Application of Proper Generalized Decomposition to Multigroup Neutron Diffusion Eigenvalue Calculations

Zachary M. Prince[a], Jean C. Ragusa[a,*]

[a] *Texas A&M University, Department of Nuclear Engineering, College Station, TX 77843, USA*

Abstract

In this paper, proper generalized decomposition (PGD) is utilized to reduce the computational burden of evaluating multigroup neutron diffusion eigenvalue problems. PGD is a reduced order modeling technique that seeks a separated representation to a multi-dimensional variable. In this application, each multigroup flux is sought as a finite sum of separable one-dimensional functions. This representation can significantly reduce the burden of evaluating multi-dimensional linear systems. Therefore, we use PGD to evaluate the linear systems within the power iteration process of the eigenvalue problem. In this paper, we discuss our implementation of PGD to these eigenvalue systems including a derivation of PGD operators for multigroup neutron diffusion problems with standard power iteration and power iteration accelerated with adaptive Wielandt shift. To illustrate PGD's effectiveness, we apply our implementation to eigenvalue problems ranging from homogeneous to highly heterogeneous geometries with one-,two-, and four-group material properties. With comparison to full-order model evaluation with MOOSE, we find that the effectiveness of PGD is problem dependent. PGD always out performs the full-order model with close to homogeneous problems, but performs more poorly with more realistic reactor problems.

Keywords: Reduced order modeling, Proper generalized decomposition,

*Corresponding author

Email addresses: zachmprince@tamu.edu (Zachary M. Prince), jean.ragusa@tamu.edu (Jean C. Ragusa)

1. Introduction

Despite the advances in computer architectures and numerical analysis, eigenvalue calculations for full-core nuclear reactors are still CPU-intense task, even more so when in the context of core reload optimization. The computational burden in these computations is partly due to the dimensionality of the problem, or the size of the phase space. A neutron diffusion eigenvalue problem involves a four dimensional phase-space, three for space and one for energy. Reduced Order Modeling (ROM) is a technique that attempts to decrease the size of a system by extracting the relevant information into a much smaller subspace. Typical application of ROM for eigenvalue problems are based on *a posteriori* projection methods, such as Proper Orthogonal Decomposition, see [1, 2, 3] for an overview of POD and its applications. Some applications of POD include computational fluid dynamics [4, 5, 6], shallow water [7, 8], and mechanics [9, 10]. Some recent applications of ROMs in the nuclear engineering community include reactor kinetics [11], particle transport [12, 13], multi-physics [14], uncertainty quantification [15], and criticality [16, 17], for instance. These methods rely on gathering data sets, or snapshots, of the full phase-space solution; that data is used to "discover" an appropriate subspace which is then used to seek a reduced-order model using a Galerkin or Petrov-Galerkin projection [18]. The adequacy of the subspace employed in the projection step can be highly dependent on the exploration of the parameter space during the snapshot generation, so "certifying" such reduced order models is often impossible. Contrary to these common approaches, the work presented in this paper utilizes an *a priori* ROM, known as Proper Generalized Decomposition, to evaluate k-eigenvalue problems in multigroup neutron diffusion.

Proper Generalized Decomposition (PGD) techniques aim at reducing the dimensionality of a system by extending the concept of separation of variables. Indeed, to address the computational burden of solving multi-dimensional sys-

tems, PGD seeks a decomposition of the system solution into a separated representation [19, 20]. As such, PGD represents a multi-variate function as a sum of products of one-dimensional functions. For instance, a D-dimensional solution is sought as a sum of N terms, each term being a product of D one-variable functions, shown in Eq. (1a), where N is large enough to represent the solution to a certain tolerance. For instance, a three-dimensional variable $u(x, y, z)$ can be decomposed into a sum of N terms consisting of products of functions of x, y, and z, shown in Eq. (1b).

$$u(x_1, x_2, ..., x_D) = \sum_{i=1}^{\infty} \prod_{j=1}^{D} u_i^{x_j}(x_j) \approx \sum_{i=1}^{N} \prod_{j=1}^{D} u_i^{x_j}(x_j)\,, \tag{1a}$$

$$u(x, y, z) = \sum_{i=1}^{\infty} X_i(x) Y_i(y) Z_i(z) \approx \sum_{i=1}^{N} X_i(x) Y_i(y) Z_i(z)\,. \tag{1b}$$

