Journal of Computational Physics 475 (2023) 111835

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Coarse-proxy reduced basis methods for integral equations

Philip A. Etter [a,*], Yuwei Fan [b], Lexing Ying [c]

[a] Meta Reality Labs, Redmond, WA 98052, United States of America
[b] Department of Mathematics, Stanford University, Stanford, CA 94305, United States of America
[c] Department of Mathematics and ICME, Stanford University, Stanford, CA 94305, United States of America

ARTICLE INFO

Article history:
Received 4 November 2021
Received in revised form 1 October 2022
Accepted 1 December 2022
Available online 22 December 2022

Keywords:
Coarse-proxy
Reduced basis method
Model order reduction
Skeleton extraction
Integral equations

ABSTRACT

In this paper, we introduce a new reduced basis methodology for accelerating the computation of large parameterized systems of high-fidelity integral equations. Core to our methodology is the use of *coarse-proxy* models (i.e., lower resolution variants of the underlying high-fidelity equations) to identify important samples in the parameter space from which a high quality reduced basis is then constructed. Unlike the more traditional POD or greedy methods for reduced basis construction, our methodology has the benefit of being both easy to implement and embarrassingly parallel. We apply our methodology to the under-served area of integral equations, where the density of the underlying integral operators has traditionally made reduced basis methods difficult to apply. To handle this difficulty, we introduce an operator interpolation technique, based on random sub-sampling, that is aimed specifically at integral operators. To demonstrate the effectiveness of our techniques, we present two numerical case studies, based on the radiative transport equation and a boundary integral formation of the Laplace equation respectively, where our methodology provides a significant improvement in performance over the underlying high-fidelity models for a wide range of error tolerances. Moreover, we demonstrate that for these problems, as the coarse-proxy selection threshold is made more aggressive, the approximation error of our method decreases at an approximately linear rate. Finally, we provide a public repository of our source code with easy instructions for reproducing all results in this paper.

1. Introduction

Across virtually all areas of science and engineering, physics-based numerical simulation has become an absolutely indispensable tool for the advancement of knowledge and the design of industrial products. However, as with any tool, there are always practical caveats. In particular, high-fidelity simulations often require tremendous computational resources and time to execute. This computational cost often precludes high-fidelity simulations from being used in many important problems, such as uncertainty quantification or Bayesian inference, that require not just one, but many queries to the underlying computational model. Making these many-query problems tractable often requires fast approximation techniques to mitigate the sheer computational cost of multiple queries to the underlying (full-order) model.

* Corresponding author.
E-mail addresses: paetter@meta.com (P.A. Etter), ywfan@stanford.edu (Y. Fan), lexing@stanford.edu (L. Ying).

https://doi.org/10.1016/j.jcp.2022.111835
0021-9991/

One such class of approximation techniques is reduced order models (ROMs). Reduced order models typically operate in two stages. First, there is a computationally expensive *offline* stage (i.e., *training* stage), wherein the ROM is trained on a collection of solutions to the full-order model (FOM). In many cases, this entails finding a basis for a low-dimensional linear subspace which captures solutions to the full-order model (i.e., a reduced basis). Once this offline stage is complete, the reduced-order model can be deployed in an *online* stage (i.e., *test* stage), where these methods can compute fast approximations to new problem instances by exploiting the problem structure learned during the offline phase. For example, one can project the new problem instance onto a set of reduced basis and solve a low-dimensional reduced problem instead of the high-dimensional full-order problem. We refer the reader to [19,6] and references therein for a more thorough overview of this topic.

In this paper, we deal specifically with the class of ROM techniques falling under the reduced basis method (RBM) [37,30]. The groundwork for the reduced basis method was set in the late 1970s with work on the approximation for nonlinear structure analysis [1,27,29], particularly for beams and arches. This groundwork later evolved into a more general framework for parameterized differential equations [17,32], with a corresponding collection of mathematical analyses of the approximation error of the method [36,5,13,34]. These nascent methods typically involved finding a low dimensional approximation space around a parameter of interest — thereby making them local approximation methods. Later, this line of inquiry evolved into finding a global approximation space constructed from a sparse set of sampled solutions to the full-order model [4,21]. More recently, the first theoretical a priori convergence guarantee was proved and numerically confirmed in [26]. This demonstrated the potential of reduced basis methods as a robust approximation for parameterized partial differential equations.

However, while these techniques are well-established for ordinary and partial differential equations, there has been relatively little work done in the regime of model order reduction for integral equations. The current most notable contributions in this under-served area are tailored specifically to boundary element formulations of the electric field equations [12,20,33,14]. The chief factor that contributes to this research gap is likely the difficulties that come from the operators that arise from discretizing integral equations, which are typically *dense*. This operator density precludes one from assembling the operators outright, which limits the applicability of many existing model order reduction techniques, in part because even sampling a single entry of the problem residual takes time on the order of the problem size. Regardless, this gap in the literature is unfortunate, as integral equations have many desirable properties over their differential counterparts. Integral equations are often better conditioned than differential equations (see chapter 6 of [22] or chapters 1, 3, and 4 of [3]). Moreover, many physical models (e.g., electromagnetism, acoustics, radiative transport, etc.) are amenable to special integral formulations with desirable properties. For example, boundary integral formulations are quintessential tools in studying electromagnetic or acoustic wave propagation in free-space. These formulations convert a 3-dimensional differential equation over a domain into a 2-dimensional integral equation over the boundary of that domain. While this reduction in dimension comes with a tradeoff — namely exhibiting dense operators instead of sparse ones — there are many mature hierarchical techniques that make solving dense integral equations with iterative methods highly efficient (i.e., Fast Multipole Method [16]).

In this paper, we develop a model reduction strategy tailored to integral equations that combines ideas from *multi-fidelity modeling* and *random matrix sampling*. In the multi-fidelity modeling literature, a common problem is combining an expensive high-fidelity (fine) model and an inexpensive low-fidelity (coarse) model to obtain high-fidelity accurate with the computational cost similar to that of the low-fidelity model. This has applications in domain areas such as stochastic collocation, where a low-fidelity model and high-fidelity model can be combined to compute high-quality collocation nodes for the parameter set of a system of parameterized differential equations [28,40]. This approach to approximating solutions to systems of parameterized differential equations is notably different from a reduced basis approach because it explicitly models the relationship between system parameter and solution via a low rank stochastic basis (e.g., polynomial chaos), whereas model reduction problems usually just constrain the system to a limited subspace. Other work on multi-fidelity techniques includes the problem of allocating limited computational resources to a set of high-fidelity models in a situation where a less accurate low-fidelity model is available [31]. Finally, we note that substantial effort has been devoted to the construction of error estimators for bi-fidelity models of the sort discussed above — these error estimators may be used to produce better pairings of high and low fidelity models [18].

In this paper, we approach model reduction assuming that we have access to a high-fidelity (fine) and low-fidelity (coarse-proxy) model pair. We use the coarse-proxy model to determine a set of candidates (skeleton set) of parameters whose corresponding high-fidelity solutions can be used to form a high-quality basis for building a reduced model. Afterwards, we use random operator sampling to reconstruct integral operators in the reduced space spanned by this basis. As we will demonstrate, this combination leads to a novel model reduction algorithm that performs quite well on parameterized integral equation problems.

2. Problem statement

The goal of this paper is to solve parameterized integral equations of the form

$$\mathsf{L}(\omega)\mathsf{u}(\omega) = \mathsf{f}(\omega), \qquad \omega \in \Omega_\infty, \tag{2.1}$$

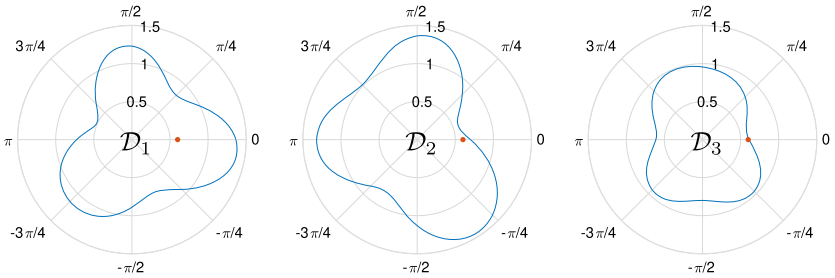


Fig. 2.1. An example of three different domains $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ for the Laplace equation. In our example, each constitutes a different parameter ω in the set Ω_∞.

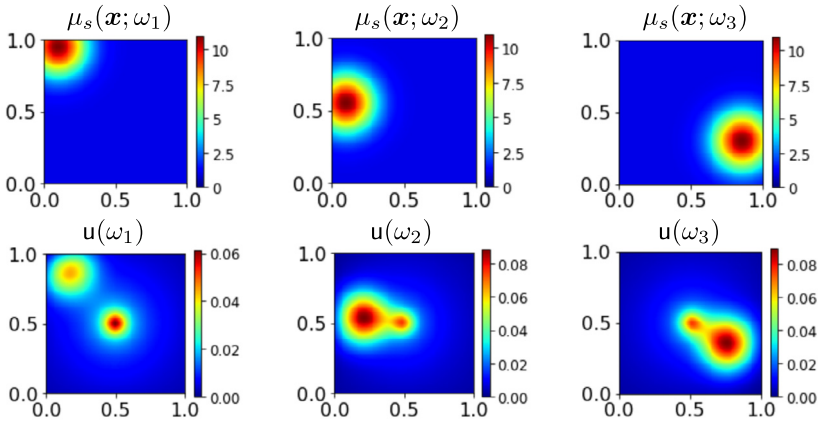


Fig. 2.2. An example of solutions to the radiative transport equation where photons are emitted from the center of a scattering medium. In our example, the scattering coefficient μ_s (top row) depends on a parameter ω that determines the location of an area with high scattering (red). These scattering coefficients produce the corresponding steady-states $\mathsf{u}(\omega)$ in the bottom row. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

where $\mathsf{L}(\omega) \in \mathbb{R}^{n\times n}$ denotes a (dense) linear elliptic integral operator, $\mathsf{f}(\omega) \in \mathbb{R}^n$ denotes a source term, and ω are parameters taken from some sample space Ω_∞. Here, we use n to refer to the number of degrees of freedom in the problem after an appropriate spatial discretization is applied.

This formulation is quite abstract, so, to connect it to an actual physical problem we will quickly reference two settings that we will use as numerical examples at the end of this paper. First, consider the problem of solving the Laplace equation over a prescribed region $\mathcal{D}$ with boundary conditions given on $\partial\mathcal{D}$. This is a simplification of a problem one might encounter in electromagnetic or acoustic scattering, where it is common to solve a Kirchoff integral equation with boundary conditions prescribed over $\partial\mathcal{D}$ [24]. One is usually interested in several different shapes of the domain $\mathcal{D}$ (see for example, Fig. 2.1). In this example, the domain $\mathcal{D}$ corresponds to our parameter ω, the set Ω_∞ is the set of all domains of interest, $\mathsf{L}(\omega)$ represents a discretized scattering operator, $\mathsf{u}(\omega)$ represents the resulting field, and $\mathsf{f}(\omega)$ represents boundary conditions.

Second, consider the problem of light propagation through an isotropic scattering medium, where the wavelength of the light is small compared the obstacles within the scattering medium (i.e., the geometric optics limit). In this situation, as one releases photons into the medium, the photons have a small chance of scattering as they travel. At any point $\boldsymbol{x}$ along their path, the probability of this event is proportional to the scattering coefficient $\mu_s(\boldsymbol{x})$ of the medium. For this problem, the steady-state distribution of photons across the medium is governed by the radiative transport equation (Fig. 2.2). Fast solution of these types of equations are prevalent in inverse problems like tomography, where fast solution of the forward problem (i.e., radiative transport) is very helpful for efficiently evaluating the inverse — that is, recovering μ_s from observations. In our framework of our setting, we can think of the scattering coefficient function $\mu_s(\boldsymbol{x})$ is the parameter $\omega \in \Omega_\infty$. Different choices of scattering coefficient $\mu_s(\boldsymbol{x})$ will produce different steady-state distributions of light $\mathsf{u}(\omega)$. Then, in the context of (2.1), the operator $\mathsf{L}(\omega)$ is the forward scattering operator, and $\mathsf{f}(\omega)$ represents sources and boundary conditions.

It is worth remarking that beyond the problems we discuss herein, there are many additional problems of this form across the scientific literature. Particularly noteworthy classes of problems include aforementioned uncertainty quantification problems and Bayesian inference problems, where one samples ω from a probability distribution with the goal to quantify statistics of the result $\mathsf{u}(\omega)$.

We note that the underlying sample space Ω_∞ is typically continuous with respect to ω, so in this paper we limit ourselves with a discrete subset Ω of Ω_∞, such as an ϵ-net where every point in Ω_∞ is close to a representative in Ω.

Approximate solutions to equations whose parameters come from outside of Ω can then be formed via interpolation. As a concrete example, Ω_∞ may be all possible deformations of the boundary of a domain, and Ω may be a finite subset set of representative deformations. Note we purposefully keep the elements of the set Ω_∞ vague as its elements may not be easily vectorizable.

