Nonlinear Programming and Domain Decomposition for Partial Differential Equations

John E. Dennis, Jr.
Robert Michael Lewis

CRPC-TR92252
September 1992

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

Nonlinear programming and domain decomposition for partial differential equations

John E. Dennis, Jr.
Robert Michael Lewis
Dept. of Mathematical Sciences
Rice University
Houston, Texas

We will discuss several approaches we have developed for integrating domain decomposition methods for partial differential equations into formulations for nonlinear programming (NLP) algorithms for solving optimization problems governed by PDE. The methods we have developed seek to exploit computational parallelism in the solution of PDE; however, our optimization methods are not simply parallel implementations of existing approaches. Instead, our NLP methods take advantage of domain decomposition for PDE to produce more efficient and robust methods for the optimization problems we seek to solve.

There is no question that we will need to exploit computational parallelism in such large-scale problems as optimal control, optimal design, and nondestructive evaluation for systems governed by partial differential equations. The dominant cost in such optimization problems is the repeated simulations of a physical system for new values of control or design parameters. In developing nonlinear programming methods for systems governed by PDE, we have concentrated on exploiting computational parallelism in the solution of the attendant differential equations. The methods we are investigating seek to reduce the effort expended on the solution of PDEs.

In the simplest instance, the problems we consider can be formulated as large-scale nonlinear programming problems of the following form:

$$\text{minimize} \quad f(x, y(x)) \tag{1}$$

where x represents the control variables and $y(x)$ the state variables. The control and state variables are related by solving some relation

$$h(x, y(x)) = 0 \tag{2}$$

for $y(x)$. In our work on such optimization problem we choose to view the state equations as constituting equality constraints for the optimization problem. The preceding problem we would thus formulate as

$$\begin{array}{ll} \text{minimize} & f(x, y) \\ \text{subject to} & h(x, y) = 0. \end{array} \tag{3}$$

This formulation is motivated by the opinion, widely held in the optimization community, that if one has equality constraints in an optimization problem, it is generally inefficient to maintain feasibility with respect to these constraints at every step of an optimization algorithm; one really need only insure that feasibility will be attained at the same time as constrained optimality. Yet, such feasibility would be enforced in the formulation given by (1) and (2). If we no longer demand feasibility at every iteration of the NLP algorithm, we are led to (3), in which we expand the parameter space to (x, y) and remove the additional

degrees of freedom at the solution via the constraints. However, the equality-constrained optimization algorithm we use allows iterates (x, y) that are not necessarily feasible with respect to the constraints. This means that we can take advantage of the additional degrees of freedom to move more quickly to a solution since we are not required to move along the manifold defined by the constraints.

We can treat these equality constraints relating the control variables to the state variables in various ways. For instance, we can maintain feasibility with respect to these constraints in every optimization iteration, which corresponds to what we call the "black-box" approach (1)–(2). At the other extreme, we have what we call the "all-at-once" approach, in which *all* the state equations are treated as equality constraints. A compromise between these two extremes leads to what we call the "in-between" approach, in which we maintain feasibility with respect to some of the state constraints while allowing infeasibility with respect to others.

These ideas are best explained by an example that we will discuss at length in our paper. Our model problem will be a simple instance of nondestructive evaluation, in which we seek to identify the coefficient in a second-order elliptic boundary-value problem (BVP). Let Ω be some smoothly bounded domain, and consider the following BVP defined on Ω:

$$\begin{aligned} -\nabla \cdot (K \nabla p) &= q \quad \text{on } \Omega \\ p &= g \qquad \text{on } \partial\Omega. \end{aligned} \tag{4}$$

The inverse problem we will consider is the following: suppose that S is some subset of Ω and that we know $p_{data} = p|_S$. Can we then identify the coefficient $K(x, y, z)$?

This problem is ill-posed, in the sense that K will not be well-determined on all of Ω by data given on only part of Ω. However, we will ignore this fact and the need for regularization for the purposes of our exposition.

We can formulate the problem of identifying K as a least-squares problem: For some choice of inner-product norm $||\cdot||$, we wish to solve

$$\text{minimize} \quad \left\| p[K]\big|_S - p_{data} \right\|^2 .$$

This formulation corresponds to the "black-box" approach to identifying $K(x, y, z)$. The "black-box" approach is characterized by the retention of the implicit relation between the state variable p and control variable K in the NLP—in order to compute the least-squares residual $p[K] - p_{data}$, we need to solve (4). Similarly, we need to solve the following auxiliary BVP in order to compute the gradient of the least-squares objective function:

$$\begin{aligned} -\nabla \cdot (K \nabla w) &= p - p_{data} \qquad \text{on } \Omega \\ w &= 0 \qquad\qquad\qquad\quad \text{on } \partial\Omega. \end{aligned}$$

We must solve yet more BVP to obtain other information needed for the optimization algorithm. The point is one we mentioned earlier; the cost of repeated simulations—BVP solutions, in this case—becomes dominant in the optimization algorithm.