The advantage of PGD is readily apparent. If a solution to a problem involves D dimensions with M nodes in each dimension, the full-order model version of the problem will involve solving a multi-dimensional system with M^D unknowns. However, if tackled with PGD, the same problem requires $N \times D \times M$ unknowns ($D \times M$ unknown for each of the N terms). Put simply, a traditional approach leads to system sizes that increase exponentially with the number of dimensions, while the system sizes only increase linearly with a PGD approach. This feature of PGD is exploited for high dimensional systems; see also [21]. Additionally, if the problem is uniformly refined in all dimensions, a traditional approach leads to an increase in the number of unknown by a factor 2^D, for the increase is only a factor $2D$ for PGD. The disadvantage with PGD is that a system of otherwise linear equations transforms into a nonlinear problem because of the representation of the solution as product of one-dimensional solutions.

The multigroup neutron diffusion criticality equations form an eigenvalue problem where the scalar flux ($\Phi \equiv [\phi^1, ..., \phi^G]$) is the eigenvector and the neutron multiplication factor (k) is the eigenvalue [22, 23]. Being an eigenvalue problem, evaluating ϕ and k requires a nonlinear process. A standard approach to evaluating the eigenvalue problem is power iteration. However, the conver-

gence rate of power iteration is directly related to the dominance ratio (the ratio of second largest to the first largest eigenvalue), which can be close to unity for nuclear core problems [24, 25]. Therefore, acceleration techniques are often employed, including utilizing Chebyshev polynomials [26, 27, 28] or Wielandt shift [29, 30, 31]. Projection-based approaches are also commonly employed, e.g., Arnoldi's methods, particularly the implicit restarted Arnoldi method (IRAM) [32, 33, 34], and Krylov subspace methods, e.g., Davidson method [35, 36, 37]. Newton-based approaches have also been employed, for instance Jacobian-free Newton-Krylov approaches [38, 39, 40]. Gonzalez et al. in [41] investigated the use of PGD with a Arnoldi-like process for a single-group k-eigenvalue problem. However, this process does not guarantee the evaluation of the largest eigenvalue or the convergence of the eigenvector. This work utilizes standard power iteration accelerated with a Wielandt shift, more specifically, an adaptive shift [31]. In addition to the work done in [41], Senecal et al. in [42] applied PGD using the MOOSE library (Multi-physics Object Oriented Simulation Environment) to steady-state neutron diffusion problems, but with mild spatial variations.

The goal of this work is to investigate PGD as solution technique for criticality problems, as found in reactor-physics applications, and to comment on its strengths and weaknesses for such applications. This paper presents three extensions to the PGD technique: (1) the application of PGD to accelerated power iterations for eigenvalue problems, (2) the design of an effective iteration procedure for PGD applied to eigenvalue problems, including compression of the fission integral at each iteration, (3) the construction of PGD operators for multigroup neutron diffusion in highly heterogeneous configurations and its effective implementation. The outline of the paper is as follows: in Section 2, we briefly review multigroup diffusion criticality problems and their numerical solution using accelerated power iteration techniques. Section 3 discusses the PGD solution process, first in a general sense followed by its application to power iterations. Results are presented in Section 4.

2. Overview of the Multigroup Eigenvalue Problem

In this section, we recall the multigroup k-eigenvalue problem and discuss the power iteration and shifted power iteration process. For criticality calculations, the eigenvalue-eigenvector pair of interest is the one associated with the largest eigenvalue (fundamental mode). The largest value of k, known as the effective neutron multiplication factor or k_{eff}, can be computed using power iteration, for example, or an accelerated variant of power iteration.

