SIAM J. SCI. COMPUT.
Vol. 37, No. 5, pp. S177–S194

PRECONDITIONING TECHNIQUES FOR REDUCED BASIS METHODS FOR PARAMETERIZED ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS*

HOWARD C. ELMAN† AND VIRGINIA FORSTALL‡

Abstract. The reduced basis methodology is an efficient approach to solve parameterized discrete partial differential equations when the solution is needed at many parameter values. An offline step approximates the solution space, and an online step utilizes this approximation, the reduced basis, to solve a smaller reduced problem, which provides an accurate estimate of the solution. Traditionally, the reduced problem is solved using direct methods. However, the size of the reduced system needed to produce solutions of a given accuracy depends on the characteristics of the problem, and it may happen that the size is significantly smaller than that of the original discrete problem but large enough to make direct solution costly. In this scenario, it may be more effective to use iterative methods to solve the reduced problem. We construct preconditioners for reduced iterative methods which are derived from preconditioners for the full problem. This approach permits reduced basis methods to be practical for larger bases than direct methods allow. We illustrate the effectiveness of iterative methods for solving reduced problems by considering two examples, the steady-state diffusion and convection-diffusion-reaction equations.

Key words. reduced basis, iterative methods, preconditioning

AMS subject classifications. 65N22, 65F08

DOI. 10.1137/140970859

1. Introduction.

This study is concerned with efficient iterative algorithms for the numerical solution of parameterized partial differential equations (PDEs). Examples where such equations arise include models of diffusion or reactions where the coefficients of diffusion or reaction are parameterized in some way, or they are uncertain. In settings of this type, we may require the computation of discrete solutions for many values of the input parameter, for example, to perform sensitivity analysis, design optimization, or statistical analysis of random processes. When an accurate spatial discretization is required (using, for example, finite element or finite difference discretization) this can be a prohibitively expensive task.

One approach for addressing this difficulty is to use reduced-order models. Costs are reduced by approximating the parameterized problem using a reduced space of significantly smaller dimension than that of the discrete PDE. We will do this using the *reduced basis method* [4, 5, 12]. Let the PDE

$$L(\vec{x}, \xi; u) = f(\vec{x}) \tag{1.1}$$

be defined on a spatial domain D and subject to boundary conditions on ∂D,

$$B(\vec{x}, \xi; u) = g(\vec{x}) \ , \tag{1.2}$$

*Received by the editors May 29, 2014; accepted for publication (in revised form) March 2, 2015; published electronically October 29, 2015. This work was supported by the U.S. Department of Energy Office of Advanced Scientific Computing Research, Applied Mathematics program under award DE-SC0009301 and by the U.S. National Science Foundation under grants DMS1115317 and DMS1418754.

http://www.siam.org/journals/sisc/37-5/97085.html

†Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (elman@cs.umd.edu).

‡Applied Mathematics & Statistics and Scientific Computation Program, University of Maryland, College Park, MD 20742 (vhfors@math.umd.edu).

where $\xi = [\xi_1, \xi_2, \ldots, \xi_m]^T$ is a vector of input parameters. Reduced basis methods compute a (relatively) small number of solutions, $u(\cdot, \xi^{(1)}), \ldots, u(\cdot, \xi^{(k)})$, known as snapshots, and then for other parameters, $\xi \neq \xi^{(j)}$, attempt to find $u(\cdot, \xi)$ in the space spanned by $\{u(\cdot, \xi^{(j)})\}_{j=1}^k$. In this paradigm, the computations are divided into so-called *offline* and *online* steps. The offline step, which may be expensive, computes the snapshots and builds a basis of the low-dimensional vector space spanned by them. The online step, which is intended to be inexpensive (because k is small), computes a projected version of the original problem (using, for example, a Galerkin projection) in the k-dimensional space. The projected problem, known as the *reduced model*, has a solution $\tilde{u}(\vec{x}, \xi)$ which is an approximation of the solution $u(\vec{x}, \xi)$.

An implicit assumption associated with this approach is that the costs of solving any linear systems that arise from the reduced model are low. The conventional wisdom is that these systems can be solved cheaply using direct methods, at costs lower than what would be needed to solve the original discrete PDE. This is reasonable when k, the size of the reduced basis, is significantly smaller than N, the size of the discrete space. However, when efficient ($O(N)$) algorithms, such as multigrid, are available for the discrete PDE, it may happen that k is smaller than N by a large amount, but direct methods (of complexity $O(k^3)$) do not lead to reduced costs.

In this study we address this issue by considering methods for the efficient solution of reduced problems when the reduced space is of moderate size. We consider iterative methods for this scenario, and, in particular, we develop preconditioning strategies for the reduced problem derived from a preconditioner for the underlying discrete PDE. We show that the construction of the preconditioner can be included in the offline portion of the computation, and that for problems depending on moderately large numbers of parameters, the preconditioned iterative solvers are more efficient than direct methods for solving the reduced algebraic systems.

An outline of the paper is as follows. In section 2, we review the reduced basis methodology for linear partial differential operators with affine dependence on parameters. In section 3, we discuss iterative methods for the solution of larger reduced problems and develop the preconditioning strategy we use with such methods. In section 4, we demonstrate the effectiveness of these techniques for solving two benchmark problems, the steady-state diffusion and convection-diffusion-reaction equations, and in section 5, we draw some conclusions.

2. Reduced basis method. In a finite element setting, we seek a discrete solution u_h of the PDE in a finite-dimensional affine space X^h such that

$$\mathcal{L}(u_h(\cdot, \xi), v_h) = l(v_h) \ \ \forall v_h \in X_0^h \ . \tag{2.1}$$

For simplicity, we consider Dirichlet problems, and X_0^h is the subset of X^h corresponding to homogeneous Dirichlet boundary conditions. Given a basis $Q = \{q_j\}_{j=1}^k$ such that $q_j \in X_0^h$, we solve the reduced model

$$\mathcal{L}(\tilde{u}_0(\cdot, \xi), v_h) = l(v_h) \ \ \forall v_h \in \text{span}(Q) \tag{2.2}$$

for $\tilde{u}_0 \in \text{span}(Q)$, which is used to construct an approximation of the full solution, $\tilde{u} = \tilde{u}_0 + u_{bc}$, where u_{bc} is the solution on the boundary. The accuracy of this approximation depends on how well the reduced basis represents the solution space. Thus, constructing this basis requires balancing two conflicting requirements: its rank, k, should be small enough so there is a benefit with respect to efficiency from using the reduced model, but k should also be large enough to ensure accuracy of the approximation.

In the offline-online paradigm, the offline step focuses on the construction of the reduced basis. A variety of approaches have been considered. For example, a proper orthogonal decomposition derived from solutions obtained for a subset of the parameter space produces such a basis [2]. Alternatively, the solutions to the full model at k samples of ξ produce $\{u(\cdot, \xi^{(j)})\}_{j=1}^{k}$, the snapshots, from which a stable basis can be formed by finding an orthogonal basis for the span of the snapshots constructed using, for example, the modified Gram–Schmidt algorithm. There are various methods for snapshot selection, including greedy sampling [5], error minimization methods [7], sparse grids [12], and an "hp" approach [11], which uses a refinement procedure to construct separate bases for subdomains of the parameter space. This (offline) portion of the computation may be expensive.