Throughout this paper, we will represent Ω as

$$\Omega \equiv \{\omega_1, \omega_2, \ldots, \omega_p\}, \tag{2.2}$$

whose elements ω_i denote the representative samples for which we would like to solve the integral equation (2.1).

If the parameter ω wildly changes the underlying problem, then it is difficult to perform this task more efficiently then simply solving all of the equations (2.1). However, in many real-world contexts, the dependence on the parameter ω is such that the solutions $\mathsf{u}(\omega)$ form a space that is approximately low dimensional. In this case, the solutions $\mathsf{u}(\omega)$ can be well represented by a few appropriately chosen degrees of freedom. The goal of reduced basis methods is to extract these relevant degrees of freedom and use them to accelerate the computation of the solutions $\mathsf{u}(\omega)$.

Therefore, we ultimately want to find a small orthogonal basis matrix $\mathsf{Q} \in \mathbb{R}^{n \times n_{rb}}$, where n_{rb} is the dimension of our reduced basis, with $n_{rb} \ll n$. Ideally, we want the columns of Q to approximately capture the desired set of solutions

$$\mathsf{S} \equiv \begin{bmatrix} \mathsf{u}(\omega_1) & \mathsf{u}(\omega_2) & \ldots & \mathsf{u}(\omega_p) \end{bmatrix}. \tag{2.3}$$

When the basis matrix Q is fixed, one wants to find a good approximation to $\mathsf{u}(\omega)$ in the subspace spanned by Q. Naturally, the closest approximation to $\mathsf{u}(\omega)$ is given by the orthogonal projection $\mathsf{Q}\mathsf{Q}^{\mathsf{T}}\mathsf{u}(\omega)$. However, this is difficult to compute without also computing $\mathsf{u}(\omega)$. In practice, one often opts to compute a As is common in the model reduction literature, we will approximate $\mathsf{Q}^{\mathsf{T}}\mathsf{u}(\omega)$ by a reduced solution $\mathsf{v}(\omega) \in \mathbb{R}^{n_{rb}}$ and solve for $\mathsf{v}(\omega)$ by using a Galerkin projection,

$$[\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\omega)\mathsf{Q}]\mathsf{v}(\omega) = [\mathsf{Q}^{\mathsf{T}}\mathsf{f}(\omega)], \qquad \mathsf{v}(\omega) \approx \mathsf{Q}^{\mathsf{T}}\mathsf{u}(\omega) \tag{2.4}$$

Since the dimension n_{rb} is much less than the dimension n of the original system (2.1), the projected system (2.4) provides us with an inexpensive way of computing approximations to the solutions $\mathsf{u}(\omega)$. First, one solves for the quantity $\mathsf{v}(\omega)$ in the $n_{rb} \times n_{rb}$ projected system (2.4). Afterwards, applying the matrix Q to the result $\mathsf{v}(\omega)$ gives an approximation of the true solution $\mathsf{u}(\omega)$.

2.1. Main difficulties

In the procedure of solving (2.4), there are two practical difficulties which arise:

1. *Assembling the reduced basis* Q *efficiently. (Offline).* There are a number of existing methods for constructing the basis Q. Unfortunately, they are typically either computationally expensive or difficult to implement. One can perform a proper orthogonal decomposition (POD) of solutions to the full-order model to obtain such a basis Q [19]. However, obtaining a high quality Q typically requires solving a large number of full-order systems — which may be intractable. There are also greedy methods [8,26,25,37,38,7], which sequentially build up a reduced basis by repeatedly selecting the solution $\mathsf{u}(\omega_i)$ which would yield the greatest reduction in an approximate error estimator. Unfortunately, the implementation and construction of error estimators are very involved. Moreover, the inherently sequential nature of greedy selection algorithms means that they tend to be difficult to parallelize. Though it may be possible to compute candidates in parallel ahead of time, this may use considerably more CPU time than a straightforward serial implementation.
2. *Assembling the projected operator* $\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\omega)\mathsf{Q}$ *efficiently. (Online).* Since $\mathsf{L}(\omega)$ is an operator, it is usually too computationally expensive to assemble the whole operator $\mathsf{L}(\omega)$ explicitly. This necessitates that we design an inexpensive way of assembling the reduced operators $\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\omega)\mathsf{Q}$ without ever explicitly assembling or applying their full-order counterparts $\mathsf{L}(\omega)$.

2.2. Contribution

In this paper, we present a novel reduced basis approach to integral equations that has the benefit of being both general-purpose and easy to implement. Our contributions are twofold: we first present a novel method for *efficient selection of training samples*. We use this selection scheme to address the first issue of assembling the reduced basis Q efficiently. Next, we present a *simple interpolation technique for assembling reduced operators*. We use this technique to address the second difficultly of assembling the projected operator $\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\omega)\mathsf{Q}$ efficiently.

The combination of these two techniques forms the core of our *coarse-proxy* reduced basis method, diagrammed in full in Fig. 2.3. Our method provides a model order reduction framework for general linear integral equation problems that addresses both of the above issues without the aforementioned pitfalls of existing methods. In our numerical experiments, we apply our method to two examples, the radiative transport equation and the boundary integral formulation of the Laplace equation—and demonstrate that on both of these problems, our methods allow for significant improvements in performance over a naive solve of all elements of the parameter space.

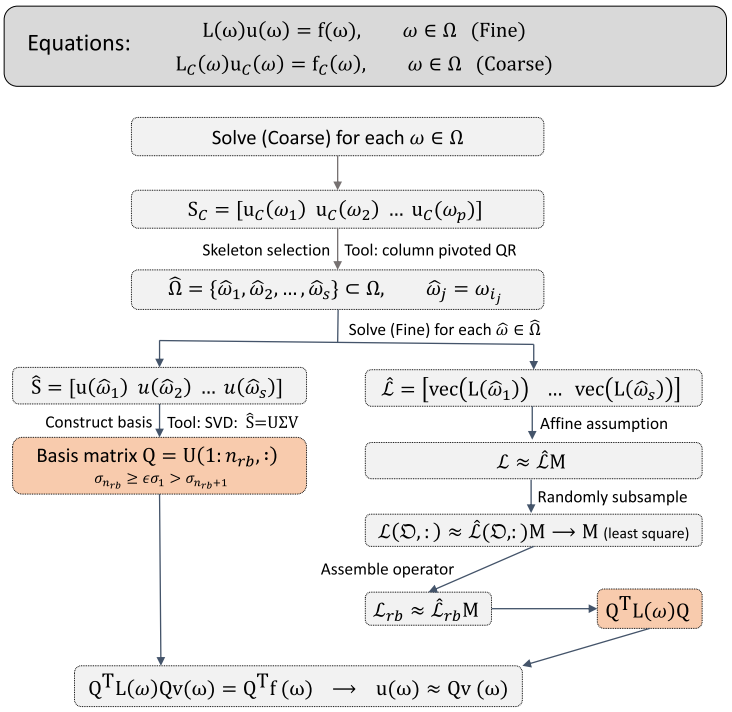


Fig. 2.3. Diagram of our proposed model reduction method for integral equations.

For the aforementioned *efficient selection of training samples*, we propose a novel method of constructing the reduced basis Q by leveraging a coarse-proxy model to identify a set of important parameters $\widehat{\omega}_1, \dots, \widehat{\omega}_s$ in the sample space Ω, where $s \ll p$. As an example, one can use an inexpensive low-resolution model to identify which parameters ω would be important in the construction of a reduced basis, and we only solve the full-order equations (2.1) for these important parameters. The method has the desirable property of being embarrassingly parallelizable.

For the aforementioned *interpolation technique for assembling reduced operators*, we propose assembling the operators $Q^T L(\omega)Q$ by levering random sampling. We draw random samples from the *dense* operators $L(\omega)$ and then use these samples to construct a linear combination of a small subset of precomputed basis operators $Q^T L(\widehat{\omega}_1)Q$, ..., $Q^T L(\widehat{\omega}_s)Q$ to approximately reconstruct $Q^T L(\omega)Q$. While similar matrix interpolation techniques exist in the literature (e.g., [9]), they are only tractable for *sparse* matrices where it is possible to explicitly assemble an entire matrix. For integral operators, this is not possible — hence, we design our interpolation technique so that the full operator never needs to be assembled explicitly.

The details of the proposed select and interpolation methods are discussed in Section 3, and numerical tests are presented in Section 4.

3. Framework details

Our framework for solving problems of the form in (2.1) is based on the idea of using an inexpensive coarse-proxy model to extract the important solutions from the solution set S. This model can be, for example, the original fine problem, but at a much lower resolution. Or, since integral equations are quite amenable to wavelets [39,15,2], one may use a sparse basis of wavelets and only solve for lower-order coefficients. Indeed, it is usually the case that very different solutions in S are quite distinguishable even at lower fidelity in the appropriate basis. It therefore makes sense to use only a coarse resolution solve to isolate a representative set of solutions and operators in S.

After selecting a coarse model, we write this coarse-proxy model as

$$L_C(\omega)u_C(\omega) = f_C(\omega), \tag{3.1}$$

where $L_C(\omega) \in \mathbb{R}^{n_C \times n_C}$ is the coarse analogue of the operator $L(\omega)$, and $u_C(\omega) \in \mathbb{R}^{n_C}$ and $f_C(\omega) \in \mathbb{R}^{n_C}$ are the coarse-proxy solution and coarse-proxy source term respectively. One should choose this coarse-proxy model so that it is inexpensive to evaluate (i.e., $n_C^2 \ll n^2$). Fortunately, as long as the solutions $u_C(\omega)$ of the coarse-proxy model can approximately capture the important features of their fine counterparts $u(\omega)$, the particular choice of coarse-proxy model is not especially relevant. However, one must still exercise the appropriate caution. For example, if the solutions $u(\omega)$ contain important high frequency content, one should not expect that solving the problem on a coarse grid will provide a good coarse-proxy model.

Notation. We use MATLAB notation to denote submatrices, i.e., if $\mathsf{M} \in \mathbb{R}^{n\times m}$, then for $A \subseteq \{1, ..., n\}$ and $B \subseteq \{1, ..., m\}$, $\mathsf{M}(A, B) \in \mathbb{R}^{|A|\times|B|}$ denotes the submatrix of M formed with rows A and columns B. In the case where either $A = \{1, ..., n\}$ or $B = \{1, ..., m\}$, we use the shorthand $\mathsf{M}(:, B) \in \mathbb{R}^{n\times|B|}$ or $\mathsf{M}(A, :) \in \mathbb{R}^{|A|\times m}$, respectively. The same notation also applies to vectors.

3.1. Skeleton extraction

To produce a reduced basis matrix Q, we select fine solution candidates $\mathsf{u}(\omega)$ that are important columns of the solution matrix S and construct Q via an SVD of those important columns. If the number of fine solution candidates in S is substantially smaller than the number of solutions in S, then this SVD operation is significantly less expensive than a traditional POD. However, retrieving a column of S still requires an expensive full-order solve. To determine the important columns of S without incurring this cost, we note that the coarse-proxy solutions $\mathsf{u}_C(\omega)$ can serve as a good proxy for their fine counterparts. That is, we can identify important columns of S by search for important solutions among their coarse proxies $\mathsf{u}_C(\omega)$. Thus, our initial step is to compute the entire set of coarse-proxy solutions (or alternatively, an appropriately subsampled version thereof), which we write in matrix form as

$$\mathsf{S}_C \equiv \begin{bmatrix} \mathsf{u}_C(\omega_1) & \mathsf{u}_C(\omega_2) & \dots & \mathsf{u}_C(\omega_p) \end{bmatrix}. \tag{3.2}$$

Note that this step is embarrassingly parallelizable. Once these solutions are ready, we identify the important elements of the sample space Ω via a column pivoted QR decomposition of S_C. This procedure returns a permutation π of the columns of S_C. Let the *skeleton indices* $\mathfrak{S}$ be the set of columns indices in π whose corresponding diagonal R_{ii} is less than a certain threshold ϵ of R_{11}. Let the parameters $\widehat{\omega}_i$ corresponding to these indices be denoted as the set of *skeleton parameters* $\widehat{\Omega} \subset \Omega$. These will be our approximation as to the important columns of S.

3.2. Skeleton extraction implementation

We give a concrete implementation of the skeleton extraction algorithm described above in Algorithm 1. This method takes in a sample space Ω and extracts the important skeleton parameters $\hat{\omega}_j$. It returns the set of indices $\mathfrak{S} = \{i_1, i_2, ..., i_s\}$ corresponding to the indices of these skeletons, i.e. $\hat{\omega}_j = \omega_{i_j}$. It is possible that the implementation can be better tailored to the problem, but we provide this algorithm as a general-purpose default.

Algorithm 1: GetSkeletons: Skeleton Extraction with a Coarse-Proxy Model (Offline).

