\partial\tilde{\Omega}} \sim h^{p+1}$. That is, using u_h as the background state will significantly reduce the magnitude of the regularization and, therefore, diminish the negative influence of it on the accuracy of the solution. Another unique feature of the regularization in (2) is that it is not scaled by a regularization parameter; such regularization parameters are typically necessary in inverse problems and their determination can be problematic. Furthermore, assuming the inverse problem (2) can be solved, we hypothesize that the misfit term will also be driven to the same order, i.e., $\|u_h - u_D\|_{\partial\Omega} \sim h^{p+1}$.

The regularization is easy to compute since both u_h and c_h are defined on $\partial\tilde{\Omega}$. On the other hand, in order to evaluate the misfit term the discrete solution must be interpolated onto the physical boundary $\partial\Omega$ since, in general, the computational boundary $\partial\tilde{\Omega}$ does not coincide with the physical boundary. In this work u_h is interpolated onto Gauss quadrature points on piecewise subsets of $\partial\Omega$ (pink square dots in Figure 1). The locations of these Gauss points are independent of $\partial\tilde{\Omega}$ and do not even require a watertight geometry; this suggests the proposed method will be able to handle CAD geometries with gaps. Based on numerical experiments, we suggest using a sufficient number of Gauss points to resolve the physical boundary and achieve an optimal convergence rate. In addition, the conditioning of the Karush-Kuhn-Tucker (KKT) system discussed in next subsection depends little on the number of points used to evaluate the misfit.

Finally, we note the inverse problem (2) is consistent with the PDE (1). As $h \to 0$, $\tilde{\Omega} \to \Omega$ and $\partial\tilde{\Omega} \to \partial\Omega$. Then the exact solution also solves the inverse problem (2), with $c_h \to u_D$ and $J_h \to 0$.

B. KKT system

A solution to (2) must satisfy the first optimality conditions, also known as the KKT conditions. In this subsection, we will show how the inverse problem (2) can be solved equivalently as a system of linear or nonlinear equations. For convenience, we introduce the following definitions for the derivative submatrices:

$$\mathsf{J}_u = \frac{\partial R_h}{\partial u_h}, \qquad \mathsf{J}_c = \frac{\partial R_h}{\partial c_h}, \tag{3}$$

$$\mathsf{H}_{uu} = \frac{\partial^2 L_h}{\partial u_h^2}, \qquad \mathsf{H}_{cc} = \frac{\partial^2 L_h}{\partial c_h^2}, \tag{4}$$

$$\mathsf{H}_{uc} = \frac{\partial^2 L_h}{\partial u_h \partial c_h}, \quad \mathsf{H}_{cu} = \frac{\partial^2 L_h}{\partial c_h \partial u_h}. \tag{5}$$

The matrix $\mathsf{H}_{uc} = \mathsf{H}_{cu}{}^T$ for problems with continuous second derivatives, but they are kept distinct in the following sections.

To begin, we define the Lagrangian

$$L_h(u_h, c_h, \psi_h) = J_h + \psi_h^T R_h, \tag{6}$$

where ψ_h is the Lagrange multiplier. By differentiating the Lagrangian with respect to u_h, c_h and ψ_h, and setting the derivatives to zero, we obtain the KKT conditions

$$\frac{\partial L_h}{\partial x} = \begin{bmatrix} \frac{\partial J_h}{\partial u_h} + \psi_h^T \mathsf{J}_u \\ \frac{\partial J_h}{\partial c_h} + \psi_h^T \mathsf{J}_c \\ R_h \end{bmatrix} = 0, \tag{7}$$

where we have introduced the compound vector

$$x = \begin{bmatrix} u_h^T & c_h^T & \psi_h^T \end{bmatrix}^T.$$

The conditions (7) define a system of equations that inherit the linearity/nonlinearity of the original problem (1). That is, if (1) is linear, so is (7); if (1) is nonlinear, (7) is also nonlinear. The first equation in (7) is typically called the adjoint equation, and the multiplier ψ_h is the adjoint. The second equation is the total derivative of the objective J_h with respect to c_h, and the third is the equality constraint in (2), i.e., the state discretization.

For the linear problems considered in this work, (7) is equivalent to the following KKT system:

$$\mathsf{K}\Delta x = -\frac{\partial L_h}{\partial x} \tag{8}$$

where

$$\mathsf{K} = \frac{\partial^2 L_h}{\partial x^2} = \begin{bmatrix} \mathsf{H}_{uu} & \mathsf{H}_{uc} & \mathsf{J}_u^T \\ \mathsf{H}_{cu} & \mathsf{H}_{cc} & \mathsf{J}_c^T \\ \mathsf{J}_u & \mathsf{J}_c & 0 \end{bmatrix}$$

is the second derivative of the Lagrangian, and $\Delta x = x^* - x$ with x^* being the solution to the KKT system.

There are two approaches that can be used to solve the KKT system (8): the reduced-space approach and the full-space approach. In the reduced-space approach, the state and adjoint variables are first eliminated to produce an equation for c_h. Once the control is known, R_h can be solved to find u_h. This whole process may need to be repeated for nonlinear problems until convergence. In the full-space approach the three variables are solved simultaneously. We use the full-space approach in this work, because, eventually, we are interested in solving nonlinear BVPs governed by, e.g., the Euler and Navier-Stokes equations, and solving (7) in the full space is usually more efficient for nonlinear problems. Readers are referred to [14–16] for further discussion on the relative merits of the two approaches.