We will also assume that the operators L and B in (1.1) and (1.2) are affinely dependent on functions of ξ, i.e., for L,

$$L(\vec{x}, \xi; u) = \sum_{i=1}^{s_L} \psi_i(\xi)\, l_i(\vec{x}; u) \ , \tag{2.3}$$

where $\{l_i(\vec{x}; u)\}_{i=1}^{s_L}$ are parameter-independent operators and $\psi_i : \mathbb{R}^m \to \mathbb{R}$. This assumption leads to efficiencies for linear operators as well as mildly nonlinear (say, quadratic) ones because part of the reduced model can be precomputed as part of the offline step and the cost of solving the reduced model does not depend on the size of the full model. For example, for a linear PDE the solution of the full model in (2.1) is obtained by solving a matrix equation of the form

$$A(\xi) u_\xi = f \ , \tag{2.4}$$

where the order of the system, N, depends on the number of points in the spatial discretization of D and is assumed to be large. Let Q be an $N \times k$ orthogonal matrix whose columns span the same space as that determined by the coefficient vectors of the set of snapshots. The Galerkin projection of the reduced model of order k is

$$Q^T A(\xi) Q u_{r,\xi} = Q^T f \ , \tag{2.5}$$

where $\tilde{u}_\xi = Q u_{r,\xi}$ is the approximation of the solution of (2.4) on the interior of D. Because of the assumption of affine dependence, the coefficient matrix has the structure

$$A(\xi) = \sum_{i=1}^{s_L+s_B} \psi_i(\xi) A_i \ , \tag{2.6}$$

and the reduced model can be written as

$$\sum_{i=1}^{s_L+s_B} \psi_i(\xi) [Q^T A_i Q] u_{r,\xi} = Q^T f \ . \tag{2.7}$$

In this form, the matrices $\{Q^T A_i Q\}$ are parameter independent and thus can be precomputed as part of the offline step. The online step of the reduced model includes the assembly of the sum in (2.7). The cost of this computation is of order $(s_L + s_B)k^2$, and the total online cost is this plus the cost of solving an order k linear system. Hence, the cost of the reduced model is independent of N, the size of the full model. For reduced basis methods applied to problems with nonaffine dependence on parameters, see [1, 5, 8, 20]. For comparison of reduced basis methods with stochastic collocation methods, see [9].

3. Iterative methods for the reduced model. The conventional point of view is that the reduced model will be significantly less expensive to solve than the full model. The traditional choice for solving the reduced model in (2.5) is direct methods, at a computational cost of $O(k^3)$. On the other hand, it is often possible to use multigrid methods to solve the (full-sized) linear system arising from discretized PDEs, at cost $O(N)$ [6, 13]. Therefore, using the reduced model with a direct method is effective only when $k \ll N$. The focus of this study is the case when the rank of the reduced basis k is of the magnitude where the cost of direct methods for the reduced problem is not smaller than for solving the full problem, even though k is still of moderate size. In such situations, there may be an advantage to using alternative solution methods.

Consider the use of iterative methods for the reduced model (2.5). In this case, the cost of such methods is $O(pk^2)$, where p is the number of iterations required for convergence; the factor of k^2 comes from the cost of a dense matrix-vector product by $Q^T A(\xi)Q$. Thus, this will be an effective approach when p is small. It is well known that preconditioners are needed for the fast convergence of iterative methods. Thus, we need a preconditioner for the reduced matrix. Consider a reformulated version of (2.5) given by

$$\begin{bmatrix} A^{-1} & Q \\ Q^T & 0 \end{bmatrix} \begin{bmatrix} v \\ u_r \end{bmatrix} = \begin{bmatrix} 0 \\ -Q^T f \end{bmatrix} . \tag{3.1}$$

Equation (3.1) has the form of a saddle point system, a well-studied problem, for which a preconditioner may take the form [13]

$$\begin{bmatrix} F & 0 \\ 0 & S \end{bmatrix} .$$

With the formal choice $F = A^{-1}$, it can be shown that the optimal choice for S is the Schur complement [18], which for (3.1) is

$$S = Q^T A Q \ . \tag{3.2}$$

That this is equivalent to the matrix of the reduced model suggests that the reduced model in (2.5) can be preconditioned with an approximation to the Schur complement.

To produce a preconditioner for (3.2), we will mimic an approach used successfully in a different context (for models of computational fluid dynamics), the so-called least-squares commutator method [13]. Here the Schur complement is approximated by the matrix

$$\hat{P}_S \equiv (Q^T Q)(Q^T A^{-1} Q)^{-1}(Q^T Q) \ . \tag{3.3}$$

Since Q is orthogonal, this operator simplifies to $\hat{P}_S = \left(Q^T A^{-1} Q\right)^{-1}$. This preconditioner depends on A^{-1}, which is the operator we are trying to approximate, and thus is impractical. Recall that A depends on a parameter ξ. We could choose a single representative vector of parameters, $\xi^{(0)}$, to define the preconditioner, which allows the construction of the preconditioner to be moved offline. In this case we would solve k full systems to compute $A^{-1}(\xi^{(0)})Q$ and premultiply by Q^T. A variation of this idea is to use a collection of representative parameter vectors to define a collection of preconditioners, all computed in the offline step.

In the preconditioner $\hat{P}_S$, we can also replace A with a spectrally equivalent operator, i.e., one for which there exist σ_0, σ_1 independent of spatial dimension such that

$$\sigma_0 \leq \frac{y^T A y}{y^T P_A y} \leq \sigma_1 \ . \tag{3.4}$$

Thus we can use a preconditioner designed for A to produce a preconditioner of S, yielding

$$P_S = (Q^T P_A^{-1} Q)^{-1} \quad \text{or} \quad P_S^{-1} = Q^T P_A^{-1} Q \ . \tag{3.5}$$

In this case we can construct P_S^{-1} explicitly by doing the following:

- Constructing what is needed for a representation of P_A^{-1}. We will define P_A^{-1} using an algebraic multigrid (AMG) method. Therefore, this step consists of computing the sequence of coarse grids, grid transfer operators, and smoothing operators obtained for a multigrid solution of systems of discrete PDEs. With these, we have what is needed to apply the action of P_A^{-1} to a vector.
- Explicitly computing the (dense) order-k matrix $Q^T P_A^{-1} Q$ by applying the AMG operation to each of the columns of Q and then premultiplying the matrix $P_A^{-1} Q$ by Q^T.

The construction (offline) of P_S is cheaper than that of $\hat{P}_S$, and in experiments we found $\hat{P}_S$ to provide little (if any) advantage in terms of iteration counts over P_S. In our study of performance we focus on the preconditioner P_S specified by (3.5).

4. Numerical results. To illustrate the effectiveness of these ideas, we apply the reduced basis method to two examples of PDEs with random coefficients. We compare the performance of the iterative solver for the reduced model with the direct reduced solution and the multigrid solution of the full system.