Input: A sample space Ω.
Output: The indices $\mathfrak{S}$ of the important skeleton parameters.
/* Construct coarse-proxy solutions S_C. */
for ω_i *in* Ω **do**
  $\mathsf{L}_C(\omega_i) \leftarrow$ CoarseOperator(ω_i);
  $\mathsf{f}_C(\omega_i) \leftarrow$ CoarseSourceTerm(ω_i);
  $\mathsf{S}_C(:, i) \leftarrow \mathsf{L}_C(\omega_i)^{-1}\mathsf{f}_C(\omega_i)$
end
/* Perform column pivoted QR factorization on S_C and denote the column permutation of the CPQR factorization by ρ. */
$(\mathsf{Q}_C, \mathsf{R}_C, \rho) \leftarrow \mathrm{CPQR}(\mathsf{S}_C)$;
/* Select all important column indices ρ_i based on $\mathsf{R}_{C,ii}$. */
$\mathfrak{S} \leftarrow \rho(\{i \mid \mathsf{R}_{C,ii} \geq \epsilon \mathsf{R}_{C,11}\})$;
return $\mathfrak{S}$;

3.3. Reduced basis construction

Once we have selected the skeletons $\hat{\Omega}$, we calculate the corresponding solutions for the full-order model. We denote these corresponding fine solutions denoted as the *fine skeleton set*,

$$\widehat{\mathsf{S}} \equiv \mathsf{S}(:, \widehat{\Omega}) = \begin{bmatrix} \mathsf{u}(\widehat{\omega}_1) & \mathsf{u}(\widehat{\omega}_2) & \dots & \mathsf{u}(\widehat{\omega}_s)) \end{bmatrix}. \tag{3.3}$$

Note that this step is once again embarrassingly parallelizable.

To compute the reduced basis, we apply an SVD decomposition to the fine skeleton set $\widehat{\mathsf{S}}$ to obtain $\mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}} = \widehat{\mathsf{S}}$. To build a reduced basis, we crop U by discarding all columns with singular values σ_i less than $\epsilon\sigma_1$,

$$\mathsf{Q} \equiv \mathsf{U}(:, 1 : n_{rb}), \quad \text{with } n_{rb} \text{ such that } \sigma_{n_{rb}} \geq \epsilon\sigma_1 > \sigma_{n_{rb}+1}, \tag{3.4}$$

where ϵ is the same ϵ used in Algorithm 1.

We take a moment to note that the coarse-proxy model is only used to select the skeleton parameters and not for the actual construction of the reduced basis. Hence, it is sufficient for the coarse-proxy model to be good enough to capture the main features of the full-order model and for the important columns of the coarse-proxy solution matrix S_C to roughly correspond to the important columns of the fine solution matrix S.

3.4. Reduced operator construction

Once we have constructed the reduced basis Q, it remains to solve the projected problem

$$[\mathsf{Q}^{\mathrm{T}}\mathsf{L}(\omega)\mathsf{Q}]\,\mathsf{v}(\omega) = [\mathsf{Q}^{\mathrm{T}}\mathsf{f}(\omega)]\,, \quad \mathsf{u}(\omega) \approx \mathsf{Q}\,\mathsf{v}(\omega) \tag{3.5}$$

for arbitrary ω. As such, we require a fast method of assembling the projected operator $\mathsf{Q}^{\mathrm{T}}\mathsf{L}(\omega)\mathsf{Q}$. Assembling the full operator $\mathsf{L}(\omega)$ and then projecting it is prohibitively expensive. However, in solving for the fine solutions $\widehat{\mathsf{S}}$ in (3.3), we have already assembled a subset of the operators $\mathsf{L}(\omega)$. As we will see, the operators assembled during (3.3) can be used to construct arbitrary $\mathsf{Q}\mathsf{L}(\omega)\mathsf{Q}^{\mathrm{T}}$ via interpolation.

If the matrix of the vectorized fine operators $\mathsf{L}(\omega)$ is denoted by

$$\mathcal{L} \equiv \begin{bmatrix} \mathrm{vec}(\mathsf{L}(\omega_1)) & \mathrm{vec}(\mathsf{L}(\omega_2)) & \dots & \mathrm{vec}(\mathsf{L}(\omega_p)) \end{bmatrix}, \tag{3.6}$$

then, in the process of computing the fine skeleton set $\widehat{\mathsf{S}}$, we have already assembled a subset of the columns of $\mathcal{L}$, given by

$$\widehat{\mathcal{L}} \equiv \mathcal{L}(:, \widehat{\Omega}) = \begin{bmatrix} \mathrm{vec}(\mathsf{L}(\widehat{\omega}_1)) & \mathrm{vec}(\mathsf{L}(\widehat{\omega}_2)) & \dots & \mathrm{vec}(\mathsf{L}(\widehat{\omega}_s)) \end{bmatrix}. \tag{3.7}$$

Since the dependence of $\mathsf{L}(\omega)$ on the parameter ω is assumed to be low dimensional, it stands to reason that it should be possible to use the operators we've already constructed to somehow assemble arbitrary columns of the full set of projected operators,

$$\mathcal{L}_{rb} \equiv \begin{bmatrix} \mathrm{vec}(\mathsf{Q}^{\mathrm{T}}\mathsf{L}(\omega_1)\mathsf{Q}) & \mathrm{vec}(\mathsf{Q}^{\mathrm{T}}\mathsf{L}(\omega_2)\mathsf{Q}) & \dots & \mathrm{vec}(\mathsf{Q}^{\mathrm{T}}\mathsf{L}(\omega_p)\mathsf{Q}) \end{bmatrix}. \tag{3.8}$$

We propose a linear interpolation method based on random samples of the fine operators $\mathsf{L}(\omega)$.

To motivate our method, we first make an affine assumption. That is, we assume it is possible to assemble the $\mathsf{L}(\omega)$ by interpolating between the skeleton operators $\mathsf{L}(\widehat{\omega})$ in $\widehat{\mathcal{L}}$ as such,

$$\mathsf{L}(\omega_i) \approx \sum_{j=1}^{s} \mathsf{L}(\widehat{\omega}_j)\, m_{ji}. \tag{3.9}$$

It follows by linearity, that

$$\mathsf{Q}^{\mathrm{T}}\mathsf{L}(\omega_i)\mathsf{Q} \approx \sum_{j=1}^{s} \mathsf{Q}^{\mathrm{T}}\mathsf{L}(\widehat{\omega}_j)\mathsf{Q}\, m_{ji}. \tag{3.10}$$

Note that both (3.9) and (3.10) can be written in matrix form,

$$\mathcal{L} \approx \widehat{\mathcal{L}}\mathsf{M}, \tag{3.11}$$

$$\mathcal{L}_{rb} \approx \widehat{\mathcal{L}}_{rb}\mathsf{M}, \tag{3.12}$$

where $\widehat{\mathcal{L}}_{rb} \equiv \mathcal{L}_{rb}(:, \mathfrak{S})$ are the fine skeleton operators projected into the reduced basis space and the $\mathsf{M} = (m_{ji}) \in \mathbb{R}^{s\times p}$ is the *mixing matrix* of interpolation coefficients. However, we must now consider how to actually compute such a mixing matrix M.

Our answer is based on the observation that if one makes the affine assumption, then to recover the coefficients m_{ji}, it suffices to randomly subsample important parts of the operators (i.e., rows of $\mathcal{L}$) and use the resulting samples to perform least squares regression to obtain M. Let these important samples / row indices be denoted by $\mathfrak{O}$. The choice $\mathfrak{O}$ can be heavily dependent on the application. For example, if our operators are diagonally dominant, then it would make sense to include the diagonal of the fine operators in $\mathfrak{O}$. We can also select $\mathfrak{O}$ to be slices of the operator, which are cheap to construct, like a randomly chosen set of columns in the fine operators. Ideally, we should have $|\mathfrak{O}| \ll n_{rb}^2$.

Taking the rows corresponding to $\mathfrak{O}$ in the above (3.11) gives

$$\mathcal{L}(\mathfrak{O}, :) \approx \widehat{\mathcal{L}}(\mathfrak{O}, :)\mathsf{M}. \tag{3.13}$$

After computing $\mathcal{L}(\mathfrak{O}, :)$ for all fine operators, we then construct the mixing matrix M via least-squares regression on (3.13), i.e.,

$$\mathsf{M} = \underset{\mathsf{X}}{\operatorname{argmin}} \|\mathcal{L}(\mathfrak{O}, :) - \widehat{\mathcal{L}}(\mathfrak{O}, :)\mathsf{X}\|_F^2 . \tag{3.14}$$

Once M is constructed, we can assemble any projected operator $\mathsf{Q}\mathsf{L}(\omega)\mathsf{Q}^{\mathsf{T}}$ by performing the linear interpolation given by (3.10).

3.5. Reduced basis and mixing matrix construction implementation

Here, we provide an example implementation of both the construction of the reduced basis, as described in Section 3.3, as well as the construction of the mixing matrix described in Section 3.4. The pseudo-code for this example implementation is given in Algorithm 2. To use the algorithm, we require that the user implement the following primitives:

- FineSolve(ω_i): This method takes in the parameter ω_i and outputs the corresponding fine solution $\mathsf{u}(\omega_i)$ as well as the corresponding vectorized fine operator $\mathrm{vec}(\mathsf{L}(\omega_i))$. **Nota bene** that, in practice, it may be the case that $\mathrm{vec}(\mathsf{L}(\omega_i))$ may be too large to store in memory. This is not an obstacle, as we only use $\mathrm{vec}(\mathsf{L}(\omega_i))$ for notational convenience. To implement the following algorithms, one only needs to be able to apply the operator $\mathsf{L}(\omega_i)$ and to be able to sample a sparse subset of the entries of $\mathrm{vec}(\mathsf{L}(\omega_i))$.
- GetOperatorSamples(): This method chooses the set of operator entries $\mathfrak{O}$ (i.e., rows of the matrix $\mathcal{L}$) to sample and outputs the operator samples $\mathcal{L}(\mathfrak{O}, :)$, as described in Section 3.4.

Note that there is a part of the implementation which involves adding additional skeletons to the skeleton set. This segment of the algorithm will be addressed in Section 3.8.

Algorithm 2: Reduced Basis and Mixing Matrix Computation. (Offline).

Input: A sample space Ω
Output: A reduced basis matrix Q, a mixing matrix M, projections $\widehat{\mathcal{L}}_{rb}$ of fine skeleton operators into the reduced basis space.

```
/* Compute the important skeletons in the sample space                    */
```
$\mathfrak{S} \leftarrow$ GetSkeletons(Ω);
$\widehat{\Omega} \leftarrow \Omega(:, \mathfrak{S})$;
```
/* Compute the corresponding fine skeleton solutions                      */
```
for $\widehat{\omega}_j$ *in* $\widehat{\Omega}$ **do**
  $(\widehat{\mathsf{S}}(:, j), \widehat{\mathcal{L}}(:, j)) \leftarrow$ FineSolve($\widehat{\omega}_j$);

```
/* (Optional) Use additional skeleton extraction as described in Section 3.8 */
```
if *Using additional skeleton extraction* **then**
  $(\widehat{\mathsf{S}}(:, j), \widehat{\mathcal{L}}(:, j)) \leftarrow$ AdditionalSkeletons($\Omega, \widehat{\mathsf{S}}, \widehat{\mathcal{L}}, \mathfrak{S}, \mathcal{L}_{samp}$);

```
/* Construct reduced basis matrix Q from fine skeletons Ŝ by taking the first few left singular vectors
   of Ŝ.                                                                  */
```
$(\mathsf{U}, \Sigma, \mathsf{V}) \leftarrow \mathrm{SVD}(\widehat{\mathsf{S}})$;
$\mathsf{Q} \leftarrow \mathsf{U}(:, \Sigma > \epsilon\sigma_1)$;
```
/* Compute the samples 𝓛(𝔒, :) from each fine operators                  */
```
$\mathcal{L}_{samp} \leftarrow$ GetOperatorSamples();
```
/* Perform least squares regression using the samples 𝓛_samp to compute the mixing matrix M.  */
```
$\mathsf{M} \leftarrow$ LeastSquares($\mathcal{L}_{samp}, \mathcal{L}_{samp}(:, \mathfrak{S})$);
```
/* Project the skeleton operators 𝓛̂ into the reduced basis space given by Q. Note L(ω̂_j) has been reshaped
   into a matrix.                                                         */
```
for *vec*($\mathsf{L}(\widehat{\omega}_j)$) *in* $\widehat{\mathcal{L}}$ **do**
  $\widehat{\mathcal{L}}_{rb}(:, j) \leftarrow \mathrm{vec}(\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\widehat{\omega}_j)\mathsf{Q})$;

return $(\mathsf{Q}, \mathsf{M}, \widehat{\mathcal{L}}_{rb})$;

3.6. Online reduced basis solve implementation

In this subsection, we provide pseudo-code in Algorithm 3 for using the offline computations performed in Algorithms 1 and 2 to compute fast online approximations to $\mathsf{u}(\omega)$ for arbitrary $\omega \in \Omega$. We suppose that we are provided with the following primitive:

- AssembleRightHandSide(Q, ω_i): This method takes in the reduced basis Q and a parameter ω_i and returns $\mathsf{Q}^{\mathsf{T}}\mathsf{f}(\omega_i)$ or an approximation thereof. Depending on the problem being solved, there might be some intricacies to this. However, if $\mathsf{f}(\omega_i)$ is inexpensive to assemble, then the oracle can simply compute $\mathsf{f}(\omega_i)$ and apply Q^{T}. In other situations, one can use mathematical manipulations to obtain an expression for $\mathsf{f}(\omega_i)$ in terms of already computed expressions. See the radiative transport equation Section 4.2 for a nontrivial case. In the worst case, if the entries of $\mathsf{f}(\omega_i)$ are not overly expensive to sample, one can sub-sample the $\mathsf{f}(\omega_i)$ and use the samples to linearly interpolate between $\mathsf{Q}^{\mathsf{T}}\mathsf{f}(\widehat{\omega}_i)$ by constructing a mixing matrix using the technique in Section 3.4. A more involved sub-sampling alternative could be

Algorithm 3: Reduced Basis Solve for $\mathsf{u}(\omega)$ (Online).