III. Convergence Study

In this section we carry out a convergence study of our proposed approach using the method of manufactured solutions. The model BVP is a steady linear convection-diffusion equation. By adjusting the advection velocity and the diffusion coefficient pure convection and diffusion problems can be recovered as extreme cases. The discretization R_h is a DG finite-element method, which is described in greater detail below.

A. Model problem

As mentioned above, we consider the linear convection-diffusion equation:

$$\begin{aligned} \nabla \cdot (\boldsymbol{a}u - \mu \nabla u) &= f, && \text{in} \quad \Omega, \\ u &= u_D, && \text{on} \quad \partial\Omega, \end{aligned} \tag{9}$$

where $\boldsymbol{a} \in \mathbf{R}^2$ is the convection velocity, and μ is the positive diffusion coefficient. The problem domain Ω is the unit disk, i.e., $\Omega = \{[x, y] : x^2 + y^2 \leq 1\}$. In the following tests, three sets of parameters $\boldsymbol{a}$ and μ are chosen to model different physics:

- $\boldsymbol{a} = [1, 1]$, $\mu = 0$ for a pure convection problem;
- $\boldsymbol{a} = [0, 0]$, $\mu = 1$ for a pure diffusion problem;
- $\boldsymbol{a} = [1, 1]$, $\mu = 10^{-2}$ for a convection-diffusion problem.

The manufactured solution used to derive the source f and the boundary data u_D is defined to be

$$u = e^{x+y} \sin(\pi x) \sin(\pi y). \tag{10}$$

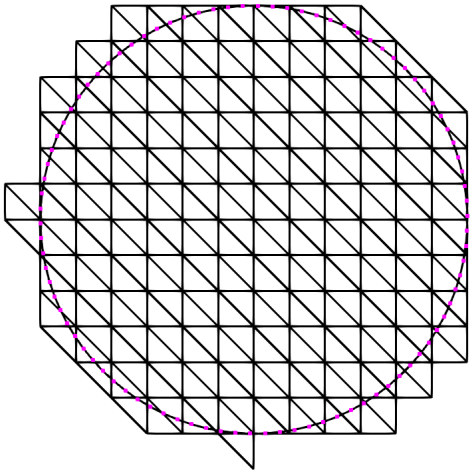


Fig. 2 The coarsest mesh and the physical boundary used for the convergence study.

B. Discontinuous Galerkin discretization

We use a DG finite-element method to discretize (9). The symmetric interior penalty Galerkin (SIPG) method[17] is used to discretize the diffusion term while upwinding is used for the advection part[18]. In order to be more specific, we begin by introducing some notation. Let $\tilde{\Omega}_h$ be a shape-regular subdivision of $\tilde{\Omega}$ into disjoint elements $K \in \tilde{\Omega}_h$, and let $\mathcal{V}_h$ be a broken function space on $\tilde{\Omega}_h$ such that $\mathcal{V}_h(K) \subset H^2$. The set of interior faces is denoted by Γ_I. Additionally, we introduce the standard jump and mean operators on both scalar and vector variables. For an interior face $e \in \Gamma_I$, these operators are given by

$$\begin{aligned} \{\{u\}\} &= (u^+ + u^-)/2, & \{\{\boldsymbol{q}\}\} &= (\boldsymbol{q}^+ + \boldsymbol{q}^-)/2, \\ [\![u]\!] &= u^+ \boldsymbol{n}^+ + u^- \boldsymbol{n}^-, & [\![\boldsymbol{q}]\!] &= \boldsymbol{q}^+ \cdot \boldsymbol{n}^+ + \boldsymbol{q}^- \cdot \boldsymbol{n}^-, \end{aligned}$$

where $\boldsymbol{n}^+$ and $\boldsymbol{n}^-$ are the outward unit normals of ∂K^+ and ∂K^-, respectively, and u^+ and u^- are the traces along the common face from the interior of K^+ and K^-, respectively. Finally, the subscript h will be used to indicate a function from a finite dimensional space; for example, the approximate solution to the PDE is denoted as u_h which is previously introduced in Section II.

The bilinear weak form of the discretization R_h corresponding to (9) reads

$$
\begin{aligned}
R_h(u_h, c_h; v_h) = & \int_{\tilde{\Omega}_h} [\nabla v_h \cdot (\boldsymbol{a} u_h - \mu \nabla u_h) + v_h f_h] \, \mathrm{d}\Omega \\
& - \int_{\Gamma_I} \left[[\![v_h]\!] \cdot \hat{F}(u_h^+, u_h^-) \right] \mathrm{d}\Gamma - \int_{\partial\tilde{\Omega}} \left[v_h \hat{F}(u_h, c_h) \cdot \boldsymbol{n} \right] \mathrm{d}\Gamma \\
& + \int_{\Gamma_I} [[\![v_h]\!] \cdot \{\!\{ \mu \nabla u_h \}\!\} + \{\!\{ \mu \nabla v_h \}\!\} \cdot [\![u_h]\!] - \epsilon \{\!\{ \mu \}\!\} [\![u_h]\!] \cdot [\![v_h]\!]] \, \mathrm{d}\Gamma \\
& + \int_{\partial\tilde{\Omega}} [v_h \mu \nabla u_h \cdot \boldsymbol{n} + (\mu \nabla v_h) \cdot \boldsymbol{n} (u_h - c_h) - \epsilon \mu v_h (u_h - c_h)] \, \mathrm{d}\Gamma,
\end{aligned}
\tag{11}
$$