The first example is a steady-state diffusion equation with a parameter-dependent diffusion coefficient,

$$\begin{aligned} -\nabla \cdot a(\vec{x},\xi)\nabla u(\vec{x},\xi) &= f(\vec{x}) \quad \text{in} \quad D \times \Gamma \ , \\ u(\vec{x},\xi) &= g_D(\vec{x}) \quad \text{on} \quad \partial D_D \times \Gamma \ , \\ a(\vec{x},\xi)\tfrac{\partial u(\vec{x},\xi)}{\partial n} &= 0 \quad \text{on} \quad \partial D_N \times \Gamma \ , \end{aligned} \tag{4.1}$$

where $D \subset \mathbb{R}^2$ and the diffusion coefficient, $a(\vec{x},\xi)$, is a random field depending on a vector of m random variables, $\xi = [\xi_1, \xi_2, \ldots, \xi_m]^T$. The second example is a steady-state convection-diffusion-reaction equation with an uncertain reaction coefficient, $r(\vec{x},\xi)$,

$$\begin{aligned} -\nu\nabla^2 u(\vec{x},\xi) + \vec{w} \cdot \nabla u(\vec{x},\xi) + r(\vec{x},\xi)u(\vec{x},\xi) &= f(\vec{x}) \quad \text{in} \quad D \times \Gamma \ , \\ u(\vec{x},\xi) &= g_D(\vec{x}) \quad \text{on} \quad \partial D_D \times \Gamma \ , \\ \tfrac{\partial u(\vec{x},\xi)}{\partial n} &= 0 \quad \text{on} \quad \partial D_N \times \Gamma \ , \end{aligned} \tag{4.2}$$

where the domain $D \subset \mathbb{R}^2$, ν is the diffusion coefficient, $\vec{w}$ is the convective velocity, and $\nabla \cdot \vec{w} = 0$.

We turn now to the methodology used to compute a reduced basis Q. Assume the full discretized model $A(\xi)u_\xi = f$ is defined on a parameter space $\Gamma = \prod_{i=1}^m \Gamma_i$ such that $\xi_i \in \Gamma_i \coloneqq [a_i, b_i]$. The reduced basis is constructed using an adaptive

algorithm summarized in Algorithm 1. The procedure begins with Q as a single vector, the normalized discrete solution $u_{\xi^{(0)}}$, where $\xi^{(0)} = E[\xi]$. The parameter space is randomly sampled M times, and for each sample, ξ, the reduced model is solved with the current Q. This produces an approximation to the solution $\tilde{u}_\xi = Qu_{r,\xi} + u_{bc}$ whose accuracy is estimated by an error indicator, η_ξ. If η_ξ exceeds a predefined tolerance, τ, the full solution for this ξ is computed and the new snapshot, u_ξ, is used to update the reduced basis. The basis matrix Q is augmented using the modified Gram–Schmidt algorithm, ensuring that the basis will have orthogonal columns. We used as an error indicator the relative residual

$$\eta_\xi = \frac{||A(\xi)\tilde{u}_\xi - f||_2}{||f||_2} \ . \tag{4.3}$$

This method is applied to the steady-state diffusion equation and the steady-state convection-diffusion-reaction equation beginning with $M = 2000$ random samples of ξ. This produces a candidate basis Q. To assess the quality of this basis, we computed the reduced solution for an additional 100 samples; if each of these reduced solutions satisfied the error tolerance, then Q was accepted as the reduced basis. For case 1 of the diffusion equation (see below) and the convection-diffusion-reaction equation, this strategy produced an acceptable Q with a few exceptions. In general, $M \geq 3000$ was required for some experiments with the diffusion equation (referred to as case 2 below, where the details are stated).

Algorithm 1. Construction of the reduced basis using random selection.

```
for j = 1 : M do
    Select random sample ξ^(j) ∈ Γ
    Solve the reduced problem Q^T A(ξ^(j)) Q u_{r,ξ^(j)} = Q^T f
    Compute η_{ξ^(j)}
    if η_{ξ^(j)} > τ then
        Compute u(·, ξ^(j)) using the full model
        Use the snapshot to augment Q
    end if
end for
```

Remark. The convergence properties of this strategy for offline computation of the basis are not known, in contrast to greedy search algorithms, which produce reduced bases of quasi-optimal dimension [4]. In a numerical comparison of Algorithm 1 with a greedy algorithm, we found that for multiple examples of the benchmark problems studied in this section, the size of the reduced basis was never more than 10% larger than that produced by a greedy algorithm and in many cases the basis sizes were identical. The cost (in CPU time) of Algorithm 1 is significantly lower. Our concern is efficient implementation of the *online* step, and for simplicity we used Algorithm 1 for the offline computation.

4.1. Diffusion equation. The steady-state diffusion problem with a random coefficient in (4.1) can be used to model the effects of groundwater flow [25]. For more details on this problem, see [9]. The weak formulation for a fixed value of ξ is

$$(a_\xi \nabla u, \nabla v) = (f, v) \ \ \forall v \in H_0^1(D) \ . \tag{4.4}$$

Bilinear Q_1 elements are used to generate a discretized system, $A(\xi)u_\xi = f$, of order N for the full model [13]. We use source term $f(\vec{x}) = 1$. Boundary conditions will be

addressed below for each case.

We consider two finite-dimensional representations of the random field for the diffusion coefficient $a(\vec{x}, \xi)$: a truncated Karhunen–Loève (KL) expansion (case 1) and a piecewise constant coefficient (case 2). These representations give rise to operators with affine dependence on the parameters as discussed in section 2. The truncated KL expansion is defined by

$$a(\vec{x}, \xi(\omega)) = \mu(\vec{x}) + \sum_{i=1}^{m} \sqrt{\lambda_i} a_i(\vec{x}) \xi_i(\omega) \ , \tag{4.5}$$

where $\mu(\vec{x})$ is the mean of the random field, λ_i and $a_i(\vec{x})$ are the eigenvalues and eigenfunctions of the covariance function, and $\xi_i(\omega)$ are independent uniform random variables. We take the covariance function to be

$$C(\vec{x}_1, \vec{x}_2) = \sigma^2 \exp\left(-\frac{|x_1 - x_2|}{c} - \frac{|y_1 - y_2|}{c} \right) \ , \tag{4.6}$$

where σ is the standard deviation and c is the correlation length, which describes the strength of the relationship between the value of the random field at $\vec{x}_1 = (x_1, y_1)$ and $\vec{x}_2 = (x_2, y_2)$. A large value of c implies that $a(\vec{x}_1, \xi)$ and $a(\vec{x}_2, \xi)$ are likely to be highly correlated. We will also use the truncated KL expansion to represent the reaction coefficient in the convection-diffusion-reaction equation (4.2).

For the piecewise constant diffusion coefficient, the domain, D, is divided into $m = n_d \times n_d$ subdomains as in Figure 1, where

$$a(\vec{x}, \xi) = \xi_i \ , \tag{4.7}$$

on the ith subdomain. Here $\{\xi_i\}_{i=1}^m$ are independent uniform random variables defined on $\Gamma_i = [0.01, 1]$.

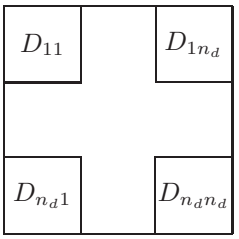


FIG. 1. *Domain for diffusion equation case* 2: *Piecewise random coefficients.*

Consider the influence of the parameters on the overall value of the coefficient for these two representations. The impact of the parameters in the truncated KL expansion is unequal because the expansion weights the parameters by the eigenvalues of the covariance operator. Thus, for example, ξ_1 and ξ_2 are more influential to the value of $a(\vec{x}, \xi)$ than ξ_{m-1} and ξ_m, when the eigenvalues are labeled in decreasing order. In contrast, the piecewise random coefficients are equally weighted.