Input: A sample $\omega \in \Omega$ for which to compute a reduced basis approximation, the mixing matrix M, the projected skeleton operators $\widehat{\mathcal{L}}_{rb}$, and the reduced basis matrix Q.
Output: An approximation u_{rb} of $\mathsf{u}(\omega)$.

```
/* Assemble our approximation for the projected operator L_rb ≡ Q^T L(ω) Q using the projected skeleton
   operators L̂_rb and the mixing matrix M.                                                          */
vec(L_rb) ← L̂_rb M(:, i);
/* Have the oracle assemble the right hand side of the equation, i.e., f_rb(ω) ≡ Q^T f(ω), for us and project it
   into the reduced basis space.                                                                    */
f_rb ← AssembleRightHandSide(Q, ω);
/* Solve the system and return the result.                                                          */
v ← L_rb^{-1} f_rb;
/* Lift result from reduced basis space to ℝ^n.                                                     */
u_rb ← Q v;
return u_rb;
```

to use a discrete empirical interpolation method such as Q-DEIM [10] to compute sub-sampling entries in $\mathsf{f}(\omega)$ and interpolation weights for $\mathsf{Q}^T\mathsf{f}(\omega)$.

3.7. A note on gappy matrix POD

We remark that the above method of constructing reduced operators is close to Gappy Matrix POD in [9]. However, one key distinction is that we do not orthogonalize the skeleton operators $\mathsf{Q}^T\mathsf{L}(\widehat{\omega}_i)\mathsf{Q}$. Gappy Matrix POD would involve vectorizing the skeleton operators $\mathsf{L}(\widehat{\omega}_i)$, taking SVD to find a set of orthogonalized operators $\mathsf{L}_1^{\perp}, ..., \mathsf{L}_r^{\perp}$, projecting them into the reduced basis space, and then using $\mathsf{Q}^T\mathsf{L}_1^{\perp}\mathsf{Q}, ...\mathsf{Q}^T\mathsf{L}_r^{\perp}\mathsf{Q}$ to interpolate the general projected operators $\mathsf{Q}^T\mathsf{L}(\omega_i)\mathsf{Q}$. We do not do this. This is intentional. While performing this orthogonalization may sometimes result in increased stability of interpolation, for integral operators, it is not desirable to represent the underlying operators as full dense matrices. Moreover, by virtue of how we select the skeleton operators, we ensure to some extent that the interpolation problem is already relatively well-conditioned.

3.8. Additional skeleton extraction

Sometimes, the fine operators $\widehat{\mathcal{L}}$ we assemble during our fine solves may not be sufficiently rich to reconstruct all of the operators in $\mathcal{L}_{rb}$ via interpolation. If this is the case, then we must add additional columns to our set of skeleton operators $\widehat{\mathcal{L}}$. Note that we can use the fine operator samples $\mathcal{L}(\mathfrak{O}, :)$ in the previous section to get a rough idea the important operators in $\mathcal{L}$. To find the operators we have failed to represent well with our choice of skeletons $\mathcal{L}$, we can consider the operator samples $\mathcal{L}(\mathfrak{O}, :)$ with our skeletons $\mathcal{L}(\mathfrak{O}, \mathfrak{S})$ projected out,

$$\mathcal{L}_{res} \equiv \mathcal{L}(\mathfrak{O}, :) - P\mathcal{L}(\mathfrak{O}, :), \tag{3.15}$$

where P is a projector onto the column space of $\mathcal{L}(\mathfrak{O}, \mathfrak{S})$. We call these the *residual operator samples*. This projection can be done via modified Gram-Schmidt, for example.

Then, before we compute the mixing matrix, we can perform a column pivoted QR decomposition of $\mathcal{L}_{res}$ to find operators we're unable to approximate well. Similar to what was done in Section 3.3, we select the columns with a diagonal R-factor which is smaller than $\eta\epsilon$ multiplied by the largest column norm in the unprojected $\mathcal{L}(\mathfrak{O}, :)$, where η is an arbitrary constant set by the user. Whatever columns $\mathfrak{A}$ are selected by this process, we append them to our set of fine operator skeletons $\widehat{\mathcal{L}}$ as such,

$$\widehat{\mathcal{L}} \leftarrow \begin{bmatrix} \widehat{\mathcal{L}} & \mathcal{L}(:, \mathfrak{A}) \end{bmatrix}. \tag{3.16}$$

In addition, depending on the problem at hand, one can also add the corresponding fine solutions of $\mathfrak{A}$ to the fine solution skeleton set $\widehat{\mathsf{S}}$, as these may add important fine scale information which our coarse-proxy model may have missed,

$$\widehat{\mathsf{S}} \leftarrow \begin{bmatrix} \widehat{\mathsf{S}} & \mathsf{S}(:, \mathfrak{A}) \end{bmatrix}. \tag{3.17}$$

Afterwards, one can continue with everything detailed in Section 3.4 without any changes, using (3.16) instead of (3.7) for the skeleton operators $\mathcal{L}$.

3.9. Implementation of additional skeleton extraction

We now provide a pseudo-code implementation in Algorithm 4 of the additional skeleton extraction algorithm presented above in Section 3.8.

Algorithm 4: AdditionalSkeletons: Optional Additional Skeleton Extraction (Offline).

Input: A sample space Ω, a set of fine skeletons $\widehat{\mathsf{S}}$, their corresponding operator skeletons $\widehat{\mathcal{L}}$ and indices $\mathfrak{S}$, and a matrix of operator samples $\mathcal{L}_{samp}$.
Output: A possibly enlarged set of operator skeletons $\widehat{\mathcal{L}}$ and fine solution skeletons $\widehat{\mathsf{S}}$.
/* Compute the maximum energy in the operator samples before we project out the fine skeletons. */
$a \leftarrow \max_i \|\mathcal{L}_{samp}(:, i)\|_2$;
/* Project out the fine skeletons $\mathcal{L}_{samp}(:, \mathfrak{S})$ we've computed in algorithm 1 from the samples $\mathcal{L}_{samp}$. */
$\mathcal{L}_{res} \leftarrow \text{ProjectOut}(\mathcal{L}_{samp}, \mathcal{L}_{samp}(:, \mathfrak{S}))$;
/* Extract important operators we've missed during reduced basis extraction via QR decomposition of residual samples $\mathcal{L}_{res}$. */
$(\mathsf{Q}, R, \rho) \leftarrow \text{CPQR}(\mathcal{L}_{res})$;
/* Select only column indices for which $r_{ii} \geq \eta \epsilon a$. In practice, this should be done by stopping the above QR factorization when this first happens. */
$\mathfrak{A} \leftarrow \rho(r_{ii} \geq \eta \epsilon a)$;
/* Compute new set of additional fine operators $\widehat{\mathcal{L}}_A$ and additional fine solutions $\widehat{\mathsf{S}}_A$ for the selected columns $\mathfrak{A}$. */
for ω'_j *in* $\Omega(:, \mathfrak{A})$ **do**
  $(\widehat{\mathsf{S}}_A(:, j), \widehat{\mathcal{L}}_A(:, j)) \leftarrow \text{FineSolve}(\omega'_j)$;
end
/* Add additional skeleton solutions $\widehat{\mathsf{S}}_A$ to our existing skeleton solutions $\widehat{\mathsf{S}}$. */
$\widehat{\mathsf{S}} \leftarrow [\widehat{\mathsf{S}} \;\; \widehat{\mathsf{S}}_A]$;
/* Append new skeletons $\widehat{\mathcal{L}}_A$ to our existing skeletons $\widehat{\mathcal{L}}$ */
$\widehat{\mathcal{L}} \leftarrow [\widehat{\mathcal{L}} \;\; \widehat{\mathcal{L}}_A]$;
return $(\widehat{\mathsf{S}}, \widehat{\mathcal{L}})$;

3.10. Interpolating operators with an offset

There are many problems in which the operators $\mathsf{L}(\omega)$ take on a natural form,

$$\mathsf{L}(\omega) = \mathsf{A} + \mathsf{B}(\omega), \tag{3.18}$$

where A does not depend explicitly on ω, and is shared among all of the operators $\mathsf{L}(\omega)$. We will see such examples of this later. In such situations, it may be more advisable to interpolate the operator $\mathsf{B}(\omega)$ instead of the full operator $\mathsf{L}(\omega)$ when constructing reduced operators. All techniques from Section 3.4 carry over with minimal modification. One assumes that there exist interpolation coefficients for the operators $\mathsf{L}(\omega)$,

$$\mathsf{B}(\omega_i) \approx \sum_j \alpha_{ij} \mathsf{B}(\widehat{\omega}_j). \tag{3.19}$$

Then one can find a mixing matrix M with

$$\mathcal{B}_{rb} \approx \widehat{\mathcal{B}}_{rb} \mathsf{M}, \tag{3.20}$$

by simply performing a least squares solve of the equation

$$\mathcal{B}(\mathfrak{O}, :) \approx \widehat{\mathcal{B}}(\mathfrak{O}, :) \mathsf{M}, \tag{3.21}$$

where $\mathcal{B}$, $\widehat{\mathcal{B}}$, $\mathcal{B}_{rb}$, $\widehat{\mathcal{B}}_{rb}$ are defined analogously to $\mathcal{L}$, $\widehat{\mathcal{L}}$, $\mathcal{L}_{rb}$, and $\widehat{\mathcal{L}}_{rb}$ in Section 3.4.

Once the mixing matrix M has been computed, note that the corresponding coefficients m_{ji} can then be used to interpolate the reduced operators $\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\omega_i)\mathsf{Q}$,

$$\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\omega_i)\mathsf{Q} \approx \mathsf{Q}^{\mathsf{T}}\mathsf{A}\mathsf{Q} + \sum_{j=1}^{s} \mathsf{Q}^{\mathsf{T}}\mathsf{B}(\widehat{\omega}_j)\mathsf{Q} m_{ji}. \tag{3.22}$$

The quantity $\mathsf{Q}^{\mathsf{T}}\mathsf{A}\mathsf{Q}$ can be computed alongside the skeleton operators $\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\widehat{\omega}_j)\mathsf{Q}$ when projecting operators into the reduced basis space.

4. Numerical results

To demonstrate that our framework is both a practical and efficient approach to model order reduction for integral equations, we perform simulations on the two following examples.

4.1. Boundary integral formulation of the Laplace equation

We consider the Laplace equation

$$\begin{aligned} \Delta\varphi &= 0, \quad \text{in } \mathcal{D}, \\ \varphi &= f, \quad \text{on } \partial\mathcal{D}, \end{aligned} \tag{4.1}$$

where $\mathcal{D} \subset \mathbb{R}^2$ is a bounded Lipschitz domain. Introduce the single layer potential u, which is given by the solution to the integral equation

$$f(\boldsymbol{x}) = \frac{1}{2}u(\boldsymbol{x}) - \int_{\partial\mathcal{D}} \frac{\partial G(\boldsymbol{x}, \boldsymbol{y})}{\partial n(\boldsymbol{y})} u(\boldsymbol{y})\, \mathrm{d}s(\boldsymbol{y}), \tag{4.2}$$

where G denotes the Green's function of the Laplace equation in 2D,

$$G(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2\pi} \ln \frac{1}{|\boldsymbol{x} - \boldsymbol{y}|}. \tag{4.3}$$

Then once u has been computed by solving the integral equation (4.2), the solution ϕ to Laplace equation can be recovered by

$$\varphi(\boldsymbol{x}) = -\int_{\partial\mathcal{D}} \frac{\partial G(\boldsymbol{x}, \boldsymbol{y})}{\partial n(\boldsymbol{y})} u(\boldsymbol{y})\, \mathrm{d}s(\boldsymbol{y}). \tag{4.4}$$

Hence, the key of this problem is to solve the boundary integral equation (4.2).

To bring this problem in the many-query setting of our framework, we suppose that the shape of the domain $\mathcal{D}(\omega)$ is parameterized by ω taken from a sample space Ω_∞. Thus, the integral equations we would like to solve are given by

$$f(\boldsymbol{x}) = \frac{1}{2}u(\boldsymbol{x}; \omega) - \int_{\partial\mathcal{D}(\omega)} \frac{\partial G(\boldsymbol{x}, \boldsymbol{y})}{\partial n(\boldsymbol{y})} u(\boldsymbol{y}; \omega)\, \mathrm{d}s(\boldsymbol{y}), \tag{4.5}$$

where $f(\boldsymbol{x})$ is a function prescribed on $\mathbb{R}^2$, which we hold constant across all problem instances in Ω_∞. One can discretize this equation by taking a discrete number of samples of u on $\partial\mathcal{D}$, and using the trapezoidal rule to evaluate the integral above. An example oracle for this problem is one that uses a significantly reduced number of quadrature points on $\partial\mathcal{D}$ as its coarse-proxy model. Note that using a coarse-proxy model with low resolution becomes difficult when the source function f exhibits singular behavior near the boundary $\partial\mathcal{D}$, as the high frequency content in u is difficult to resolve. Nonetheless, even if such a coarse-proxy model is used for skeleton extraction, our results suggest that using the additional skeleton extraction techniques discussed in Section 3.8 will compensate for the information which the coarse-proxy model cannot resolve, since this high-frequency information will present itself in the operator samples.