where $v_h \in \mathcal{V}_h$ is the test function, $\hat{F}$ is the upwinding flux function, and ϵ is the SIPG penalty parameter[17]. Note that the discrete source f_h is obtained by projecting the exact source f derived from the manufactured solution (10) onto $\tilde{\Omega}$.

As mentioned in Section II.A, the control variable is the Dirichlet boundary value on $\partial\tilde{\Omega}$. These boundary conditions are weakly imposed by penalizing the discrepancy between u_h and c_h.

Lastly, the Lagrangian polynomials with degree $p \in \{1, 2, 3, 4\}$ are chosen as the basis of $\mathcal{V}_h$.

C. Numerical results

In the following test cases, the linear KKT system (8) is solved using a sparse direct solver. In Section IV we will discuss the iterative solution of (7).

To estimate the asymptotic convergence rate, we use a sequence of five uniformly refined triangular meshes. The coarsest mesh together with the immersed boundary $\partial\Omega$ is shown in Figure 2, with element size $h = H = 0.1178513$. Here element size h is taken to be the square root of the element area. Each element is subdivided into four to obtain a refined mesh; that is, the element sizes of the finer meshes are $H/2$, $H/4$, $H/8$ and $H/16$. We can see from Figure 2 that the physical boundary intersects mesh elements at different locations, including vertices, which will help demonstrate the robustness of the proposed approach.

We would like to assess the accuracy of the discrete solution u_h, and this is typically accomplished by evaluating the L^2 error on a sequence of ever finer grids; however, evaluating the L^2 solution error in Ω is not straightforward, because element-based cubature rules do not apply on the elements cut by the boundary. One could develop cubature rules for the elements cut by the immersed boundary, but this is a research topic in its own right[19]. The approach adopted in this paper is to set the solution error on $\tilde{\Omega}_h \setminus \Omega$ to zero.

The solution contours using $p = 1$ and $p = 4$ basis functions on the coarsest mesh are compared against the exact contours in Figure 3a and 3b, respectively. We can see that in both cases the discrete solution matches well with the exact manufactured solution. Furthermore, as expected, a higher-order approximation produces better results on the same mesh.

Figure 4 plots the solution error versus element size h for the specific convection, diffusion and convection-diffusion problems defined earlier. For all problems, our approach achieves the optimal convergence rates of $p + 1$.

IV. Iterative solution of the KKT system

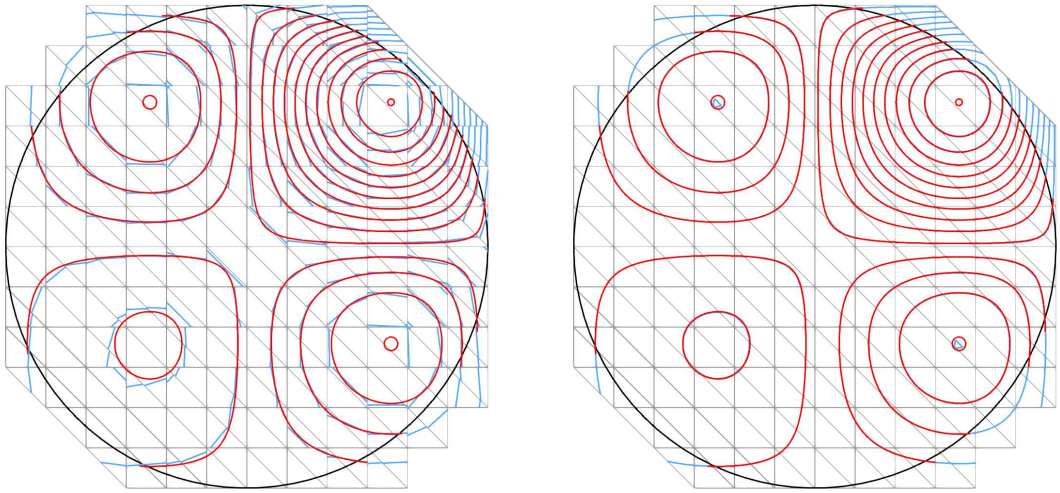
An alternative to the direct factorization used in Section III is an iterative method. Iterative methods are suitable for solving very large systems in terms of memory consumption, and they often lend themselves well to parallelization.

However, compared to a conforming-mesh discretization, the system (8) is an indefinite and highly ill-conditioned saddle-point problem. Therefore, an effective preconditioner is essential for an iterative solution of (8) to be practical. In [14] a set of preconditioners based on the block factorization of the KKT matrix were introduced, and we will apply two of them to our inverse problem.