Algorithm 1 is used to generate the reduced basis Q. Once the reduced basis is generated we are able to solve the reduced problem defined in (2.5). Recall that the preconditioner for this system, discussed in section 3 and defined in (3.5), utilizes P_A^{-1}, a preconditioner of A. We will specify P_A^{-1} using multigrid, which is well known to be effective for diffusion problems [6]. For the implementation, we use a smoothed

aggregation algebraic multigrid routine from the Python Algebraic Multigrid package (PyAMG) with the default settings [3]: the presmoother and postsmoother are one iteration of Gauss–Seidel, the maximum size of the coarse grid is 500, and the pseudoinverse is used to solve the system on the coarse grid. To compute the preconditioner for the reduced problem, the multigrid operator P_A^{-1} is applied to Q by performing one V-cycle on each of the k columns of Q. We study three ways to select the parameter used to specify P_A.

1. Single-parameter offline: P_0 is derived from multigrid applied to $A(\xi^{(0)})$, where $\xi^{(0)}$ is the mean parameter, $E[\xi]$.
2. Multiple-parameter offline: A set of s parameters is used to define s precomputed offline preconditioners, $P_1, \ldots, P_s$. This is done using multigrid applied to $A(\xi^{(1)}), \ldots, A(\xi^{(s)})$. For the online component given ξ, $\xi^{(j)} \in \{\xi^{(1)}, \ldots, \xi^{(s)}\}$ is selected such that $||\xi^{(j)} - \xi||_2$ is minimized and P_j is used as the preconditioner. There are several possibilities for choosing $\{\xi^{(1)}, \ldots, \xi^{(s)}\}$, including random sampling, quasi-random sampling, and sparse grids. Sparse grids are used to limit costs of quadrature and interpolation of functions depending on high-dimensional parameter sets. Since we are working with high-dimensional parameter spaces and would like to represent the parameter space with as few parameters as possible, we choose the so-called No Boundary sparse grid [15]. The first level of the grid, of size $s = 2m + 1$, is obtained using the `spinterp` toolbox [16].
3. Online: $P_{A(\xi)}$ comes from multigrid applied to $A(\xi)$, where ξ is the same parameter whose solution we are seeking. The time to construct the preconditioner online is quite large. It requires building the coarse grid and smoothing operators and the significantly more expensive step of applying them to each column of Q in order to compute $Q^T P_{A(\xi)}^{-1} Q$. Note that this approach does not allow offline construction of the preconditioner and thus is not meant to be used in practice. It is included here to give a lower bound for how well offline preconditioning could perform.

The examples are implemented using Python and run with an Intel 2.9 GHz i7 processor and 8 GB of RAM. (The full model finite element discretizations are imported from the Incompressible Flow and Iterative Solver Software (IFISS) package which is implemented in MATLAB [22].) The full solution is obtained using AMG with stopping criterion

$$||f - A(\xi)u_j||_2 \leq 10^{-5}||f||_2 \,,$$

where u_j is the solution after j iterations of multigrid, implemented with the same settings outlined above. For iterative solution of the reduced problem, we use the preconditioned conjugate gradient (PCG) method with stopping criterion

$$\frac{||Q^T f - Q^T A Q u_{r,j}||}{||Q^T f||} < \frac{\tau}{10} \,, \tag{4.8}$$

where $u_{r,j}$ is the reduced iterate at step j. The given times for online preconditioning do not include the significant time required to construct the multigrid preconditioner, and the time for multiple-parameter preconditioning does not include the trivial time to find the minimizer ξ^*.

Case 1. Truncated KL expansion. The random field, $a(\vec{x}, \xi)$, is represented by a truncated KL expansion defined on $D = [0, 1] \times [0, 1]$ and described in (4.5).

Dirichlet boundary conditions $g_D(\vec{x}) = 0$ are imposed on the boundary where $x = 0$ and $x = 1$ and homogeneous Neumann conditions are used on the remainder of the boundary.

We choose ξ_i to be independent and uniformly distributed random variables on $\Gamma_i = [-1, 1]$ and fix $\mu(\vec{x}) = 1$ and $\sigma = 0.5$. The correlation length c is varied; the number of parameters m is chosen to ensure that 95% of the variance in the random field is captured, i.e.,

$$\frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{N} \lambda_i} \geq 0.95 \, . \tag{4.9}$$

Algorithm 1 with $M = 2000$ was used to construct a basis using both $\tau = 10^{-5}$ and $\tau = 10^{-8}$ for the error tolerance.[1] Decreasing the tolerance has the effect of increasing the size of the reduced basis, and for smaller τ the reduced model solutions from both direct and iterative methods require additional time; this tolerance has no effect on the full system solution.

To assess performance, we solve the reduced problem for 100 randomly chosen parameters using a direct method, the conjugate gradient method without preconditioning, and the conjugate gradient method with three preconditioners: single-parameter offline, multiple-parameter offline, and online. The average iteration counts for the conjugate gradient method are presented in Tables 1 and 3. The time (in seconds) for the full AMG solution, the reduced direct method, and the single-parameter offline conjugate gradient method are presented in Tables 2 and 4 with the fastest method for each case in bold.

TABLE 1

*Average iteration counts for the PCG algorithm applied to the reduced diffusion problem in case 1 (KL expansion), with $\tau = 10^{-5}$. The case with the * is anomalous because the offline preconditioners converge in one less iteration than the online preconditioner for several samples.*

N	c	3	1.5	0.75	0.375
	m	7	17	65	325
33^2	k	36	91	237	501
	None	25.4	44.4	65.6	76.5
	Single	6.0	6.3	6.4	6.4
	Multiple	6.0	6.2	6.4	6.4
	Online	6.0	6.0	6.0	6.0
65^2	k	35	93	250	603
	None	25.1	46.9	82.3	121.7
	Single	6.0	6.1	6.1	6.1
	Multiple	6.0	6.1	6.1	6.1
	Online	6.0	6.0	6.0	6.0
129^2	k	35	95	259	642
	None	24.8	49.4	90.3	150.2
	Single	6.1	7.2	8.0	8.1
	Multiple	6.1	7.1	8.0	8.1
	Online	6.0	7.0	8.0	8.0
257^2	k	35	96	263	657
	None	24.0	49.4	92.1	161.7
	Single	6.1	7.6	8.0	8.1
	Multiple	6.1	7.6	8.0	8.1
	Online	6.0	8.0*	8.0	8.0

[1] The example with $m = 325$ parameters (see Table 1) required $M = 3000$ for $\tau = 10^{-5}$, $N = 257^2$, and $\tau = 10^{-8}$, $N \geq 65^2$.

TABLE 2
Average CPU time solving the reduced diffusion problem in case 1 *(KL expansion), with* $\tau = 10^{-5}$.