2.1. Multigroup Eigenvalue Problem

The multigroup neutron diffusion k-eigenvalue equations are

$$-\nabla \cdot D^g \nabla \phi^g + \Sigma_t^g \phi^g = \sum_{g'=1}^{G} \Sigma_s^{g \leftarrow g'} \phi^{g'} + \frac{1}{k} \chi^g \sum_{g'=1}^{G} \nu \Sigma_f^{g'} \phi^{g'} \,,\ g = 1, ..., G \,. \quad (2)$$

Each term in this equation is defined as:

G - total number of energy groups

ϕ^g - multigroup scalar neutron flux

D^g - multigroup diffusion coefficient

Σ_t^g - total macroscopic cross section

$\Sigma_s^{g \leftarrow g'}$ - macroscopic scattering cross section from group g' to group g

χ^g - fraction of neutrons generated in group g from fission (fission spectrum)

$\nu \Sigma_f^{g'}$ - product of the macroscopic fission cross section and the average number of neutrons emitted per fission in group g'

The equations in (2) can then be written into matrix form,

$$(\boldsymbol{D} + \boldsymbol{A} - \boldsymbol{S})\, \Phi = \frac{1}{k} \boldsymbol{F} \Phi \,, \quad (3)$$

where

$$(\Phi)_g = \phi^g \,, \quad (4a)$$

$$(\boldsymbol{D})_{gg'} = -\nabla \cdot D^g \nabla \delta_{gg'} \,, \tag{4b}$$

$$(\boldsymbol{A})_{gg'} = \Sigma_t^g \delta_{gg'} \,, \tag{4c}$$

$$(\boldsymbol{S})_{gg'} = \Sigma_s^{g \leftarrow g'} \,, \tag{4d}$$

$$(\boldsymbol{F})_{gg'} = \chi^g \nu \Sigma_f^{g'} \,. \tag{4e}$$

The neutron diffusion criticality equations form a generalized eigenvalue problem, which can be written in operator notation as:

$$\mathcal{A}\Phi = \frac{1}{k}\mathcal{B}\Phi \,, \tag{5}$$

with

$$\mathcal{A} = \boldsymbol{D} + \boldsymbol{A} - \boldsymbol{S} \quad \text{and} \quad \mathcal{B} = \boldsymbol{F} \,.$$

This is a nonlinear problem for the eigenpair (k, Φ). Numerous methods have been devised to solve this eigenproblem; see, for instance, the references cited in the introduction. This work will focus on the application of power iteration and shifted power iteration.

2.2. Power Iteration and Shifted Power Iteration

The standard power iteration process proceeds as follows. Given a previous iterate ℓ for the pair (k, Φ), the new iteration is obtained by solving

$$\mathcal{A}\Phi^{\ell+1} = \frac{1}{k^\ell}\mathcal{B}\Phi^\ell \,, \tag{6a}$$

for the new estimate of the eigenvector; the next estimate of the eigenvalue is then computed using

$$k^{\ell+1} = \frac{\left\langle \Phi^{\ell+1}, \mathcal{B}\Phi^{\ell+1} \right\rangle}{\left\langle \Phi^{\ell+1}, \mathcal{A}\Phi^{\ell+1} \right\rangle} = \frac{\sum_{g=1}^{G} \int_{\mathcal{D}} (\Phi^{\ell+1})_g (\mathcal{B}\Phi^{\ell+1})_g d\vec{r}}{\sum_{g=1}^{G} \int_{\mathcal{D}} (\Phi^{\ell+1})_g (\mathcal{A}\Phi^{\ell+1})_g d\vec{r}} \,, \tag{6b}$$

where ℓ is the iteration index and the definition of k is known as the (inverse of the) Rayleigh quotient [43]. This iterative technique is simple to implement, but suffer from slow convergence behavior in reactor physics where the dominance ratio come be close to one [24]. Shifted power iteration is an example

of accelerated power iterations whereby a shifting parameter is introduced to move the dominance ratio away from unity. This shift is applied as

$$\left(\mathcal{A} - \sigma^{\ell}\mathcal{B}\right)\Phi^{\ell+1} = \left(\frac{1}{k^{\ell}} - \sigma^{\ell}\right)\mathcal{B}\Phi^{\ell}\,, \tag{7}$$