A. Preconditioners for KKT matrix

An exact factorization of the KKT matrix K is given by

$$
\mathsf{K} = \begin{bmatrix} \mathsf{H}_{uu}\mathsf{J}_u^{-1} & 0 & \mathsf{I} \\ \mathsf{H}_{cu}\mathsf{J}_u^{-1} & \mathsf{I} & \mathsf{J}_c^T \mathsf{J}_u^{-T} \\ \mathsf{I} & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathsf{J}_u & \mathsf{J}_c & 0 \\ 0 & \mathsf{H}_z & 0 \\ 0 & \mathsf{H}_y^T & \mathsf{J}_u^T \end{bmatrix},
\tag{12}
$$



(a) $p = 1$ **(b)** $p = 4$

Fig. 3 Solution contour of the convection-diffusion problem on the coarsest mesh. Exact solution: red line, discrete solution: blue line.

where we have introduced H_y and the reduced Hessian H_z:

$$\begin{aligned} \mathsf{H}_y &= \mathsf{H}_{cu} - \mathsf{J}_c^T \mathsf{J}_u^{-T} \mathsf{H}_{uu}, \\ \mathsf{H}_z &= \mathsf{J}_c^T \mathsf{J}_u^{-T} \mathsf{H}_{uu} \mathsf{J}_u^{-1} \mathsf{J}_c - \mathsf{J}_c^T \mathsf{J}_u^{-T} \mathsf{H}_{uc} - \mathsf{H}_{cu} \mathsf{J}_u^{-1} \mathsf{J}_c + \mathsf{H}_{cc}. \end{aligned} \tag{13}$$

The factorization (12) is significant because it is equivalent (up to permutation) to a block LU factorization of K; thus, it permits a sequential solution of the three variables, c_h, u_h, and ψ_h. It suggests that we can build a preconditioner, denoted by P_1, for the KKT matrix by replacing the reduced Hessian H_z with an approximation B_z, and replacing the state Jacobian J_u with its own preconditioner $\tilde{\mathsf{J}}_u$

$$\mathsf{P}_1 = \begin{bmatrix} \mathsf{H}_{uu}\tilde{\mathsf{J}}_u^{-1} & 0 & \mathsf{I} \\ \mathsf{H}_{cu}\tilde{\mathsf{J}}_u^{-1} & \mathsf{I} & \mathsf{J}_c^T \tilde{\mathsf{J}}_u^{-T} \\ \mathsf{I} & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{\mathsf{J}}_u & \mathsf{J}_c & 0 \\ 0 & \mathsf{B}_z & 0 \\ 0 & \tilde{\mathsf{H}}_y^T & \tilde{\mathsf{J}}_u^T \end{bmatrix}, \tag{14}$$

where

$$\tilde{\mathsf{H}}_y = \mathsf{H}_{cu} - \mathsf{J}_c^T \tilde{\mathsf{J}}_u^{-T} \mathsf{H}_{uu}.$$

Furthermore, if all the second derivative matrices H_{uu}, H_{uc}, H_{cu} and H_{cc} are discarded, P_1 is reduced to another preconditioner P_2:

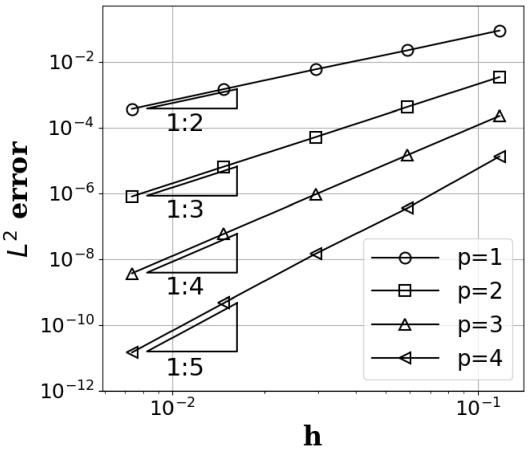
$$\mathsf{P}_2 = \begin{bmatrix} 0 & 0 & \mathsf{I} \\ 0 & \mathsf{I} & \mathsf{J}_c^T \tilde{\mathsf{J}}_u^{-T} \\ \mathsf{I} & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{\mathsf{J}}_u & \mathsf{J}_c & 0 \\ 0 & \tilde{\mathsf{H}}_z & 0 \\ 0 & 0 & \tilde{\mathsf{J}}_u^T \end{bmatrix}. \tag{15}$$

P_1 and P_2 correspond to $\tilde{\mathsf{P}}_4$ and $\tilde{\mathsf{P}}_2$ from [14], where it was shown that P_1 requires four applications of the state preconditioner (or its transpose) and P_2 requires two applications of the state preconditioner (or its transpose).