N	c		3	1.5	0.75	0.375
	m		7	17	65	325
33^2	k		36	91	237	501
	Full	AMG	0.0206	0.0232	0.0211	0.0219
	Reduced	Direct	**0.0001**	**0.0003**	0.0014	0.0100
	Reduced	Iterative	0.0003	0.0004	**0.0006**	**0.0018**
65^2	k		35	93	250	603
	Full	AMG	0.1770	0.1786	0.1850	0.1866
	Reduced	Direct	**0.0001**	**0.0003**	0.0016	0.0183
	Reduced	Iterative	0.0003	**0.0003**	**0.0005**	**0.0025**
129^2	k		35	95	259	642
	Full	AMG	0.1161	0.1251	0.1248	0.1304
	Reduced	Direct	**0.0002**	**0.0003**	0.0018	0.0205
	Reduced	Iterative	0.0003	0.0004	**0.0007**	**0.0037**
257^2	k		35	96	263	657
	Full	AMG	0.3065	0.3300	0.3193	0.3063
	Reduced	Direct	**0.0002**	**0.0003**	0.0024	0.0222
	Reduced	Iterative	0.0003	0.0004	**0.0008**	**0.0039**

TABLE 3
Average iteration counts for the PCG algorithm applied to the reduced diffusion problem in case 1 *(KL expansion), with* $\tau = 10^{-8}$.

N	c	3	1.5	0.75	0.375
	m	7	17	65	325
33^2	k	97	254	607	982
	None	60.1	90.7	101.7	103.9
	Single	10.0	9.3	9.5	8.9
	Multiple	10.0	9.3	9.5	8.9
	Online	10.0	9.0	9.0	8.0
65^2	k	100	265	699	1679
	None	68.8	129.3	175.5	200.3
	Single	10.0	10.0	8.5	8.7
	Multiple	10.0	10.0	8.5	8.7
	Online	10.0	9.8	8.0	8.0
129^2	k	102	269	729	1808
	None	70.1	149.5	252.5	339.1
	Single	11.2	14.6	12.9	11.0
	Multiple	11.2	14.6	12.9	11.0
	Online	11.0	14.8	13.0	11.0
257^2	k	102	275	740	1846
	None	70.4	154.0	293.6	473.7
	Single	11.0	13.7	15.4	13.5
	Multiple	11.0	13.7	15.4	13.5
	Online	11.0	13.0	15.0	13.0

Tables 1 and 3 show that the number of iterations needed for PCG grows only slightly as the size of the reduced basis grows, whereas the iterations for unpreconditioned conjugate gradient grow significantly. Also note that the single-parameter preconditioner performs nearly as well as the online preconditioner, so using the mean parameter to construct the preconditioner is an effective choice for the entire parameter space.

Tables 2 and 4 illustrate that the single-parameter offline PCG method is faster than direct methods when the reduced basis is of size $k \geq 237$. For $\tau = 10^{-5}$, this holds for $m = 65$, and for $\tau = 10^{-8}$, this holds for both $m = 17$ and $m = 65$. The

TABLE 4
Average CPU time solving the reduced diffusion problem in case 1 (KL expansion), with $\tau = 10^{-8}$.

N	c		3	1.5	0.75	0.375
	m		7	17	65	325
33^2	k		97	254	607	982
	Full	AMG	0.0202	0.0205	0.0214	0.0228
	Reduced	Direct	**0.0003**	0.0016	0.0181	0.0699
	Reduced	Iterative	0.0004	**0.0008**	**0.0036**	**0.0103**
65^2	k		100	265	699	1679
	Full	AMG	0.1768	0.1961	0.1947	0.1974
	Reduced	Direct	**0.0003**	0.0021	0.0262	0.3207
	Reduced	Iterative	0.0004	**0.0010**	**0.0044**	**0.0252**
129^2	k		102	269	729	1808
	Full	AMG	0.1195	0.1286	0.1347	0.1443
	Reduced	Direct	**0.0003**	0.0020	0.0287	0.4452
	Reduced	Iterative	0.0005	**0.0013**	**0.0070**	**0.0449**
257^2	k		102	275	740	1846
	Full	AMG	0.3163	0.2988	0.3030	0.3778
	Reduced	Direct	**0.0004**	0.0024	0.0302	0.4498
	Reduced	Iterative	0.0005	**0.0012**	**0.0088**	**0.0619**

improvement is more dramatic for the case of $m = 65$ and $\tau = 10^{-8}$, when the reduced basis size is $k \approx 700$. For all values of m and N the reduced iterative method is more efficient than solving the full system.

For this example, the size of the reduced basis is consistent as the spatial size, N, is increased. This is especially clear for the smaller values of m. This is expected; see discussion in [12] suggesting that this size is in correspondence with the rank of the underlying solution space associated with the continuous model. There is some growth in k, the basis size, for the larger values of m, but we expect these values to eventually tend toward a constant as the spatial resolution increases. Since the cost of solving the full system grows with N, as expected, the advantage of using the reduced model also increases as the mesh is refined.

Case 2. Piecewise constant coefficient. The diffusion coefficient, $a(\vec{x}, \xi)$, for this case is defined in (4.7) on a domain $D = [-1, 1] \times [-1, 1]$ with $g_D(\vec{x}) = 0$ on the entire boundary. Algorithm 1 with $M = 3000$ and $\tau = 10^{-8}$ was used to construct the bases.[2] The average iteration counts for solving 100 reduced problems are given in Table 5 for the conjugate gradient method.

In contrast to the results for case 1, the iteration counts for the offline preconditioners are somewhat larger than those for the online ones (see Table 5). We attribute this to the fact that for this example, all the parameters are weighted equally in their contribution to the model, unlike the situation for the KL expansion. Thus, the single (or small) number of parameter sets used for the offline preconditioners are not as effective at capturing the character of the parameter space. Despite this, the important trends for the preconditioned solvers are the same as for case 1: iteration counts depend only mildly on the number of terms m in (2.3) or the size k of the reduced basis. There is little advantage of the "multiple-parameter" over the "single-parameter" approach. Thus we use this single-parameter PCG method as the iterative method to compare to the reduced direct and full multigrid methods in Table 6.

We highlight the trends displayed in Table 6 as follows:

[2]The example with $m = 100$ parameters and $N = 257^2$ required $M = 4000$ to construct a basis that meets the criteria discussed earlier in this section.

TABLE 5
Average iteration counts for the PCG algorithm applied to the reduced diffusion problem in case 2, with $\tau = 10^{-8}$.

	m	4	16	36	64	100
33^2	k	27	193	321	449	577
	None	31.9	113.9	126.4	127.9	128.0
	Single	17.2	32.3	44.0	52.0	59.5
	Multiple	15.7	30.1	42.4	50.3	57.0
	Online	11.4	13.0	13.6	12.7	12.0
65^2	k	29	309	625	897	1153
	None	42.3	234.1	254.9	258.3	256.4
	Single	20.1	38.2	47.0	54.8	64.2
	Multiple	18.7	35.5	44.9	53.1	65.4
	Online	14.3	17.0	18.2	18.9	18.9
129^2	k	33	359	862	1519	2219
	None	60.3	432.9	493.6	519.2	518.9
	Single	24.2	37.5	47.6	58.1	64.9
	Multiple	22.7	35.2	45.2	56.0	72.4
	Online	19.1	19.0	22.0	24.1	25.2
257^2	k	36	394	979	1789	2801
	None	82.0	808.9	976.8	1035.6	1037.3
	Single	30.4	44.0	50.9	62.2	71.1
	Multiple	28.6	41.7	48.5	60.3	84.1
	Online	25.1	25.8	25.7	27.8	29.7

TABLE 6
Average CPU time solving the reduced diffusion problem in case 2 (piecewise constant), with $\tau = 10^{-8}$.