Regardless of the exact implementation of the oracle's scheme for extracting skeletons, the discretized equation (4.5) reads

$$\mathsf{L}(\omega)\mathsf{u}(\omega) = \mathsf{f}(\omega), \tag{4.6}$$

where $\mathsf{u}(\omega)$ is the discretized version of the double layer potential u, $\mathsf{f}(\omega)$ is the source term f sampled on $\partial\mathcal{D}(\omega)$. And $\mathsf{L}(\omega)$ is a matrix that has the form

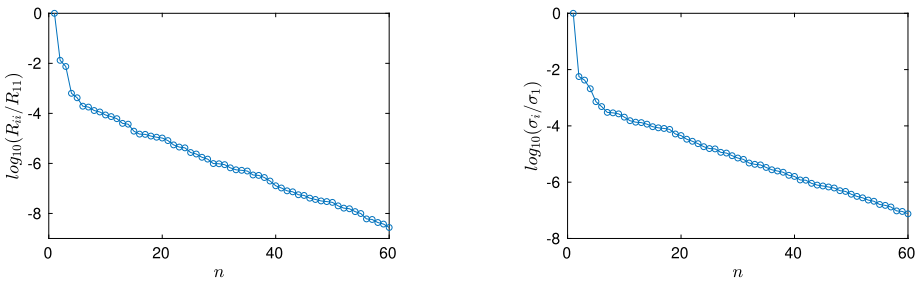
$$\mathsf{L}(\omega) = \frac{1}{2}\mathsf{I} - \mathsf{G}(\omega), \tag{4.7}$$

where I is the identity, and $\mathsf{G}(\omega)$ is the discretized integral kernel in (4.5).

After skeleton extraction and construction of the reduced basis Q, the projected equations in the reduced basis are given by

$$\left(\frac{1}{2}\mathsf{I} - \mathsf{Q}^{\mathsf{T}}\mathsf{G}(\omega)\mathsf{Q}\right)\mathsf{v}(\omega) = \mathsf{Q}^{\mathsf{T}}\mathsf{f}(\omega) \quad \mathsf{u}(\omega) \approx \mathsf{Q}\,\mathsf{v}(\omega) \tag{4.8}$$

One can construct the reduced operators $\mathsf{Q}^{\mathsf{T}}\mathsf{G}(\omega)\mathsf{Q}$ via the techniques discussed in Section 3.4. For our operator samples, we sample a few columns of all operators $\mathsf{G}(\omega)$ as well as the diagonals. In general, $\mathsf{f}(\omega)$ is inexpensive to assemble, so one can simply construct the source term $\mathsf{f}(\omega)$ and project it into the reduced basis space by applying Q^{T} during an online reduced basis solve.



(a) Diagonal of R (b) Sigular value σ_k

Fig. 4.1. Left: the normalized diagonal values of R in the skeleton selection for the described Laplace equation problem. Right: the normalized singular values in the basis construction for the described Laplace equation problem.

4.1.1. Results

We parameterize the boundary $\partial\mathcal{D}(\omega)$ by a polar curve $\gamma(\theta;\omega):\mathbb{R}\times\Omega\longrightarrow\mathbb{R}^2$, where $\theta\in[0,2\pi)$ and

$$\gamma(\theta;\omega)\equiv r(\theta;\omega)\,[\cos(\theta),\sin(\theta)]^T\,, \tag{4.9}$$

where $r(\theta;\omega):\mathbb{R}\times\Omega\longrightarrow\mathbb{R}$ is the radial distance of this curve from the origin at angle θ. To specify the radial function, we chose a set of interpolation points $\theta=2\pi k/N$ for $k\in\{0,\ldots,N-1\}$ and require that

$$r(2\pi k/N;\omega)=b_k(\omega) \tag{4.10}$$

where $b_k(\omega)$ are interpolation value for the radial function $r(\theta;\omega)$ at the interpolation points $\theta=2\pi k/N$. We then determine remainder of the curve γ by Fourier interpolation. Viewing the continuous parameter set Ω_∞ as a probability space, we take the radial interpolation points $b_k(\omega)$ to be i.i.d. uniformly random in the interval,

$$b_k(\omega)\sim\mathcal{U}([1-\kappa,1+\kappa]). \tag{4.11}$$

The reader may see examples of such curves as defined here in either Fig. 4.3 or Fig. 2.1. Finally, to make the problem challenging, we take the source function $f(\boldsymbol{x})$ to have a singularity which can potentially be situated near the curve γ. In particular, we take

$$f(\boldsymbol{x})\equiv\frac{1}{\|\boldsymbol{x}-\boldsymbol{x}_0\|_2}. \tag{4.12}$$

To ensure there are problem instances where the curve γ comes close to the singularity located at $\boldsymbol{y}$ we take our parameters to be

$$\kappa=0.4,\qquad \boldsymbol{x}_0=(0.6,0),\qquad N=8. \tag{4.13}$$

Finally, to extract a discrete parameter space $\Omega\subset\Omega_\infty$, we draw $|\Omega|=32768$ random samples from Ω_∞.

For our full-order model, we use a total of $n_f=2048$ quadrature samples. Whereas, for our coarse-proxy model, we use a total of $n_c=128$ quadrature samples. For selecting additional skeletons, we use a selection threshold multiplier of $\eta=1.5$. We sample a total of 10 columns randomly plus the diagonal.

Plot (a) in Fig. 4.1 presents the diagonal values of R in Algorithm 1 for this numerical experiment. As indicated in Algorithm 1, we compute these diagonal values from the coarse-proxy model solutions and use them to select skeleton parameters. Plot (b) in Fig. 4.1 shows the values of the singular values σ_k in Algorithm 2, which are used to construct the reduced basis. One sees that, if the threshold ϵ satisfies $\epsilon<1\times10^{-4}$, then $\log_{10}(R_{ii}/R_{11})$ and $\log_{10}(\sigma_k/\sigma_1)$ decay almost linearly. We also provide a visualization of the first 36 reduced basis (with respect to σ_k) in Fig. 4.2. Note that, as the singular values σ_k decay, the corresponding basis vectors contain increasingly more high frequency information. For further evaluation, we provide Fig. 4.3, a side-by-side comparison of our reduced basis method for $\epsilon=1\times10^{-6}$ with the underlying full-order model on three different parameter instances. We note that our method provides an accurate approximation of the solution to the Laplace problem.

To quantitatively demonstrate the performance gains of our reduced-order model over the full-order model, as well as the reduced-order model's validity for different thresholds, we evaluated both the full-order and reduced-order solutions for each element of Ω. The resulting relative L^2 error $\|\mathsf{u}(\omega)-\mathsf{u}_{rb}(\omega)\|_2/\|\mathsf{u}(\omega)\|_2$ and the runtime are all recorded in Table 4.1. When evaluating all solutions in bulk, our reduced basis method provides between a ten-fold and twenty-fold performance increase over the full-order model for a wide range of accuracy targets between .1% and 10% relative L^2 error, as recorded in (4.6). Note, crucially, that this figure *includes the expensive offline phase of our reduced basis method*. Moreover, if one halves the threshold ϵ, the average relative L^2 shrinks by an approximate factor of two. Therefore, this method approximately exhibits linear convergence with respect to the threshold. This fact is further illustrated in Fig. 4.4.

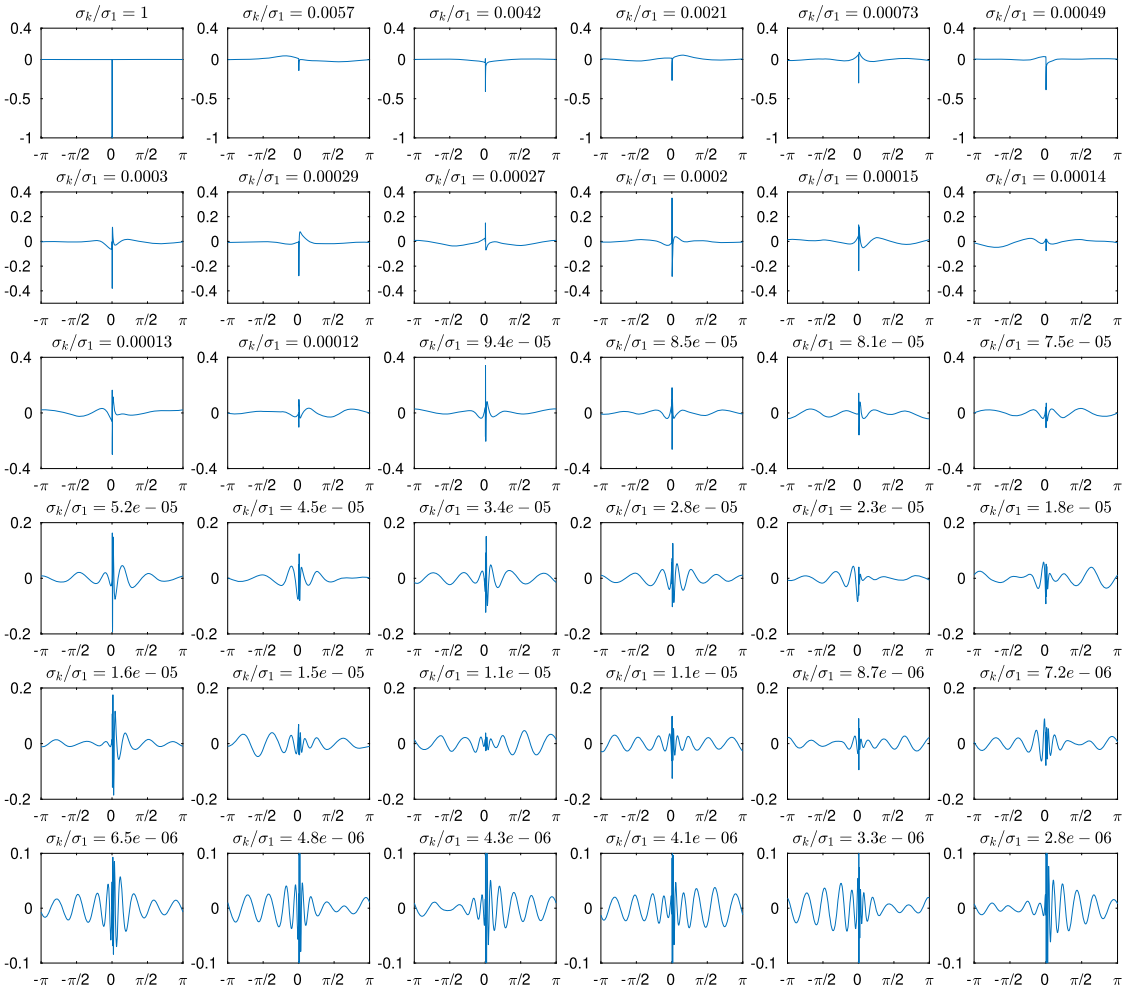


Fig. 4.2. Profiles of the reduced basis generated for the described boundary integral form of the Laplace equation problem for different singular values σ_k.

We also note that the offline stage for $\epsilon = 5 \times 10^{-6}$ in Table 4.1 takes less time than the offline stage for $\epsilon = 1 \times 10^{-5}$, which is because proportionally fewer skeletons are selected during the additional skeleton extraction phase.

4.2. Radiative transport equation with isotropic scattering

We consider the steady state radiative transport equation with the form

$$\begin{aligned} \boldsymbol{v} \cdot \nabla_{\boldsymbol{x}} \Phi(\boldsymbol{x}, \boldsymbol{v}) + \mu_t(\boldsymbol{x}) \Phi(\boldsymbol{x}, \boldsymbol{v}) &= \frac{\mu_s(\boldsymbol{x})}{2\pi} \int_{\mathbb{S}^1} \Phi(\boldsymbol{x}, \boldsymbol{v}') \, \mathrm{d}\boldsymbol{v}' + g(\boldsymbol{x}), \text{ in } \mathcal{D} \times \mathbb{S}^1, \\ \Phi(\boldsymbol{x}, \boldsymbol{v}) &= 0, \text{ on } \Gamma_-, \end{aligned} \tag{4.14}$$

where $\Phi(\boldsymbol{x}, \boldsymbol{v})$ denotes the photon flux at spatial position $\boldsymbol{x} \in \mathbb{R}^2$ in direction $\boldsymbol{v} \in \mathbb{S}^1$, and $g(\boldsymbol{x})$ is the light source. $\mathcal{D} \subset \mathbb{R}^2$ is the problem domain, $\mathbb{S}^1$ is the unit sphere in $\mathbb{R}^2$. Γ_- is the inward facing problem boundary, given by

$$\Gamma_- \equiv \{(\boldsymbol{x}, \boldsymbol{v}) \in \partial\mathcal{D} \times \mathbb{S}^1 \mid n(\boldsymbol{x}) \cdot v < 0\}, \tag{4.15}$$

where $n(x)$ is the normal of domain $\mathcal{D}$ at position $\boldsymbol{x}$. The boundary condition in (4.2) enforces that no light is entering the domain of interest. The transport coefficient $\mu_t(\boldsymbol{x})$ measures the total absorption at $\boldsymbol{x}$, which results from both physical absorption as well as from scattering, the latter of which is quantified by the scattering coefficient $\mu_s(\boldsymbol{x})$.