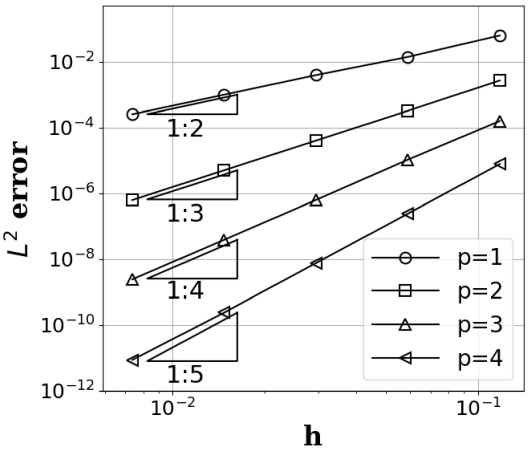
Once $\tilde{\mathsf{J}}_u$ and B_z are chosen, the preconditioner for the whole KKT system is fully determined. In our work, we use the Crout version of Incomplete LU factorization (ILUC)[20] as the preconditioner $\tilde{\mathsf{J}}_u$. The accuracy of the ILUC factorization is determined by a threshold parameter τ_{ILUC}; when $\tau_{\text{ILUC}} = 0$ ILUC is equivalent to a complete LU factorization.

Two choices of the approximate reduced Hessian, B_z, are considered. Both choices are based on an approximate reduced Hessian $\tilde{\mathsf{H}}_z$ rather than the exact one. $\tilde{\mathsf{H}}_z$ approximates H_z by replacing the state Jacobian J_u with its preconditioner $\tilde{\mathsf{J}}_u$; that is

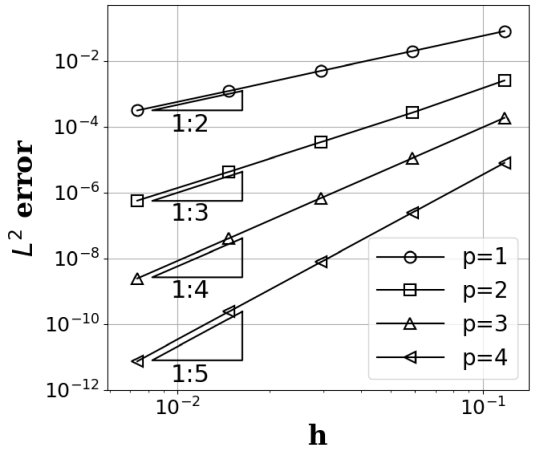
$$\tilde{\mathsf{H}}_z = \mathsf{J}_c^T \tilde{\mathsf{J}}_u^{-T} \mathsf{H}_{uu} \tilde{\mathsf{J}}_u^{-1} \mathsf{J}_c - \mathsf{J}_c^T \tilde{\mathsf{J}}_u^{-T} \mathsf{H}_{uc} - \mathsf{H}_{cu} \tilde{\mathsf{J}}_u^{-1} \mathsf{J}_c + \mathsf{H}_{cc}. \tag{16}$$



(a) convection problem



(b) diffusion problem



(c) convection-diffusion problem

Fig. 4 L^2 solution error versus element size h.

The first reduced Hessian preconditioner, denoted by $\mathsf{B}_z^{\text{ILUC}}$, is obtained by applying the ILUC factorization to (16). Since $\mathsf{B}_z^{\text{ILUC}}$ is based on an ILUC factorization, an explicit expression for $\tilde{\mathsf{H}}_z$ is required, which limits its application to small- or medium-size problems.

The second choice for B_z, denoted by B_z^{LC}, is a Lanczos-Chebyshev preconditioner (the two-step stationary method in [14]). This preconditioner applies a fixed number of Chebyshev iterations using (16) to precondition H_z. Chebyshev iteration is a stationary method suitable for positive definite matrices and requires estimates of the smallest and the largest eigenvalues. Since the only operations required is are Hessian-vector products, it is matrix free*. See [21] for more information on Chebyshev iteration and the Lanczos algorithm.

In the following results, when B_z^{LC} is used we denote P_1 and P_2 as $\mathsf{P}_1(\mathsf{B}_z^{\text{LC}})$ and $\mathsf{P}_2(\mathsf{B}_z^{\text{LC}})$, respectively. The same convention also applies to $\mathsf{B}_z^{\text{ILUC}}$.

B. Numerical results

In this subsection, the convection-diffusion problem described in Section III is solved with a restarted left-preconditioned Generalized Minimal Residual (GMRES) method[22] in order to assess the performance of the preconditioners P_1 and P_2. The ILUC factorization with $\tau_{\text{ILUC}} = 10^{-4}$ is used as the state preconditioner $\tilde{\mathsf{J}}_u$. For $\mathsf{B}_z^{\text{ILUC}}$, the threshold of the ILUC factorization is $\tau_{\text{ILUC}} = 10^{-8}$. All cases use $p = 4$ Lagrange basis functions. The number of GMRES restart steps and the number of Lanczos iterations are both set to 100, and the number of Chebyshev iterations is fixed at 20. The GMRES iteration terminates once the residual norm is reduced by a factor of 10^{13}.

The convergence histories in terms of GMRES iterations are plotted in Figure 5. As can be seen, the Krylov iteration converges in fewer than 200 steps using preconditioner P_1, for all the meshes, and in fewer than 900 steps using P_2. Additionally, with the same B_z, it takes many fewer iterations (around a third) with P_1 than with P_2. Even taking into account that each application of P_1 requires twice the number of applications of $\tilde{\mathsf{J}}_u$ as P_1, this suggests that P_2 is more efficient than P_2, at least for problems considered.