N	m		4	16	36	64	100
33^2	k		27	193	321	449	577
	Full	AMG	0.0218	0.0203	0.0215	0.0210	0.0208
	Reduced	Direct	**0.0001**	**0.0010**	**0.0032**	**0.0073**	**0.0152**
	Reduced	Iterative	0.0006	0.0019	0.0045	0.0090	0.0181
65^2	k		29	309	625	897	1153
	Full	AMG	0.1679	0.1601	0.1669	0.1811	0.1760
	Reduced	Direct	**0.0002**	**0.0026**	0.0187	0.0543	0.1088
	Reduced	Iterative	0.0007	0.0034	**0.0176**	**0.0458**	**0.0832**
129^2	k		33	359	862	1519	2219
	Full	AMG	0.1134	0.1202	0.1357	**0.1184**	**0.1194**
	Reduced	Direct	**0.0002**	**0.0038**	0.0461	0.2319	0.6659
	Reduced	Iterative	0.0009	0.0041	**0.0364**	0.1340	0.3060
257^2	k		36	394	979	1789	2801
	Full	AMG	0.3376	0.3519	0.3291	0.3365	**0.3568**
	Reduced	Direct	**0.0002**	**0.0051**	0.0670	0.3555	1.2972
	Reduced	Iterative	0.0010	0.0060	**0.0485**	**0.1928**	0.5365

- For the reduced problem, the iterative solver is more efficient than the direct solver for large reduced bases, in particular whenever the size k of the reduced basis is greater than or equal to 625.
- As the dimension of the spatial discretization increases, the solution of the reduced model is less expensive than the solution of the full model. Moreover, as in case 1, the size of the reduced basis tends to a constant as the mesh is refined, so solution costs also tend to a constant.
- For fixed spatial dimension, the costs of solving the full system are constant, whereas the size of the reduced model increases with the number of parameters, m, and N. For the largest choices of these values, $m = 100$ and

$N = 257^2$, the full AMG costs are lowest. However, for fine enough spatial meshes such that k has stabilized (as in case 1), we expect that the cost of the reduced model will be smaller.

4.2. Behavior of eigenvalues. The performance of the conjugate gradient method for solving the reduced problem depends on the extremal values of the Rayleigh quotient

$$\frac{x^T Q^T A Q x}{x^T (Q^T P_A^{-1} Q)^{-1} x} = \frac{x^T Q^T A Q x}{x^T (Q^T A^{-1} Q)^{-1} x} \; \frac{x^T (Q^T A^{-1} Q)^{-1} x}{x^T (Q^T P_A^{-1} Q)^{-1} x} \; . \tag{4.10}$$

Specifically, let us consider the second quotient on the right-hand side of (4.10) for the online preconditioner (i.e., where A and P_A come from the same parameter ξ). Note that an offline preconditioner requires an additional problem-dependent quotient which depends on the relationship between the parameter used to construct the preconditioner and the problem we are trying to solve. We have assumed in (3.4) that P_A is spectrally equivalent to A. When A and P are symmetric positive definite, an analogous bound also holds for the inverses [24],

$$\sigma_0 \leq \frac{y^T P_A^{-1} y}{y^T A^{-1} y} \leq \sigma_1 \;\; \forall y \in \mathbb{R}^N \; .$$

We have assumed that this bound holds for all y, so specifically it holds for y on the range of Q (i.e., $y = Qx$). Using this fact and applying inverses yields

$$\sigma_0 \leq \frac{x^T (Q^T A^{-1} Q)^{-1} x}{x^T (Q^T P_A^{-1} Q)^{-1} x} \leq \sigma_1 \; . \tag{4.11}$$

Therefore the second quotient in (4.10) is bounded by σ_0 and σ_1.

We can obtain insight into the first quotient of (4.10) by experimentally examining the eigenvalues of

$$Q^T A(\xi^{(0)})^{-1} Q Q^T A(\xi^{(0)}) Q$$

using the benchmark problem from the previous section, case 2 of the diffusion equation. Figure 2 illustrates the eigenvalues for four values of m considered for this problem. All eigenvalues are bounded below by 1, and the largest eigenvalues grow only slightly with spatial dimension for the three cases where $m > 4$. This suggests that the condition number of the preconditioned reduced matrix is only weakly dependent on the spatial mesh.

4.3. Convection-diffusion-reaction equation. The convection-diffusion-reaction equation (4.2) has applications in modeling fluid flow and chemical reactions. It can be used to model the transportation of contaminants in a flow subject to diffusive effects and/or chemical reactions [17]. Such models depend on parameters for the diffusion coefficient, the velocity, and the reaction coefficient. Any of these parameters could be uncertain [23]; here we consider the case where the reaction rate is taken to be a random field depending linearly on a random vector. The weak formulation is

$$\nu(\nabla u, \nabla v) + (\vec{w} \cdot \nabla u, v) + (r_\xi u, v) = (f, v) \;\; \forall v \in H_0^1(D) \; . \tag{4.12}$$

We present results for the steady-state model posed on domain $D = [-1, 1] \times [-1, 1]$ with Dirichlet boundary conditions

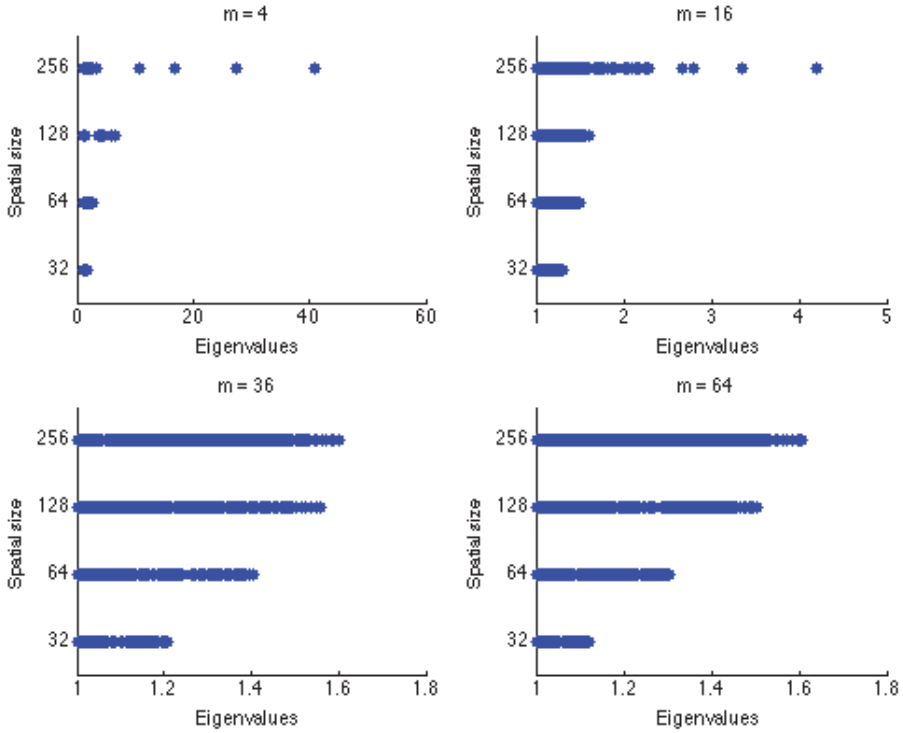


FIG. 2. *Eigenvalues of* $Q^T A^{-1} Q Q^T A Q$.