In this scenario, our quantity of interest is the local mean density $m(\boldsymbol{x})$ defined as

$$m(\boldsymbol{x}) \equiv \frac{1}{2\pi} \int_{\mathbb{S}^1} \Phi(\boldsymbol{x}, \boldsymbol{v}') \, \mathrm{d}\boldsymbol{v}'. \tag{4.16}$$

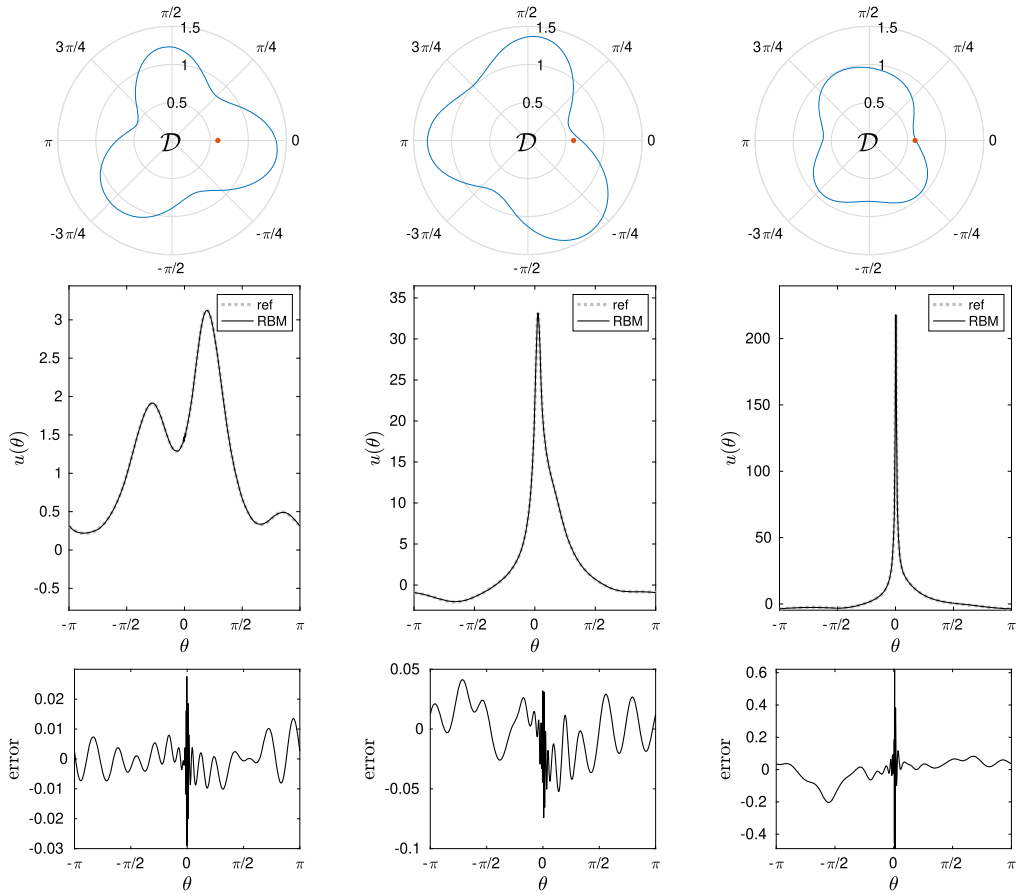
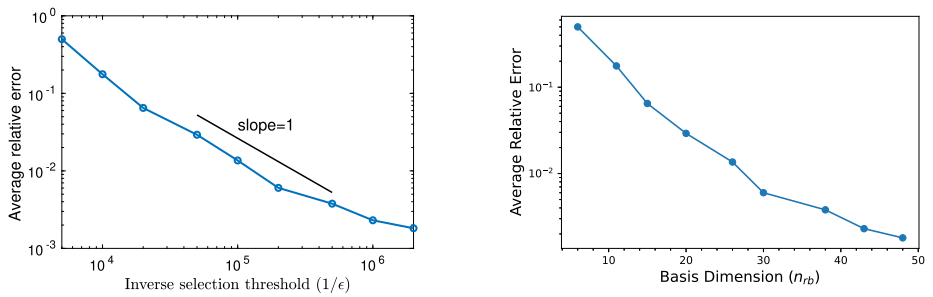


Fig. 4.3. Three examples of the solutions evaluated by the reduced basis method (RBM) for the single layer potential $u(\theta;\omega)$ for the parameter set Ω described for the boundary integral form of the Laplace equation and their corresponding reference solution (ref) with threshold $\epsilon = 1 \times 10^{-6}$. The upper figures are the domain $\mathcal{D}$ and the red point is the location of the singularity in (4.13).



(a) Selection Threshold vs. Error (b) Basis Dimension vs. Error

Fig. 4.4. Left: A log-log convergence plot of our method on the Laplace equation example, showing the average error $\langle E \rangle$ over the parameter set Ω against the inverse of the selection threshold ϵ. Right: The same, except with basis dimension on the x-axis.

As studied in [35,11], one can reformulate the differential equation (4.14) into an integral equation using the method of characteristics. This transformation yields the integral equation

$$\left[\frac{1}{\mu_s(\boldsymbol{x})} - \mathcal{K}\right] u(\boldsymbol{x}) = \mathcal{K} g(\boldsymbol{x}) \quad \text{with} \quad \mathcal{K}\phi(\boldsymbol{x}) \equiv \int_{\mathcal{D}} K(\boldsymbol{x}, \boldsymbol{y})\phi(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y}, \tag{4.17}$$

Table 4.1
Test results for our reduced basis method on the described Laplace equation problem. Here ϵ denotes the selection threshold used for reduced basis construction, s denotes the number of skeletons selected by our method (i.e., the number of fine solves used to construct the reduced basis), n_{rb} denotes the dimension of the reduced basis constructed, $T_{rb}^{(offline)}$ denotes the amount of time in seconds used in reduced basis construction, $T_{rb}^{(online)}$ denotes the amount of time in seconds used to solve all problem instances from Ω using our method once the reduced basis has been constructed, $T_{rb} = T_{rb}^{(offline)} + T_{rb}^{(online)}$ denotes the total amount of time in seconds used by our method to compute approximations to all problem instances in Ω, and T_{fine}/T_{rb} denotes the ratio between the time taken by our method to compute approximate solutions to all problem instances in Ω and the time T_{fine} taken to naively compute all exact fine solutions in Ω, i.e., the computational speed-up our algorithm provides. Finally, L^2 error denotes the average relative L^2 error, i.e., $\|\mathsf{u}(\omega) - \mathsf{u}_{rb}(\omega)\|_2 / \|\mathsf{u}(\omega)\|_2$ averaged over the parameter set Ω.

ϵ	s	n_{rb}	$T_{rb}^{(offline)}$	$T_{rb}^{(online)}$	T_{fine}	T_{fine}/T_{rb}	L^2 error
2×10^{-4}	52	6	157 s	4.00 s	4380 s	27.2×	0.4997
1×10^{-4}	67	11	186 s	6.09 s	4380 s	22.8×	0.1764
5×10^{-5}	77	15	206 s	6.48 s	4380 s	20.6×	0.0648
2×10^{-5}	98	20	252 s	7.51 s	4380 s	16.9×	0.0292
1×10^{-5}	115	26	283 s	8.92 s	4380 s	15.0×	0.0136
5×10^{-6}	132	30	229 s	10.03 s	4380 s	18.3×	0.0060
2×10^{-6}	150	38	254 s	12.59 s	4380 s	16.4×	0.0038
1×10^{-6}	179	43	295 s	15.77 s	4380 s	14.1×	0.0023
5×10^{-7}	194	48	315 s	18.11 s	4380 s	13.2×	0.0018

where $u(\boldsymbol{x}) = \mu_s(\boldsymbol{x})m(\boldsymbol{x})$ the integral kernel $K(\boldsymbol{x}, \boldsymbol{y})$ of the operator $\mathcal{K}$ is given by

$$K(\boldsymbol{x}, \boldsymbol{y}) \equiv \frac{1}{|\mathbb{S}^1|} \frac{1}{|\boldsymbol{x} - \boldsymbol{y}|} \exp\left(-|\boldsymbol{x} - \boldsymbol{y}| \int_0^1 \mu_t(\boldsymbol{x} - \tau(\boldsymbol{x} - \boldsymbol{y}))\, \mathrm{d}\tau \right). \tag{4.18}$$

To bring this problem into the many-query setting of our reduced basis framework, we now suppose that the scattering and transmission coefficients μ_s and μ_t have an explicit dependence on a parameter ω taken from some sample space Ω_∞. Henceforth, we will therefore write them as $\mu_s(\boldsymbol{x}; \omega)$ and $\mu_t(\boldsymbol{x}; \omega)$. Here, ω can encode small fluctuations or uncertainties about the underlying medium that the light propagates through. Making the dependence on the parameter ω explicit in (4.17) gives us the set of integral equations to solve,

$$[\mathcal{I} - \mu_s(\boldsymbol{x}; \omega)\mathcal{K}(\omega)]\, u(\boldsymbol{x}; \omega) = \mu_s(\boldsymbol{x}; \omega)\mathcal{K}(\omega) g(\boldsymbol{x}), \tag{4.19}$$

where $\mathcal{I}$ is the identity operator.

To discretize the above equation, we use a collocation method combined with Gauss-Legendre quadrature, as outlined in [11]. This discretization gives us the linear system

$$\mathsf{L}(\omega)\mathsf{u}(\omega) = \mathsf{f}(\omega), \tag{4.20}$$

where $\mathsf{L}(\omega)$ and $\mathsf{f}(\omega)$ have the forms

$$\mathsf{L}(\omega) = \mathsf{I} + \mathsf{B}(\omega), \qquad \mathsf{f}(\omega) \equiv -\mathsf{B}(\omega)\mathsf{g}, \tag{4.21}$$

and I, $\mathsf{B}(\omega)$, and g are the discretized versions of $\mathcal{I}$, $-\mu_s(\boldsymbol{x}; \omega)\mathcal{K}(\omega)$, and $g(\boldsymbol{x})$ respectively.

The application of our framework to this problem is now straightforward. To solve the full-order model, we use hierarchical interpolative factorization [23]. For our coarse-proxy model, we simply use significantly fewer collocation points in $\mathcal{D}$. We then use the method described in Sections 3.1 and 3.3 to compute a suitable reduced basis matrix Q for this problem in the offline stage.

In the remainder of the offline stage, we use the method described in Sections 3.4 and 3.10 to sample $\mathsf{B}(\omega)$ and construct a mixing matrix M such that

$$\mathcal{B}_{rb} \approx \widehat{\mathcal{B}}_{rb}\mathsf{M}, \tag{4.22}$$

where once again we define $\mathcal{B}_{rb}$ to be the matrix vectorized reduced operators,

$$\mathcal{B}_{rb} \equiv \left[\text{vec}(\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega_1)\mathsf{Q}) \quad \text{vec}(\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega_2)\mathsf{Q}) \quad \dots \quad \text{vec}(\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega_n)\mathsf{Q}) \right], \tag{4.23}$$

and $\widehat{\mathcal{B}}_{rb} \equiv \mathcal{B}_{rb}(:, \mathfrak{S})$ are our reduced operator skeletons. For the samples $\mathfrak{O}$ in the computation of M, we use ten randomly selected columns as well as the diagonal.

With the offline stage complete, we now switch focus to the online stage. To solve the desired equation

$$(\mathsf{I} + \mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega)\mathsf{Q})\mathsf{v}(\omega) = -\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega)\mathsf{g}, \quad \mathsf{u}(\omega) \approx \mathsf{Q}\,\mathsf{v}(\omega), \tag{4.24}$$

we can assemble the reduced operator $\mathsf{I} + \mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega)\mathsf{Q}$ from the reduced operator skeletons $\mathcal{B}_{rb}$ by using (4.22). Consider the matrix

$$\mathcal{F}_{rb} \equiv \left[-\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega_1)\mathsf{g} \quad -\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega_2)\mathsf{g} \quad \dots \quad -\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega_n)\mathsf{g} \right], \tag{4.25}$$

and note that the interpolation weights computed for the reduced operators $\mathsf{Q}^{\mathsf{T}}\mathsf{B}(\omega)\mathsf{Q}$ carry over to this matrix. That is,

$$\mathcal{F}_{rb} \approx \widehat{\mathcal{F}}_{rb}\mathsf{M}, \tag{4.26}$$

where $\widehat{\mathcal{F}}_{rb} \equiv \mathcal{F}(:, \mathfrak{S})$. Since we must assemble the quantities $-\mathsf{B}(\widehat{\omega}_i)\mathsf{g}$ during the computation of fine solutions for our reduced basis regardless, computing the matrix $\widehat{\mathcal{F}}_{rb}$ is fairly inexpensive, and only involves applying the matrix Q^{T} to the vectors $-\mathsf{B}(\widehat{\omega}_i)\mathsf{g}$. This means, if we compute $\widehat{\mathcal{F}}_{rb}$ during the offline stage, we then have an inexpensive method of assembling the right hand side, regardless of the fact that the expression involves the operator $\mathsf{B}(\omega)$.