A comparison between the reduced Hessian preconditioners $\mathsf{B}_z^{\text{ILUC}}$ and B_z^{LC} shows that with the current settings the former is superior to the latter in terms of the number of Krylov iteration. However, to construct $\mathsf{B}_z^{\text{ILUC}}$ the explicit expression of the approximate reduced Hessian $\tilde{\mathsf{H}}_z$ has to be available, which needs at least $size(c_h)$ applications of $\tilde{\mathsf{J}}_u$. On the other hand, the cost for B_z^{LC} is composed of two parts: the Lanczos iteration used to estimate the smallest and the largest eigenvalues of $\tilde{\mathsf{H}}_z$, which has to be performed once per Krylov solve, and the Chebyshev iteration for each B_z^{LC} application.

Although Figure 5 shows the number of iterations increases as the problem size gets larger, this does not imply that P_1 and P_2 themselves are not effective, because their performance depends strongly on the performance of the state preconditioner $\tilde{\mathsf{J}}_u$. For instance, we have observed that a fixed threshold value for the ILUC factorization works better for small problems than for large problems. For example, with the threshold value mentioned above, the ratio between the number of nonzero values in $\tilde{\mathsf{J}}_u^{-1}$ and that in J_u, $nnz(\tilde{\mathsf{J}}_u^{-1})/nnz(\mathsf{J}_u)$, drops from 67.85% on the coarsest mesh to 56.13% on the finest mesh; keep in mind a larger ratio indicates fewer dropped entries and a more accurate approximation.

To reduce the influence of $\tilde{\mathsf{J}}_u$ on our assessment of the preconditioners, we examine the number of GMRES iterations required to solve the preconditioned adjoint equation (the first equation in (7))

$$\tilde{\mathsf{J}}_u^{-T}\mathsf{J}_u^T\psi_h = -\frac{\partial J_h}{\partial u_h},$$

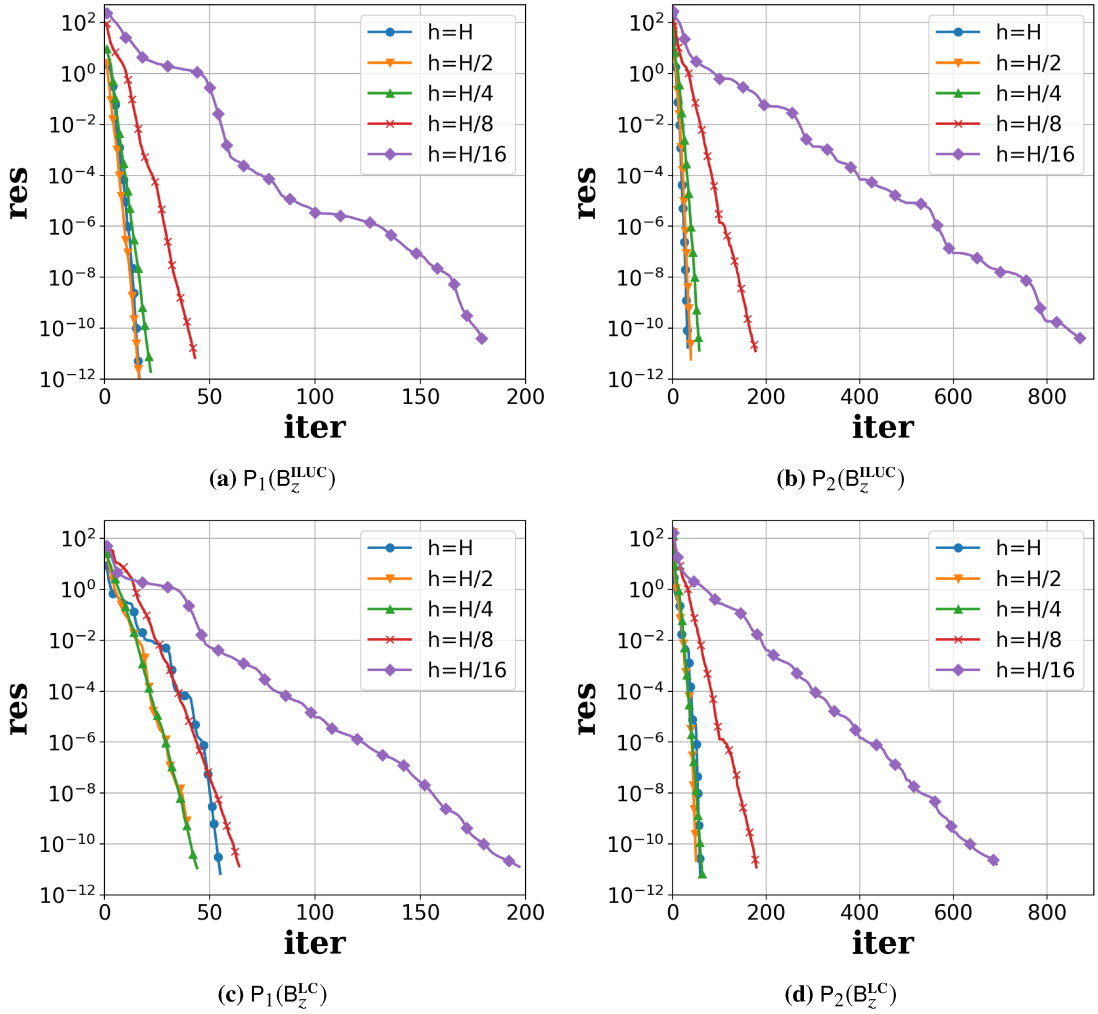
and use this to normalize the number of GMRES iterations required to solve the KKT system. Note that this does not reflect the cost of constructing or using the B_z preconditioners.