Faced with the expense of such computations, we might naturally appeal to a domain decomposition method for handling the BVP. For this model problem, we choose a non-overlapping decomposition method devised by Glowinski and Wheeler [1], [2]. The idea of

this method is to subdivide the domain into smaller subdomains, add additional boundary values at the subdomain interfaces introduced by the decomposition, solve the resulting BVP on the subdomains, and then iteratively adjust the boundary values on the subdomain interfaces until fluxes between the subdomains match. To express this more precisely, we will assume for simplicity that Ω is subdivided into only two subdomains Ω_1 and Ω_2. We then solve the following problem. Choose Dirichlet data π for the boundary between Ω_1 and Ω_2, and solve, for $i = 1, 2$,

$$\begin{aligned} -\nabla \cdot (K \nabla p_i) &= q \quad \text{on } \Omega_i \\ p_i &= g \quad \text{on } \partial\Omega_i \cap \partial\Omega \\ p_i &= \pi \quad \text{on } \partial\Omega_i \setminus \partial\Omega. \end{aligned}$$

The game then becomes to choose π in such a way that the jump in the fluxes between Ω_1 and Ω_2 is zero; on $\partial\Omega_1 \cap \partial\Omega_2$, we want

$$[(K \nabla p) \cdot \nu] \equiv (K \nabla p_1) \cdot \nu_1 + (K \nabla p_1) \cdot \nu_1 = 0, \tag{5}$$

where ν_i is the outward pointing normal on $\partial\Omega_i$. This flux matching condition can be enforced by solving an auxiliary linear system that is symmetric and positive definite.

From an optimization perspective, enforcing the flux matching condition in the solution of the BVP represents enforcing feasibility with respect to a constraint that expresses the consistency of the subdomain solutions. As we previously mentioned, this is not a good idea from the standpoint of optimization. The idea of the "in-between" approach is to make explicit in the nonlinear programming formulation the flux-matching constraint that is implicit to the domain decomposition method.

In the "in-between" approach, we make this implicit flux matching constraint explicit and add it as a constraint to the nonlinear programming problem. Since our optimization algorithm allows iterates which may be infeasible with respect to this constraint, we are not concentrating our efforts on solving the BVP that represents the physical system. Instead, the progress of the optimization drives the accuracy with which we solve the state equations. We solve the PDE accurately—which is equivalent to feasibility with respect to the constraints—only as we achieve optimality.

Making the flux matching condition explicit in the NLP results in the following constrained least-squares formulation:

$$\begin{aligned} \text{minimize} \quad & \left\| p[K, \pi]\big|_S - p_{data} \right\|^2 \\ \text{subject to} \quad & [(K \nabla p) \cdot \nu] = 0, \end{aligned}$$

where $p[K, \pi]$ is given by the solving on each subdomain Ω_i the BVP.

$$\begin{aligned} -\nabla \cdot (K \nabla p_i) &= q \quad \text{on } \Omega_i \\ p_i &= g \quad \text{on } \partial\Omega_i \cap \partial\Omega \\ p_i &= \pi \quad \text{on } \partial\Omega_i \setminus \partial\Omega. \end{aligned}$$

We have expanded the parameter space now from K to (K, π); the constraint removes the additional degrees of freedom. However, since our equality constrained optimization

algorithm allows iterates to be infeasible, we can take advantage of the additional degrees of freedom in order to make more rapid progress towards solution of this problem.

We can go further and formulate the "all-at-once" method for the domain decomposition formulation of the BVP; in this approach we treat the subdomain BVP as constraints as well:

$$\begin{array}{lll} \text{minimize} & \left\| p\big|_S - p_{data} \right\|^2 & \\ \text{subject to} & [(K\nabla p)\cdot\nu] = 0 & \text{across subdomain interfaces} \\ & -\nabla\cdot(K\nabla p_i) = q & \text{on } \Omega_i \\ & p_i = g & \text{on } \partial\Omega_i \cap \partial\Omega_i \\ & p_i = \pi & \text{on } \partial\Omega_i \setminus \partial\Omega_i. \end{array}$$

In this case the control variables are now (p, π, K).

The "in-between" and "all-at-once" methods each have advantages. The "in-between" method has fewer constraints than the "all-at-once" method, so its implementation is somewhat simpler. On the other hand, the "in-between" method retains an implicit relation between K and p, which complicates the computation of derivatives with respect to K. We anticipate that further numerical tests will help better clarify the advantages and disadvantages of these two methods.

These ideas are applicable to other optimization problems and other domain decomposition techniques. The key is the observation that in a typical domain decomposition method, the region over which a PDE is to be solved is divided into smaller regions, and the PDE is then solved on each of the smaller regions in parallel. Some manner of correction is made and the subdomain solutions are continued until the solutions on the collection of subdomains represents the solution on the original domain. The consistency condition that one attempts to enforce can be lifted into the formulation of the optimization problem.

We will describe in greater detail how to apply these ideas to the model problem presented here and to other optimization problems involving PDE and domain decomposition, and how our domain decomposition formulations lead to computationally and analytically more tractable problems. We will also describe the trust-region algorithm for equality constrained optimization that we use and discuss how its design is affected by the domain decomposition approach. Finally, we will give results of numerical tests and comparisons of our Intel Gamma and Delta implementations of the "black-box", "in-between" and "all-at-once" methods for our model problem.

References

[1] L. C. COWSAR AND M. F. WHEELER, *Parallel domain decomposition method for mixed finite elements for elliptic partial differential equations*, in Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations, R. Glowinski, Y. A. Kuznetsov, G. Meurant, J. Périaux, O. B. Widlund, eds., Philadelphia, 1991, SIAM.

[2] R. GLOWINSKI AND M. F. WHEELER, *Domain decomposition and mixed finite element methods for elliptic problems*, in First International Symposium on Domain Decompo-

sition Methods for Partial Differential Equations, R. Glowinski, G. Golub, G. Meurant, and J. Periaux, eds., Philadelphia, 1988, SIAM.