where $k^{\ell+1}$ is computed using Eq. (6b). The parameter σ is known as the Wielandt shift. Selecting $\sigma = 0$ would equate to a standard power iteration process. Application of shift can greatly improve the convergence of the iteration procedure, a value of σ closest to the value of $\frac{1}{k_{\text{eff}}}$ is optimal [43]. The Rayleigh quotient iteration uses a shift equal to the last computed eigenvalue, i.e., $\sigma^{\ell} = 1/k^{\ell}$. This choice of the Wielandt shift can result in a cubic convergence of the eigenvalue, while a constant shift converges linearly [44]. However, since the eigenvector typically converges more slowly than the eigenvalue, the system defined in Eq. (7) can become ill-conditioned before the eigenvector is fully converged. Yee et. al [31] analyzed several other adaptive approaches for σ, the one chosen for this work is defined as

$$\sigma^{\ell} = \max\left(\frac{1}{k^{\ell}} - c_1\left|\frac{1}{k^{\ell}} - \frac{1}{k^{\ell-1}}\right| - c_0, \frac{1}{k_{\max}}\right)\,, \tag{8}$$

where c_1, c_0, and $k_{\max}$ are parameters chosen to be 10, 0.01, and 3, respectively, in this paper. These values were chosen for analysis in [31]. Different values may prove more optimal for specific eigenvalue problems, but this is beyond the scope of this work.

The systems of equations defined by Eqs. (6a) and (7) are linear systems that can be represented in the form $\mathcal{L}\Phi = q$. Solving such systems requires an inversion of the operator $\mathcal{L}$, given a source q. It is at this stage that the PGD method is applied. PGD seeks a separated representation as an approximation to Φ. As discussed earlier, there are iterative steps associated with PGD, (1) solving for the various one-dimensional components of the separated representation and (2) enrichment the representation with enough terms. These PGD iterations will thus be nest inside the power iteration process. The nesting of power iteration and PGD solver, as well as special considerations regarding the PGD operator decompositions, are discussed in the next section.

3. Proper Generalized Decomposition

So far, we have discussed the derivation of multigroup criticality as a generalized eigenvalue problem and the power iteration procedure to resolve the nonlinearities of the eigenvalue-eigenvector pair. In this section, we discuss the application of PGD within a power iteration. Fig. 1 illustrates the design of this application.

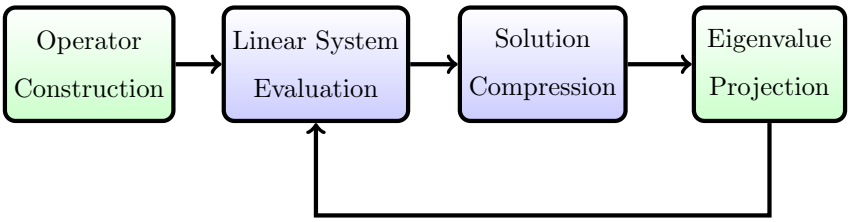


Figure 1: Design of PGD application to power iteration procedure

As stated earlier, PGD seeks a separated representation, as shown in Eq. (1b), for the solution of a system. For brevity, we describe here the PGD process in the case of a two-dimensional problem, but the 3-D derivation can easily inferred from our exposé. Note that once a solution is approximated (and sought) as products of one-dimensional functions, it is advantageous from an implementation point-of-view to represented the operators in a similar manner. This operator decomposition is discussed in Section 3.1. After the operator has been defined and a certain iteration's solution established, the resulting linear system can be solve using the PGD technique, discussed in Section 3.2. Once a solution from the linear system is resolved, the separated form can be compressed in order to alleviate the cost of eigenvalue evaluation and source construction. PGD compression is explained in Section 3.3. Finally, Section 3.4 discusses the computation of the eigenvalue after a PGD projection step; the error definitions for the iteration procedures are also introduced at that moment.

3.1. Operator Decomposition

In the case of a two-dimensional problem, a PGD representation (decomposition) of the eigenvector as given by Eq. (9a) [19].

$$\Phi(x,y) = \sum_{i=1}^{N} X_i(x) Y_i(y)\,. \tag{9a}$$

The operators of the generalized eigenvalue problem can be similarly decomposed:

$$\mathcal{A}(x,y) = \sum_{j=1}^{A} \mathcal{A}_j^x(x) \mathcal{A}_j^y(y)\,. \tag{9b}$$

$$\mathcal{B}(x,y) = \sum_{j=1}^{B} \mathcal{B}_j^x(x) \mathcal{B}_j^y(y)\,. \tag{9c}$$