4.2.1. Results

As a test case for the above example, we consider the domain $\mathcal{D} \equiv [0, 1]^2$. We let $\mu_s(\boldsymbol{x}; \omega)$ and $\mu_t(\boldsymbol{x}; \omega)$ be Gaussians with varying centers and widths,

$$\mu_t(\boldsymbol{x}; \omega) \equiv \mu_s(\boldsymbol{x}; \omega) \equiv 1 + A_\omega \exp(-((x_1 - c_{1\omega})^2 - (x_2 - c_{2\omega})^2)/\theta_\omega^2), \tag{4.27}$$

where the parameters $\omega \in \Omega$ have the form

$$\omega \equiv \left[A_\omega \quad c_{1\omega} \quad c_{2\omega} \quad \theta_\omega \right]. \tag{4.28}$$

We take the source term $g(\boldsymbol{x})$ to be

$$g(\boldsymbol{x}) \equiv \exp(-256((x_1 - 0.5)^2 + (x_2 - 0.5)^2)). \tag{4.29}$$

To build the parameter space Ω we vary both the width and the location of the Gaussian ensemble above. Let $\Omega_{A,\theta,N}$ be defined as

$$\Omega_{A,\theta,N} \equiv \left\{ \left[A \quad i/N \quad j/N \quad \theta \right] \mid i, j = 0, ..., N \right\}, \tag{4.30}$$

that is, parameters for Gaussians with width θ and amplitude A centered at grid points $(i/N, j/N)$. Take our parameter space Ω to consist of these Gaussians with three different widths/amplitudes,

$$\Omega \equiv \bigcup_{A \in \mathcal{A}} \bigcup_{\theta \in \Theta} \Omega_{A,\theta,N}, \tag{4.31}$$

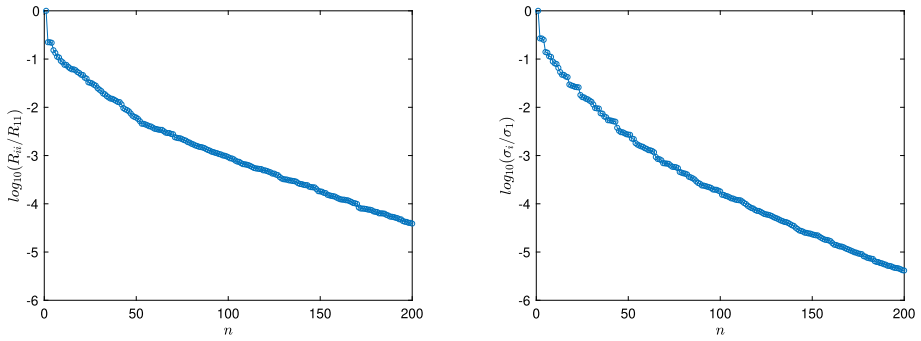
where

$$\mathcal{A} \equiv \{2, 4, 6, 8, 10\}, \qquad \Theta \equiv \{0.2, 0.3, 0.4, 0.5, 0.6\}, \quad N = 20.$$

This gives a total parameter space size of $|\Omega| = 11025$. Our full-order model is the model described in Section 4.2, with a grid size of $n_f \times n_f$ where $n_f = 128$. We use the algorithm described in this paper to build a reduced basis for this model and approximate true solutions. For our coarse-proxy model with the same model described in Section 4.2 but with a grid size of $n_c \times n_c$ instead, where $n_c = 32$. We use the procedure described in Section 3.8 to add additional skeletons to our skeleton set when operator samples cannot be well-represented using a linear combination the operator samples of the selected skeletons. For selecting additional skeletons, we use a selection threshold multiplier of $\eta = 1.5$.

Plot (a) in Fig. 4.5 presents the diagonal values of R in Algorithm 1 for this numerical experiment. As indicated in Algorithm 1, we compute these diagonal values from the coarse-proxy model solutions and use them to select skeleton parameters. Plot (b) in Fig. 4.5 shows the values of the singular values σ_k in Algorithm 2, which are used to construct the reduced basis for $\epsilon = 5 \times 10^{-6}$. We also provide a visualization of the first 25 reduced basis (with respect to σ_k) in Fig. 4.6. Note that, as the singular values σ_k decay, the corresponding basis vectors contain increasingly more high frequency information.

Once again for the visualization proposes, we present the reduced basis for $\epsilon = 1 \times 10^{-4}$ in Fig. 4.6 — together with side-by-side comparisons, on three different parameter instances, of our reduced basis approximation for $\epsilon = 1 \times 10^{-4}$ to the underlying full-order solution, in Fig. 4.7. We note the reduced basis method gives a good approximation of the solution.



(a) Diagonals of R (b) Sigular values σ_k

Fig. 4.5. Left: the normalized diagonal values of R in the skeleton selection for the described radiative transport problem. Right: the normalized singular values in the basis construction for the described radiative transport problem.

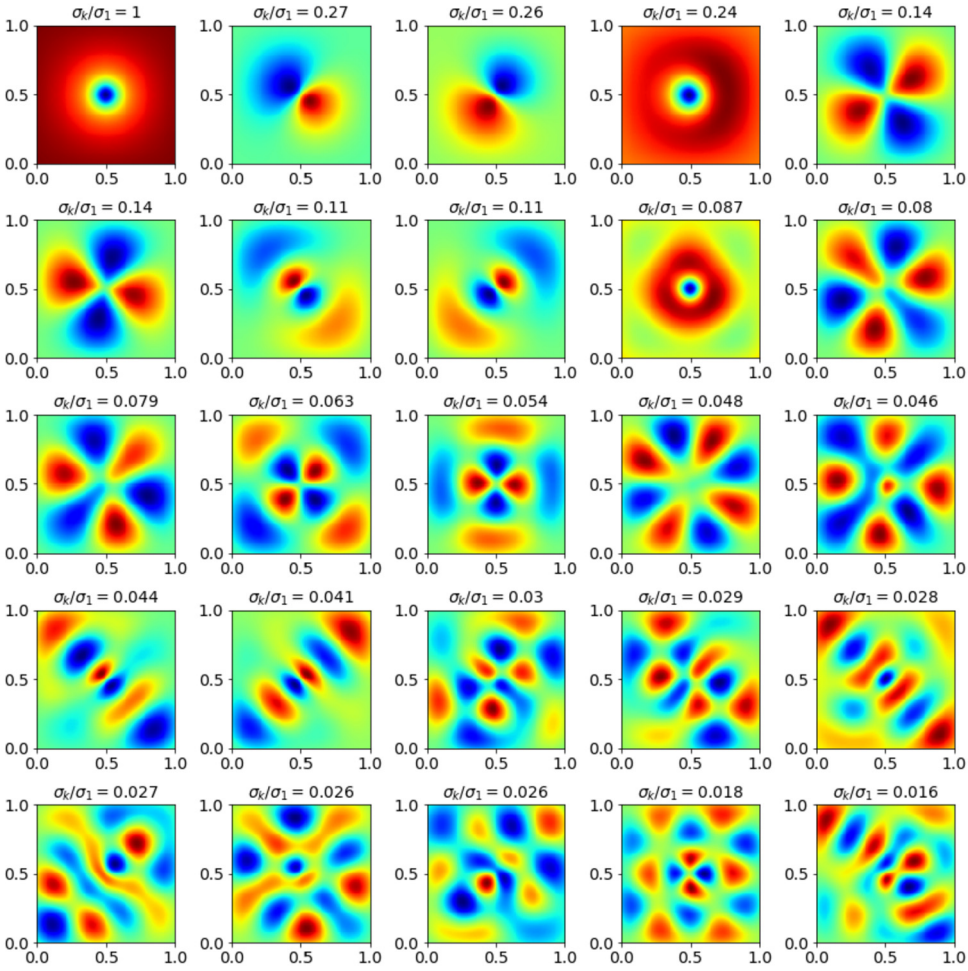


Fig. 4.6. Reduced basis vectors generated for the described radiative transport problem and their corresponding normalized singular values $\tilde{\sigma}_k$.

To further test the validity and efficiency of our reduced-order model, we run a parallel battery of tests to those we ran for the previous numerical example. For each element of Ω, we compute both the true solution $\mathrm{u}(\omega)$ and the reduced basis approximation $\mathrm{u}_{rb}(\omega)$, and afterwards evaluate the relative L^2 error $\|\mathrm{u}(\omega) - \mathrm{u}_{rb}(\omega)\|_2 / \|\mathrm{u}(\omega)\|_2$ (Fig. 4.8). We present the results of this computation in Table 4.2. When evaluating all solutions in bulk, our reduced basis method provides between a seventeen-fold and thirty-five-fold performance increase over the full-order model for a wide range of accuracy targets between .8% and 5% relative L^2 error. Note, once again, that this figure *includes the expensive offline phase of our reduced basis method*. While this numerical example does not quite match the linear convergence of the previous numerical example, we

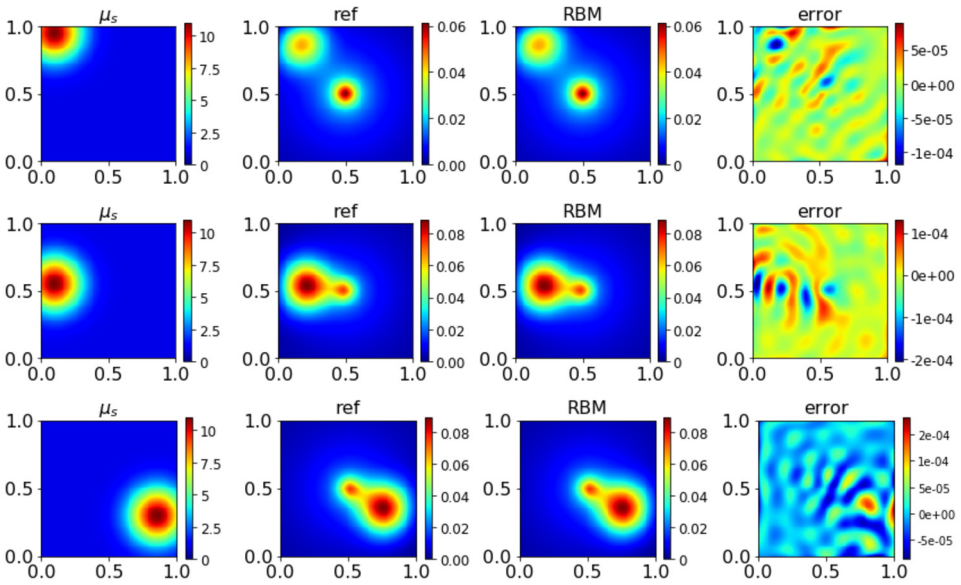
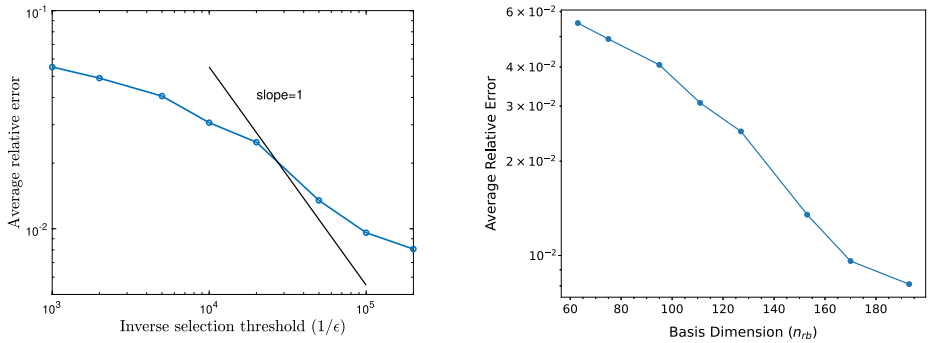


Fig. 4.7. Three examples for the solutions evaluated by the reduced basis method for the radiative transport equation for the parameter set Ω with threshold $\epsilon = 1 \times 10^{-4}$ and their corresponding reference solutions and error.



(a) Selection Threshold vs. Error (b) Basis Dimension vs. Error

Fig. 4.8. Left: A log-log convergence plot of the reduced method on the radiative transport problem, showing the average relative L^2 error over the parameter set Ω against the inverse of the selection threshold ϵ. Right: Same as left, except on the x-axis is reduced basis dimension.

still note that the error always decreases as the parameter ϵ decreases. Therefore, our method exhibits convergence, as seen in Table 4.2 and Fig. 4.6.