The results are given in Table 1.We find that $\mathsf{P}_1(\mathsf{B}_z^{\text{LC}})$ is the most scalable preconditioner. Furthermore, roughly speaking, P_1 is more scalable than P_2 for both $\mathsf{B}_z^{\text{ILUC}}$ and B_z^{LC}.

V. Conclusion

We have proposed a novel immersed boundary method that is formulated as an inverse problem. The method is notable in the sense that it is agnostic to the underlying discretization and it is high-order accurate. We verified this latter characteristic by demonstrating that a DG discretization achieves optimal convergence rates on advection, advection-diffusion, and pure diffusion problems. In order for the method to be applied to large-scale problems, iterative methods are necessary for which we need effective preconditioners. We presented potential preconditioners based

*More precisely, it is Hessian free, since the ILUC factorization of J_u is still needed for (16).



(a) $\mathrm{P}_1(\mathrm{B}_z^{\mathrm{ILUC}})$ **(b)** $\mathrm{P}_2(\mathrm{B}_z^{\mathrm{ILUC}})$

(c) $\mathrm{P}_1(\mathrm{B}_z^{\mathrm{LC}})$ **(d)** $\mathrm{P}_2(\mathrm{B}_z^{\mathrm{LC}})$

Fig. 5 Convergence histories for the various preconditioners when used to solve the convection-diffusion problem.

Table 1 Relative performance of KKT preconditioners with respect to the state preconditioner $\tilde{J}_u$ with approximation degree $p = 4$.

mesh size h	$P_1(B_z^{ILUC})$	$P_2(B_z^{ILUC})$	$P_1(B_z^{LC})$	$P_2(B_z^{LC})$
H	2.13	4.00	6.88	7.38
$H/2$	1.54	3.54	3.63	4.54
$H/4$	1.57	4.07	3.14	4.50
$H/8$	0.86	3.54	1.28	3.58
$H/16$	3.38	16.50	3.72	13.06

on using the PDE Jacobian to eliminate the state and adjoint. These preconditioners work relatively well when the effect of the state preconditioner $\tilde{J}_u$ is taken into account. Future work will focus on improving the efficiency of the preconditioner for the reduced Hessian and extending the methodology to the Euler and Navier-Stokes equations.

Acknowledgements

The authors gratefully acknowledge the financial support of Rensselaer Polytechnic Institute and RPI's Scientific Computation Research Center for the use of computer facilities.

References

[1] Slotnick, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., and Mavriplis, D., "CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences," Tech. Rep. NASA/CR-2014-218178, Langley Research Center, 2014.

[2] Peskin, C. S., "Flow patterns around heart valves: A numerical method," *Journal of Computational Physics*, Vol. 10, No. 2, 1972, pp. 252 – 271. doi:https://doi.org/10.1016/0021-9991(72)90065-4, URL http://www.sciencedirect.com/science/article/pii/0021999172900654.

[3] Lew, A. J., and Buscaglia, G. C., "A discontinuous-Galerkin-based immersed boundary method," *International Journal for Numerical Methods in Engineering*, Vol. 76, No. 4, 2008, pp. 427–454. doi:10.1002/nme.2312, URL http://dx.doi.org/10.1002/nme.2312.

[4] Tornberg, A.-K., and Engquist, B., "Regularization Techniques for Numerical Approximation of PDEs with Singularities," *Journal of Scientific Computing*, Vol. 19, No. 1, 2003, pp. 527–552. doi:10.1023/A:1025332815267, URL https://doi.org/10.1023/A:1025332815267.

[5] Lai, M.-C., and Peskin, C. S., "An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity," *Journal of Computational Physics*, Vol. 160, No. 2, 2000, pp. 705 – 719. doi:https://doi.org/10.1006/jcph.2000.6483, URL http://www.sciencedirect.com/science/article/pii/S0021999100964830.

[6] Smereka, P., "The numerical approximation of a delta function with application to level set methods," *Journal of Computational Physics*, Vol. 211, No. 1, 2006, pp. 77 – 90. doi:https://doi.org/10.1016/j.jcp.2005.05.005, URL http://www.sciencedirect.com/science/article/pii/S0021999105002627.

[7] LeVeque, R. J., and Li, Z., "The Immersed Interface Method for Elliptic Equations with Discontinuous Coefficients and Singular Sources," *SIAM Journal on Numerical Analysis*, Vol. 31, No. 4, 1994, pp. 1019–1044. doi:10.1137/0731054, URL https://doi.org/10.1137/0731054.