$$g_D(\vec{x}) = \begin{cases} 0 & \text{for} \quad [-1, y] \bigcup [x, 1] \bigcup [-1 \leq x \leq 0, -1] \\ 1 & \text{for} \quad [1, y] \bigcup [0 \leq x \leq 1, -1] \end{cases} , \tag{4.13}$$

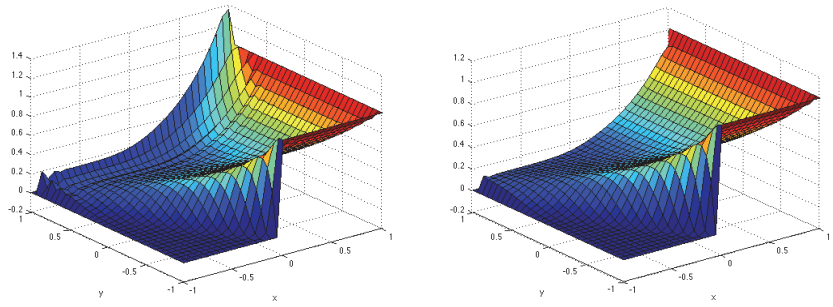
and an inflow boundary condition on the boundaries, $[x, -1]$ and $[1, y]$. We use source term $f(\vec{x}) = 0$ and a constant velocity $\vec{w} = (-\sin\frac{\pi}{6}, \cos\frac{\pi}{6})$. The diffusion coefficient is $\nu = 0.005$. The reaction rate, $r(x, \xi)$, is represented by a truncated KL expansion as in (4.5), with ξ_i independent and uniformly distributed on $\Gamma_i = [-1, 1]$, with mean $\mu = 1$, and with standard deviation $\sigma = 0.5$. As in case 1 of the diffusion equation, the value of the correlation coefficient c is varied, and the number of parameters m is chosen to capture 95% of the variance of the random field.

We again discretize using bilinear finite elements, which yields operators A, B, and $R(\xi)$, in which A represents the diffusion term, B the convective term, and $R(\xi)$ the reaction term. We include stabilization by the streamline-diffusion method [14] in the convection-dominated case when the mesh Peclet number

$$P_h = \frac{h_e ||\vec{w}||}{2\nu} > 1 , \tag{4.14}$$

where h_e is a measure of the element length in the direction of the wind. This method produces matrices S_{cd} and S_r, defined in terms of the finite element basis functions $\{\phi_i\}_{i=1}^{n_e}$ as

$$[S_{cd}]_{ij} = \sum_{e=1}^{n_e} \int_{\Omega_e} \delta_e (\vec{w} \cdot \nabla \phi_i)(-\nabla \cdot (\nu \nabla \phi_j) + \vec{w} \cdot \nabla \phi_j)$$



(a) Without streamline-diffusion method (b) With streamline-diffusion method

FIG. 3. *Solution of the convection-diffusion-reaction problem for* $N = 33^2$, $\xi = \xi_0$, $c = 2$, $m = 36$, $P_h = 7.2$ *with and without streamline-diffusion stabilization.*

and

$$[S_r(\xi)]_{ij} = \sum_{e=1}^{n_e} \int_{\Omega_e} \delta_e(\vec{w} \cdot \nabla \phi_i) r(\vec{x}, \xi) \phi_j \ ,$$

where [13, p. 247]

$$\delta_e = \frac{h_e}{2||\vec{w}||} \left(1 - \frac{1}{P_h}\right) \ .$$

The resulting linear system has the form

$$F(\xi) u_\xi = f \ , \tag{4.15}$$

where $F(\xi) = A + B + R(\xi) + S_{cd} + S_r(\xi)$. As is well known [1, 13, 19], this stabilization enhances the quality of solutions with steep gradients obtained using inadequately fine grids, limiting the presence of nonphysical oscillations in discrete solutions; see Figure 3. As the discretization is refined, the stabilization becomes unnecessary.

We now consider solving the reduced problem

$$Q^T F(\xi) Q u_r = Q^T f \ . \tag{4.16}$$

(This formulation corresponds to a stabilized version of the reduced model referred to as an "offline-online" stabilized method in [19]. Cf. also [10] for alternative ways of handling models containing convection terms.) As above, we use iterative methods where Q is constructed using Algorithm 1 with $M = 2000$ and $\tau = 10^{-8}$. Since the system is not symmetric, we use the stabilized biconjugate gradient method (BICGSTAB) in conjunction with preconditioner $Q^T P_F^{-1} Q$, where P_F^{-1} is constructed using one of two methods:

1. Offline: P_F^{-1} is a multigrid preconditioner of $F(\xi^{(0)})$, where $\xi^{(0)}$ is the mean of the parameter space, $E[\xi]$.

2. Online: P_F^{-1} is a multigrid preconditioner of $F(\xi)$.

As with the diffusion equation, the multigrid preconditioners are constructed using a smoothed aggregation AMG routine from PyAMG [3]. The examples with $N \leq 129^2$ required streamline-diffusion stabilization; for $N = 257^2$, this was not needed. However, in this case AMG required a different smoothing operator, the normed residual Gauss–Seidel smoother, where Gauss–Seidel is applied to the normal equations instead of the standard Gauss–Seidel smoother [3, 21]. We attribute this to instability of the coarse grid operators.

Table 7 contains the average iterations for BICGSTAB to solve the reduced model for 100 randomly selected parameters. We observe that the offline preconditioner is also effective for this problem. In terms of iteration counts, the offline preconditioner performs nearly as well as the online preconditioner as in case 1 of the diffusion equation. The times for offline preconditioned BICGSTAB, reduced direct, and full multigrid methods are shown in Table 8. The reduced iterative method is faster than the direct method for $m = 785$. Since decreasing N has the effect of decreasing k for this problem, the iterative methods perform best for $m = 145$, $N = 33^2, 65^2$, corresponding to $k = 372$ and greater.

TABLE 7

Average iteration counts for the reduced problem solved using BICGSTAB for the convection-diffusion-reaction problem, $\tau = 10^{-8}$.

N	c	2	1	0.5
	m	36	145	785
33^2	k	210	421	798
	None	49.5	45.9	41.7
	Single	8.2	7.0	6.1
	Online	8.3	7.0	6.0
65^2	k	178	372	952
	None	84.5	87.4	86.5
	Single	12.0	10.0	9.0
	Online	12.0	10.0	9.0
129^2	k	138	265	749
	None	122.8	153.0	176.2
	Single	12.9	13.1	13.0
	Online	12.7	13.5	13.0
257^2	k	99	197	686
	None	126.8	234.0	293.8
	Single	14.2	14.4	15.1
	Online	13.9	14.5	15.0

5. Conclusion. Reduced basis methods are an efficient way to obtain the solution to parameterized PDEs for many parameter values. The effectiveness of reduced basis methods depends on the relatively cheap cost of solving the reduced problem. This cost depends on the rank of the reduced basis, which depends on quantities such as the number of parameters, m, and the accuracy desired for the reduced solution τ. We have shown, using two examples, the steady-state diffusion equation and the convection-diffusion-reaction equation, that this cost can be reduced for larger k when iterative methods are used, and we have identified the regime of k where, for these problems, iterative methods for the reduced problem become the most effective choice. This is accomplished primarily by designing a preconditioner which (1) is computed offline, and thus does not increase the cost of solving the reduced model, and (2) is derived from a multigrid preconditioner for the full model.