We then re-write the generalized eigenvalue problem of Eq. (5) using a PGD representation:

$$\sum_{i=1}^{N}\sum_{j=1}^{A} \mathcal{A}_j^x(x) X_i(x) \mathcal{A}_j^y(y) Y_i(y) = \frac{1}{k} \sum_{i=1}^{N}\sum_{j=1}^{B} \mathcal{B}_j^x(x) X_i(x) \mathcal{B}_j^y(y) Y_i(y)\,, \tag{9d}$$

where A and B are the number of terms needed to fully decompose the $\mathcal{A}$ and $\mathcal{B}$ operators, respectively. For example, if $\mathcal{A}$ is the diffusion operator in an homogeneous domain ($\mathcal{A} = -D\partial^2_{x^2} - D\partial^2_{y^2}$), A would equal two: one for diffusion in x and one for diffusion in y, shown in Eq. (10):

$$\mathcal{A}u(x,y) := -\nabla \cdot D \nabla u(x,y) = -\sum_{i=1}^{N} \left[D \frac{d^2 X_i}{dx^2} Y_i + D X_i \frac{d^2 Y_i}{dy^2} \right]\,, \tag{10}$$

where we recall that N is the number of enrichment terms needed to represent $u(x,y)$. For an operator with spatially-dependent coefficients (e.g., piece-wise constant per material zone), the integers A and B will depend on the separated representation (decomposition) of these properties. PGD operators for heterogeneous media are explained in Section 3.5.

3.2. Linear System Solves using PGD

At each power iteration (Eqs. (6) and (7)), one needs to solve a linear system $\mathcal{L}\Phi = q$, where the source term is the lagged fission integral from the previous

power iteration. Here, we explain how PGD is employed to solve the linear system $\mathcal{L}\Phi = q$. This linear system, in the PGD framework, can be generically defined as

$$\mathcal{L}(x,y)\Phi(x,y) = \sum_{m=1}^{L}\sum_{i=1}^{N} \mathcal{L}_m^x(x) X_i(x) \mathcal{L}_m^y(y) Y_i(y)\,, \tag{11a}$$

$$q(x,y) = \sum_{n=1}^{Q} q_n^x(x) q_n^y(y)\,, \tag{11b}$$

where L and Q are the numbers of terms required to PGD-decompose $\mathcal{L}$ and q, respectively. In the context of shifted power iteration, these decomposed operators are defined as

$$\mathcal{L}_m^{x,\ell+1}\mathcal{L}_m^{y,\ell+1} = \begin{cases} \mathcal{A}_m^x \mathcal{A}_m^y & m = 1,..,A \\ -\sigma^{\ell+1}\mathcal{B}_{m-A}^x \mathcal{B}_{m-A}^y & m = A+1,...,A+B \end{cases}\,, \tag{12a}$$

$$q_n^{x,\ell+1} q_n^{y,\ell+1} = \mathcal{B}_j^x X_i^\ell(x) \mathcal{B}_j^y Y_i^\ell(y)\,, \quad n = (i-1)B + j\,. \tag{12b}$$

The total number of terms in the linear operator is then $L = A + B$. It should be noted that Eq. (12a) applies for the shifted power iteration approach; in the unshifted case ($\sigma = 0$) the terms due to $\mathcal{B}$ are not present and thus $L = A$. The total number of terms in the source is $Q = B \times N$, which can become quite large if N is large, which can occur in highly heterogeneous media. Decreasing N in a power iteration decreases the computational effort for future iterations. Methods for minimizing N are discussed in Section 3.3.

The solution Φ is built iteratively, one term in the summation at a time, starting with the first term, $X_1(x)Y_1(y)$ all the way to $X_N(x)Y_N(y)$. Suppose all terms up to index $N-1$ are known. To solve for enrichment step N, a system of equations is written for each dimension. For instance, in order to solve for a enrichment X_N, Eq. (11) is multiplied by a test function $v(x,y) = X^*(x)Y_N(y)$ and integrated over the full domain, as shown in Eq. (13). Explicit dependence on x and y has been dropped hereafter for brevity, unless otherwise needed for clarity or emphasis.