[8] Linnick, M. N., and Fasel, H. F., "A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains," *Journal of Computational Physics*, Vol. 204, No. 1, 2005, pp. 157 – 192. doi:https://doi.org/10.1016/j.jcp.2004.09.017, URL http://www.sciencedirect.com/science/article/pii/S0021999104004127.

[9] Brehm, C., and Fasel, H., "A novel concept for the design of immersed interface methods," *Journal of Computational Physics*, Vol. 242, No. Supplement C, 2013, pp. 234 – 267. doi:https://doi.org/10.1016/j.jcp.2013.01.027, URL http://www.sciencedirect.com/science/article/pii/S0021999113000715.

[10] Mattsson, K., and Almquist, M., "A high-order accurate embedded boundary method for first order hyperbolic equations," *Journal of Computational Physics*, Vol. 334, 2017, pp. 255 – 279. doi:https://doi.org/10.1016/j.jcp.2016.12.034, URL http://www.sciencedirect.com/science/article/pii/S0021999116306969.

[11] Barbosa, H. J., and Hughes, T. J., "The finite element method with Lagrange multipliers on the boundary: circumventing the Babuška-Brezzi condition," *Computer Methods in Applied Mechanics and Engineering*, Vol. 85, No. 1, 1991, pp. 109 – 128. doi:https://doi.org/10.1016/0045-7825(91)90125-P, URL http://www.sciencedirect.com/science/article/pii/004578259190125P.

[12] Moës, N., Béchet, E., and Tourbier, M., "Imposing Dirichlet boundary conditions in the extended finite element method," *International Journal for Numerical Methods in Engineering*, Vol. 67, No. 12, 2006, pp. 1641–1669. doi:10.1002/nme.1675, URL http://dx.doi.org/10.1002/nme.1675.

[13] Stenberg, R., "On some techniques for approximating boundary conditions in the finite element method," *Journal of Computational and Applied Mathematics*, Vol. 63, No. 1, 1995, pp. 139 – 148. doi:https://doi.org/10.1016/0377-0427(95)00057-7, URL http://www.sciencedirect.com/science/article/pii/0377042795000577, proceedings of the International Symposium on Mathematical Modelling and Computational Methods Modelling 94.

[14] Biros, G., and Ghattas, O., "Parallel Lagrange–Newton–Krylov–Schur Methods for PDE-Constrained Optimization. Part I: The Krylov–Schur Solver," *SIAM Journal on Scientific Computing*, Vol. 27, No. 2, 2005, pp. 687–713. doi:10.1137/S106482750241565X, URL https://doi.org/10.1137/S106482750241565X.

[15] Borzì, A., and Schulz, V., *Computational Optimization of Systems Governed by Partial Differential Equations*, Society for Industrial and Applied Mathematics, 2011. doi:10.1137/1.9781611972054, URL http://dx.doi.org/10.1137/1.9781611972054.

[16] Akçelik, V., Biros, G., Ghattas, O., Hill, J., Keyes, D., and van Bloemen Waanders, B., "Parallel Algorithms for PDE-Constrained Optimization," *Parallel Processing for Scientific Computing*, edited by M. A. Heroux, P. Raghavan, and H. D. Simon, Society for Industrial and Applied Mathematics, 2006, Chap. 16, pp. 291–322. doi:10.1137/1.9780898718133.ch16, URL http://dx.doi.org/10.1137/1.9780898718133.ch16.

[17] Shahbazi, K., "An explicit expression for the penalty parameter of the interior penalty method," *Journal of Computational Physics*, Vol. 205, No. 2, 2005, pp. 401–407. doi:10.1016/j.jcp.2004.11.017, URL http://dx.doi.org/10.1016/j.jcp.2004.11.017.

[18] Houston, P., Schwab, C., and Süli, E., "Discontinuous hp-Finite Element Methods for Advection-Diffusion-Reaction Problems," *SIAM Journal on Numerical Analysis*, Vol. 39, No. 6, 2002, pp. 2133–2163. doi:10.1137/S0036142900374111, URL https://doi.org/10.1137/S0036142900374111.

[19] Fidkowski, K. J., and Darmofal, D. L., "A triangular cut-cell adaptive method for high-order discretizations of the compressible Navier–Stokes equations," *Journal of Computational Physics*, Vol. 225, No. 2, 2007, pp. 1653 – 1672. doi:https://doi.org/10.1016/j.jcp.2007.02.007, URL http://www.sciencedirect.com/science/article/pii/S0021999107000757.

[20] Li, N., Saad, Y., and Chow, E., "Crout Versions of ILU for General Sparse Matrices," *SIAM Journal on Scientific Computing*, Vol. 25, No. 2, 2003, pp. 716–728. doi:10.1137/S1064827502405094, URL https://doi.org/10.1137/S1064827502405094.

[21] Barrett, R., Berry, M., Chan, T. F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., and der Vorst, H. V., *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*, SIAM, Philadelphia, PA, 1994.

[22] Saad, Y., and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal on Scientific and Statistical Computing*, Vol. 7, No. 3, 1986, pp. 856–869. doi:10.1137/0907058, URL https://doi.org/10.1137/0907058.