4.3. The effect of fine operator sample count on accuracy

We have found in our testing that the performance of our method is not particularly sensitive to the number of fine operator samples chosen. We can see the effect of changing the fine operator sample count in Fig. 4.9. Note that too few samples causes underfitting, while too many samples causes overfitting. These effects are particularly evident in the radiative transport example. For the problems above, we sample 10 columns of the fine operator (plus the operator diagonal).

5. Conclusion and future work

We have developed a simple and general-purpose reduced basis approximation technique for linear elliptic integral operators. As shown by the empirical results, this method results in significant performance increases on the simple problems we have applied it to. Due to the complexity scaling exhibited by numerical simulations, this method might produce even more significant performance increases at scale. Moreover, we hope that the techniques put forth in this paper will provide a useful starting point for future work in model order reduction for integral equations.

Possible avenues for such future work include the application of these techniques to larger scale problems, or perhaps, the application of these techniques to electromagnetic scattering to give a real comparison to currently existing work in [12,20]. One possible limitation of the technique we present here, and another possible areas for future work is the method

Table 4.2
Test results for the reduced basis method on the described radiative transport problem. Here ϵ denotes the selection threshold used for reduced basis construction, s denotes the number of skeletons selected by our method (i.e., the number of fine solves used to construct the reduced basis), n_{rb} denotes the dimension of the reduced basis constructed, $T_{rb}^{(offline)}$ denotes the amount of time in seconds used in reduced basis construction, $T_{rb}^{(online)}$ denotes the amount of time in seconds used to solve all problem instances from Ω using our method once the reduced basis has been constructed, $T_{rb} = T_{rb}^{(offline)} + T_{rb}^{(online)}$ denotes the total amount of time in seconds used by our method to compute approximations to all problem instances in Ω, and T_{fine}/T_{rb} denotes the ratio between the time taken by our method to compute approximate solutions to all problem instances in Ω and the time T_{fine} taken to naively compute all exact fine solutions in Ω, i.e., the computational speed-up our algorithm provides. Finally, L^2 error denotes the average relative L^2 error, i.e., $\|\mathsf{u}(\omega) - \mathsf{u}_{rb}(\omega)\|_2 / \|\mathsf{u}(\omega)\|_2$ averaged over the parameter set Ω.

ϵ	s	n_{rb}	$T_{rb}^{(offline)}$	$T_{rb}^{(online)}$	T_{fine}	T_{fine}/T_{rb}	L^2 error
1×10^{-3}	97	63	1274 s	12.25 s	46676 s	36.3×	0.0552
5×10^{-4}	118	75	1478 s	21.97 s	46676 s	31.1×	0.0491
2×10^{-4}	148	95	1489 s	83.64 s	46676 s	29.7×	0.0406
1×10^{-4}	169	111	1565 s	110.2 s	46676 s	27.9×	0.0307
5×10^{-5}	192	127	1786 s	144.7 s	46676 s	24.2×	0.0249
2×10^{-5}	225	153	1995 s	187.5 s	46676 s	21.4×	0.0135
1×10^{-5}	250	170	2095 s	239.1 s	46676 s	20.0×	0.0096
5×10^{-6}	277	193	2388 s	303.8 s	46676 s	17.3×	0.0081

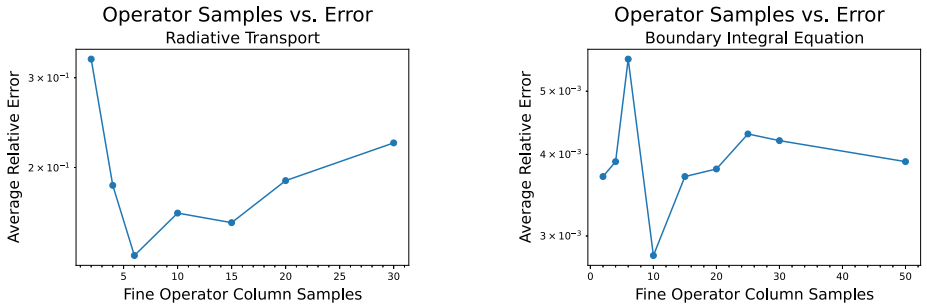


Fig. 4.9. Average relative error plotted against number of columns sampled from the fine operator. Note that the error is relatively consistent across different sampling levels. Note that the larger the number of samples, the more expensive the computation of the mixing matrix. In our experiments, we use ten sampled columns as a good default.

by which the reduced operators $\mathsf{Q}^{\mathsf{T}}\mathsf{L}(\omega)\mathsf{Q}$ are assembled. One could imagine finding a better operator sampling mask than the randomly selected ones in this paper. It may also be possible that at scale, the method we use to compute interpolation coefficients may be subject to overfitting. However, in our experience working on the radiative transport and Laplace equation examples, the interpolation error for the reduced operators is not a dominant source of error except at very small values of the threshold ϵ where the total average error is already very small. Another limitation of this method is that it is specifically tailored for integral equations due to the way it reconstructs operators. While there is no reason one could not use this technique on problems with sparse operators (i.e., differential equations), there are already a large number of competitors in that space (e.g., matrix POD, empirical interpolation), and it is not clear that the method we present here would necessarily outperform those. However, the aforementioned techniques for sparse matrices do not extend to dense matrices and hence integral equations is where the contribution of our method is most pronounced. Finally, for problems at scale, there is a trade-off that must be made between the quality and computation time for the coarse-proxy model. It may be useful in this situation to use a series of coarse-proxy models (rather than a single one), each subsequent one finer than the last, to progressively filter down the parameter set Ω to the skeleton set $\widehat{\Omega}$.

6. Source code and reproduction

All source code for this paper as well as instructions for reproducing the results in this paper are publicly available at https://github.com/UniqueUpToPermutation/coarse-proxy-benchmark.

CRediT authorship contribution statement

Philip A. Etter: Conceptualization, Investigation, Software, Writing – original draft. **Yuwei Fan:** Writing – review & editing. **Lexing Ying:** Conceptualization, Funding acquisition, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] B. Almroth, P. Stern, F.A. Brogan, Automatic choice of global shape functions in structural analysis, AIAA J. 16 (5) (1978) 525–528.
[2] B. Alpert, G. Beylkin, R. Coifman, V. Rokhlin, Wavelet-like bases for the fast solution of second-kind integral equations, SIAM J. Sci. Comput. 14 (1) (1993) 159–184.
[3] K.E. Atkinson, The numerical solution of integral equations of the second kind, Cambridge Monogr. Appl. Comput. Math. (1996).
[4] E. Balmes, Parametric families of reduced finite element models. Theory and applications, Mech. Syst. Signal Process. 10 (4) (1996) 381–394.
[5] A. Barrett, G. Reddien, On the reduced basis method, Z. Angew. Math. Mech. 75 (7) (1995) 543–549.
[6] P. Benner, S. Gugercin, K. Willcox, A survey of projection-based model reduction methods for parametric dynamical systems, SIAM Rev. 57 (4) (2015) 483–531.
[7] P. Binev, A. Cohen, W. Dahmen, R. DeVore, G. Petrova, P. Wojtaszczyk, Convergence rates for greedy algorithms in reduced basis methods, SIAM J. Math. Anal. 43 (3) (2011) 1457–1472.
[8] S. Boyaval, C. Le Bris, T. Lelievre, Y. Maday, N.C. Nguyen, A.T. Patera, Reduced basis techniques for stochastic problems, Arch. Comput. Methods Eng. 17 (4) (2010) 435–454.
[9] K. Carlberg, R. Tuminaro, P. Boggs, Preserving Lagrangian structure in nonlinear model reduction with application to structural dynamics, SIAM J. Sci. Comput. 37 (2) (2015) B153–B184.
[10] Z. Drmac, S. Gugercin, A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions, SIAM J. Sci. Comput. 38 (2) (2016) A631–A648.
[11] Y. Fan, J. An, L. Ying, Fast algorithms for integral formulations of steady-state radiative transfer equation, J. Comput. Phys. 380 (2019) 191–211.
[12] M. Fares, J.S. Hesthaven, Y. Maday, B. Stamm, The reduced basis method for the electric field integral equation, J. Comput. Phys. 230 (14) (2011) 5532–5555.
[13] J. Fink, W. Rheinboldt, On the error behavior of the reduced basis technique for nonlinear finite element approximations, Z. Angew. Math. Mech. 63 (1) (1983) 21–28.
[14] M. Ganesh, J.S. Hesthaven, B. Stamm, A reduced basis method for electromagnetic scattering by multiple particles in three dimensions, J. Comput. Phys. 231 (23) (2012) 7756–7779.
[15] J.C. Goswami, A.K. Chan, C.K. Chui, On solving first-kind integral equations using wavelets on a bounded interval, IEEE Trans. Antennas Propag. 43 (6) (1995) 614–622.
[16] L. Greengard, The Rapid Evaluation of Potential Fields in Particle Systems, MIT Press, 1988.
[17] M.D. Gunzburger, Finite Element Methods for Viscous Incompressible Flows: a Guide to Theory, Practice, and Algorithms, Elsevier, 2012.
[18] J. Hampton, H.R. Fairbanks, A. Narayan, A. Doostan, Practical error bounds for a non-intrusive bi-fidelity approach to parametric/stochastic model reduction, J. Comput. Phys. 368 (2018) 315–332.
[19] J.S. Hesthaven, G. Rozza, B. Stamm, et al., Certified Reduced Basis Methods for Parametrized Partial Differential Equations, Springer, 2016.
[20] J.S. Hesthaven, B. Stamm, S. Zhang, Certified reduced basis method for the electric field integral equation, SIAM J. Sci. Comput. 34 (3) (2012) A1777–A1799.
[21] K. Ito, S. Ravindran, A reduced-order method for simulation and control of fluid flows, J. Comput. Phys. 143 (2) (1998) 403–425.
[22] R. Kress, V. Maz'ya, V. Kozlov, Linear Integral Equations, vol. 82, Springer, 1989.
[23] K.L. Ho, L. Ying, Hierarchical interpolative factorization for elliptic operators: integral equations, Commun. Pure Appl. Math. 69 (7) (2016) 1314–1353.
[24] A.S. Lyrintzis, The Use of Kirchhoff's Method in Computational Aeroacoustics, 1994.
[25] Y. Maday, A.T. Patera, G. Turinici, Global a priori convergence theory for reduced-basis approximations of single-parameter symmetric coercive elliptic partial differential equations, C. R. Math. 335 (3) (2002) 289–294.
[26] Y. Maday, A.T. Patera, G. Turinici, A priori convergence theory for reduced-basis approximations of single-parameter elliptic partial differential equations, J. Sci. Comput. 17 (1–4) (2002) 437–446.
[27] D.A. Nagy, Modal representation of geometrically nonlinear behavior by the finite element method, Comput. Struct. 10 (4) (1979) 683–688.
[28] A. Narayan, C. Gittelson, D. Xiu, A stochastic collocation algorithm with multifidelity models, SIAM J. Sci. Comput. 36 (2) (2014) A495–A521.
[29] A.K. Noor, J.M. Peters, Reduced basis technique for nonlinear analysis of structures, AIAA J. 18 (4) (1980) 455–462.
[30] A.T. Patera, G. Rozza, et al., Reduced Basis Approximation and a Posteriori Error Estimation for Parametrized Partial Differential Equations, 2007.
[31] D.J. Perry, R.M. Kirby, A. Narayan, R.T. Whitaker, Allocation strategies for high fidelity models in the multifidelity regime, SIAM/ASA J. Uncertain. Quantificat. 7 (1) (2019) 203–231.
[32] J.S. Peterson, The reduced basis method for incompressible viscous flow calculations, SIAM J. Sci. Stat. Comput. 10 (4) (1989) 777–786.
[33] J.R. Phillips, E. Chiprout, D.D. Ling, Efficient full-wave electromagnetic analysis via model-order reduction of fast integral transforms, in: Proceedings of the 33rd Annual Design Automation Conference, ACM, 1996, pp. 377–382.
[34] T. Porsching, Estimation of the error in the reduced basis method solution of nonlinear equations, Math. Comput. 45 (172) (1985) 487–496.
[35] K. Ren, R. Zhang, Y. Zhong, A fast algorithm for radiative transport in isotropic media, arXiv preprint arXiv:1610.00835, 2016.
[36] W.C. Rheinboldt, On the theory and error estimation of the reduced basis method for multi-parameter problems, Nonlinear Anal., Theory Methods Appl. 21 (11) (1993) 849–858.

[37] G. Rozza, D.B.P. Huynh, A.T. Patera, Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations, Arch. Comput. Methods Eng. 15 (3) (2007) 1.
[38] K. Veroy, C. Prud'Homme, D. Rovas, A. Patera, A posteriori error bounds for reduced-basis approximation of parametrized noncoercive and nonlinear elliptic partial differential equations, in: 16th AIAA Computational Fluid Dynamics Conference, 2003, p. 3847.
[39] R.L. Wagner, W.C. Chew, A study of wavelets for the solution of electromagnetic integral equations, IEEE Trans. Antennas Propag. 43 (8) (1995) 802–810.
[40] X. Zhu, A. Narayan, D. Xiu, Computational aspects of stochastic collocation with multifidelity models, SIAM/ASA J. Uncertain. Quantificat. 2 (1) (2014) 444–463.