$$\sum_{m=1}^{L} \int_{\Omega_x} X^* \mathcal{L}_m^x X_N dx \int_{\Omega_y} Y_N \mathcal{L}_m^y Y_N dy = \sum_{n=1}^{Q} \int_{\Omega_x} X^* q_n^x dx \int_{\Omega_y} Y_N q_n^y dy - \sum_{j=1}^{N-1} \sum_{m=1}^{L} \int_{\Omega_x} X^* \mathcal{L}_m^x X_j dx \int_{\Omega_y} Y_N \mathcal{L}_m^y Y_j dy \,. \quad (13)$$

Note that the 2D test function is a product of two one-dimensional functions: the basis functions in x, $X^*(x)$, and the latest solution term in y, $Y_N(y)$. Similar equations can be obtained for Y_N, but using test functions $X_N(x)Y^*(y)$. Note that each of these one-dimensional systems of equations depends on X_N and Y_N, so a fixed-point process is adopted to resolve the nonlinearities and to yield the next enrichment function $X_N(x)Y_N(y)$. Also note that all the previously computed enrichments ($j = 1, \ldots, N-1$) appeared on the right-hand side and are known quantities when solving for enrichment N.

To summarize, the solution process for PGD involves sequentially adding terms to the sum in Eq. (9a), also known as enriching the solution. The first enrichment is a direct separation of variables ($u(x, y) = X_1(y)Y_1(y)$) and each dimension is solved individually using Eq. (13). The nonlinear process is resolved using a fixed-point iteration or an alternating direction strategy [19]. Once converged, a new enrichment term is added to the decomposed solution, until the contribution of the latest term is less than a user-specified tolerance. The PGD process is shown in Fig. 2 for the case of a two-dimensional x-y problem. For additional details regarding PGD, we refer the reader to the following PGD books [19, 45, 46] and review papers [47, 48].

3.3. Solution Compression

In the power iteration process, the latest iterate is used to compute the next fission source. Here, the latest iteration is a PGD-decomposed solution and may not be optimal in terms of the number of expansion terms when used to compute the fission integral. Therefore, there is an opportunity to compress the PGD solution, i.e., find a new decomposition such that the number of summation terms, N in Eq. (9a), is reduced. This compression step aims at reducing the number of

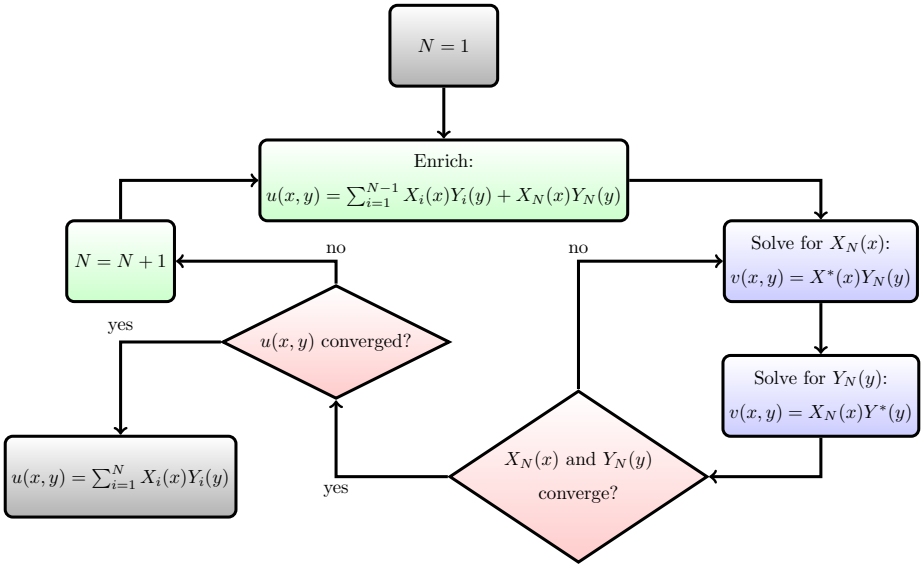


Figure 2: Visualization of the PGD solution process when solving a two-dimensional x-y problem

terms in the fission source term of the next power iteration ((*Q* in Eq. (11b))) resulting in faster evaluation of all the terms in Eq. (12b) and the right-hand-side of Eq. (13). An obvious candidate for PGD compression is employing Singular Value Decomposition (SVD) on the two-dimensional array representation of the full solution; SVD is known to yield an optimal decomposition [49]. However, SVD is only applicable to two-dimensional array representations and requires the full-order solution, which PGD attempts to avoid in the first place. Another possibility is to use a PGD approach to compress the PGD solution (see, for instance, Chapter 3 of [19]). We have opted for the latter approach.