TABLE 8
Comparison of time of the BICGSTAB algorithm with full model solved using multigrid and the reduced model solved using the direct method for the convection-diffusion-reaction problem, $\tau = 10^{-8}$.

N	c		2	1	0.5
	m		36	145	785
33^2	k		210	421	798
	Full	AMG	0.0419	0.0428	0.0440
	Reduced	Direct	0.0011	0.0067	0.0400
	Reduced	Iterative	**0.0009**	**0.0019**	**0.0066**
65^2	k		178	372	952
	Full	AMG	0.2188	0.2258	0.2311
	Reduced	Direct	**0.0009**	0.0046	0.0679
	Reduced	Iterative	0.0013	**0.0022**	**0.0148**
129^2	k		138	265	749
	Full	AMG	0.3228	0.3284	0.3271
	Reduced	Direct	**0.0007**	**0.0020**	0.0323
	Reduced	Iterative	0.0012	**0.0020**	**0.0132**
257^2	k		99	197	686
	Full	AMG	1.5330	1.5468	1.5396
	Reduced	Direct	**0.0003**	**0.0010**	0.0234
	Reduced	Iterative	0.0010	0.0016	**0.0140**

Acknowledgment. We thank Harbir Antil for several helpful discussions.

REFERENCES

[1] H. ANTIL, M. HEINKENSCHLOSS, AND D. C. SORENSEN, *Application of the discrete empirical interpolation method to reduced order modeling of nonlinear and parametric systems*, in Reduced Order Methods for Modeling and Computational Reduction, MSA. Model. Simul. Appl. 9, G. Rozza, ed., Springer, Cham, Switzerland, 2014, pp. 101–136.

[2] C. AUDOUZE, F. DE VUYST, AND P. B. NAIR, *Reduced-order modeling of parameterized PDEs using time–space-parameter principal component analysis*, Internat. J. Numer. Methods Engrg., 80 (2009), pp. 1025–1057.

[3] W. N. BELL, L. N. OLSON, AND J. B. SCHRODER, *PyAMG: Algebraic multigrid solvers in Python v*2.0, 2011. Release 2.0.

[4] P. BINEV, A. COHEN, W. DAHMEN, R. DEVORE, G. PETROVA, AND P. WOJTASZCZYK, *Convergence rates for greedy algorithms in reduced basis methods*, SIAM J. Math. Anal., 43 (2011), pp. 1457–1472.

[5] S. BOYAVAL, C. LE BRIS, T. LELIÈVRE, Y. MADAY, N. C. NGUYEN, AND A. T. PATERA, *Reduced basis techniques for stochastic problems*, Arch. Comput. Methods Eng., 17 (2010), pp. 435–454.

[6] W. L. BRIGGS, V. E. HENSON, AND S. F. MCCORMICK, *A Multigrid Tutorial*, SIAM, Philadelphia, 2000.

[7] T. BUI-THANH, K. WILLCOX, AND O. GHATTAS, *Model reduction for large-scale systems with high-dimensional parametric input space*, SIAM J. Sci. Comput., 30 (2008), pp. 3270–3288.

[8] S. CHATURANTABUT AND D. C. SORENSEN, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 32 (2010), pp. 2737–2764.

[9] P. CHEN, A. QUARTERONI, AND G. ROZZA, *A weighted reduced basis method for elliptic partial differential equations with random input data*, SIAM J. Numer. Anal., 51 (2013), pp. 3163–3185.

[10] W. DAHMEN, C. PLESKEN, AND G. WELPER, *Double greedy algorithms: reduced basis methods for transport dominated problems*, ESAIM Math. Model. Numer. Anal., 48 (2014), pp. 623–663.

[11] J. L. EFTANG, A. T. PATERA, AND E. M. RØNQUIST, *An "hp" certified reduced basis method for parameterized elliptic partial differential equations*, SIAM J. Sci. Comput., 32 (2010), pp. 3170–3200.

[12] H. C. ELMAN AND Q. LIAO, *Reduced basis collocation methods for partial differential equations with random coefficients*, SIAM/ASA J. Uncertain. Quantif., 1 (2013), pp. 192–217.

[13] H. C. ELMAN, D. J. SILVESTER, AND A. J. WATHEN, *Finite Elements and Fast Iterative Solvers:*

With Applications in Incompressible Fluid Dynamics, 2nd ed., Oxford University Press, Oxford, UK, 2014.

[14] T. J. R. Hughes and A. Brooks, *A multi-dimensional upwind scheme with no crosswind diffusion*, in Finite Element Methods for Convection Dominated Flows, AMD 34, T. J. R. Hughes, ed., ASME, New York, 1979.

[15] A. Klimke, *Sparse Grid Interpolation Toolbox – User's Guide*, Tech. Report IANS report 2007/017, University of Stuttgart, Stuttgart, Germany, 2007.

[16] A. Klimke and B. Wohlmuth, *Algorithm* 847: *spinterp: Piecewise multilinear hierarchical sparse grid interpolation in MATLAB*, ACM Trans. Math. Software, 31 (2005), pp. 561–579.

[17] J. D. Logan, *Transport Modeling in Hydrogeochemical Systems*, Interdiscip. Appl. Math. 15, Springer, New York, 2001.

[18] M. F. Murphy, G. H. Golub, and A. J. Wathen, *A note on preconditioning for indefinite linear systems*, SIAM J. Sci. Comput., 21 (2000), pp. 1969–1972.

[19] P. Pacciarini and G. Rozza, *Stabilized reduced basis method for parametrized advection-diffusion PDEs*, Comput. Methods Appl. Mech. Engrg., 274 (2014), pp. 1–18.

[20] A. Quarteroni and G. Rozza, *Numerical solution of parametrized Navier-Stokes equations by reduced basis methods*, Numer. Methods Partial Differential Equations, 23 (2007), pp. 923–948.

[21] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.

[22] D. Silvester, H. Elman, and A. Ramage, *Incompressible Flow and Iterative Solver Software (IFISS) version* 3.3, October 2013; available online from http://www.manchester.ac.uk/ifiss/.

[23] D. Venturi, D. M. Tartakovsky, A. M. Tartakovsky, and G. E. Karniadakis, *Exact PDF equations and closure approximations for advective-reactive transport*, J. Comput. Phys., 243 (2013), pp. 323–343.

[24] J. Xu, *Iterative methods by space decomposition and subspace correction*, SIAM Rev., 34 (1992), pp. 581–613.

[25] D. Zhang, *Stochastic Methods for Flow in Porous Media: Coping with Uncertainties*, Academic Press, San Diego, CA, 2001.