To compress the current power iteration iterate obtained from the PGD process described in Section 3.2, we simply use the PGD technique, but applied to a purely algebraic equation this time:

$$\sum_{i=1}^{\tilde{N}<N} \tilde{X}_i(x)\tilde{Y}_i(y) = \sum_{i'=1}^{N} X_{i'}(x)Y_{i'}(y)\,. \tag{14}$$

The right-hand-side of Eq. (14) is known and the terms on the left-hand-side are solved progressively through the aforementioned alternating direction and

enrichment process. For instance, when evaluating the term $\tilde{X}_{\tilde{N}}(x)$, a test function of $v(x,y) = X^*(x)\tilde{Y}_{\tilde{N}}(y)$ is applied and integrated over the domain, resulting in:

$$\int_{\Omega_x} X^*(x)\tilde{X}_{\tilde{N}}(x)dx \int_{\Omega_y} \tilde{Y}_{\tilde{N}}(y)\tilde{Y}_{\tilde{N}}(y)dy = \sum_{i'=1}^{N} \int_{\Omega_x} X^*(x)X_{i'}(x)dx \int_{\Omega_y} \tilde{Y}_{\tilde{N}}(y)Y_i(y)dy - \sum_{i=1}^{\tilde{N}-1} \int_{\Omega_x} X^*(x)\tilde{X}_i(x)dx \int_{\Omega_y} \tilde{Y}_{\tilde{N}}(y)\tilde{Y}_i(y)dy \ . \quad (15)$$

Noticeably, Eq. (15) involves no differential operators, thus the strong form of this equation for $\tilde{X}_{\tilde{N}}$ can be written as,

$$\beta^y_{\tilde{N},\tilde{N}}\tilde{X}_{\tilde{N}}(x) = \sum_{i'=1}^{N} \gamma^y_{\tilde{N},i'}X_{i'}(x) - \sum_{i=1}^{\tilde{N}-1} \beta^y_{\tilde{N},i}\tilde{X}_i(x) \ , \quad (16a)$$

where β and γ are scalars defined as,

$$\beta^y_{i,j} = \int_{\Omega_y} \tilde{Y}_i(y)\tilde{Y}_j(y)dy \ , \quad (16b)$$

$$\gamma^y_{i,j} = \int_{\Omega_y} \tilde{Y}_i(y)Y_j(y)dy \ . \quad (16c)$$

An equation for $\tilde{Y}_{\tilde{N}}$ can be found similarly. The additional computational effort in evaluating this compressed solution is minuscule relative to the cost of a single power iteration solve since Eq. (16a) only requires the evaluation of 1D integrals and does not result in a new linear system to be solved.

3.4. Eigenvalue Projection

This section describes the projection step of the PGD solution, whereby each term in the PGD solution is rescaled optimally. This projection is similar to the projection described in early PGD works [41, 50, 51]; however, we exploit it to also obtain an improved eigenvector through a reduced eigenvalue problem. In this projection, one attempt at improving the PGD decomposition by assuming the optimal separated representation is a linear combination of the one found from the enrichment process. Using this methodology, we define the solution

found from enrichment and compression as $\Phi^{\ell+1/2}$ and the projected solution as:

$$\Phi^{\ell+1}(x,y) = \sum_{i'=1}^{N} \alpha_{i'} X_{i'}^{\ell+1/2}(x) Y_{i'}^{\ell+1/2}(y)\,. \tag{17}$$

where the α coefficients are to be determined. To compute the α coefficients, we apply Eq. (17) to the generalized eigenvalue problem Eq. (6a). Multiplying by different enrichment terms from the current PGD solution ($X_i^{\ell+1/2}(x) Y_i^{\ell+1/2}(y)$) and integrating over space, we then develop an eigenvalue problem for these coefficients:

$$\bar{\mathcal{A}}\boldsymbol{\alpha} = \frac{1}{k^{\ell+1}} \bar{\mathcal{B}}\boldsymbol{\alpha}\,, \tag{18}$$

where

$$(\boldsymbol{\alpha})_i = \alpha_i\,, \tag{19a}$$

$$(\bar{\mathcal{A}})_{ii'} = \sum_{j=1}^{A} \int_{\Omega_x} X_i^{\ell+1/2} \mathcal{A}_j^x X_{i'}^{\ell+1/2} dx \int_{\Omega_y} Y_i^{\ell+1/2} \mathcal{A}_j^y Y_{i'}^{\ell+1/2} dy\,, \tag{19b}$$

$$(\bar{\mathcal{B}})_{ii'} = \sum_{j=1}^{B} \int_{\Omega_x} X_i^{\ell+1/2} \mathcal{B}_j^x X_{i'}^{\ell+1/2} dx \int_{\Omega_y} Y_i^{\ell+1/2} \mathcal{B}_j^y Y_{i'}^{\ell+1/2} dy\,, \tag{19c}$$

$$i = 1, ..., N\,, \quad i' = 1, ..., N\,.$$

The eigenvalue problem given in Eq. (18) is much smaller than the full-order version because the spatial variables have been integrated out; it is therefore significantly cheaper to evaluate than a single PGD linear system solve. After this projection step, the improved PGD solution ($\Phi^{\ell+1}$) is the latest power iteration solution and it is employed to obtain the latest eigenvalue ($k^{\ell+1}$).

With the combination of the PGD solution process and the power iteration technique, various nested iterations are present. We summarize the resulting iteration loops next: (1) eigenvalue (power) iteration, (2) PGD enrichment, and (3) alternating direction iteration for a given PGD enrichment step. All of these iterations require convergence tolerances; these are defined below:

1. Power iteration:

$$E_k^{\ell+1} = \frac{|k^{\ell+1} - k^{\ell}|}{k^{\ell}} < \epsilon_k \text{ (eigenvalue tolerance)}, \tag{20a}$$

$$E_\phi^{\ell+1} = \frac{||\Phi^{\ell+1} - \Phi^{\ell}||}{||\Phi^{\ell}||} < \epsilon_\phi \text{ (eigenvector tolerance)}, \tag{20b}$$

2. PGD Enrichment

$$E_{\text{enr}}^{N} = \frac{||X_N(x)||\ ||Y_N(y)||}{\sum_{i=1}^{N} ||X_i(x)||\ ||Y_i(y)||} < \epsilon_{\text{enr}} \text{ (enrichment tolerance)}, \tag{21}$$

3. PGD Alternating direction

$$E_{\text{ad}}^{p+1} = \frac{||X_N^{p+1} - X_N^{p}||\ ||Y_N^{p+1} - Y_N^{p}||}{||X_N^{p+1}(x)||\ ||Y_N^{p+1}(x)||} < \epsilon_{\text{ad}} \text{ (fixed-point tolerance)}. \tag{22}$$

Because of the nested iteration, we find it beneficial to tighten the inner iteration tolerances as the power iteration converges, allowing for looser convergence checks in the earlier stages of the power iteration process. This "inexact" approach to the PGD solution process prevents "over solving" each power iteration. Therefore, the enrichment error in this work for a particular power iteration is defined as:

$$\epsilon_{\text{enr}}^{\ell+1} = 0.01 \times \max\left(E_k^{\ell}, E_\phi^{\ell}\right) . \tag{23}$$

The resulting procedure for shifted power iteration with PGD is shown by Algorithm 1.

3.5. PGD for Multigroup Neutron Diffusion

In this section, we discuss the application of PGD to the operators and variables in the case of multigroup equations. Generally speaking, the application of the PGD process to the multigroup setting is relatively straightforward, just like extending a traditional one-group solver to the multigroup case, as in [52, 53], for instance. To begin, we define the scalar neutron flux and the linear operators as a decomposed spatial variable for each group (using a 2D example for brevity),

$$\phi^g(x, y) \approx \sum_{i=1}^{N} X_i^g(x) Y_i^g(y)\,, \tag{24a}$$

Algorithm 1 2D Generalized Eigenvalue Problem using PGD

procedure $[\Phi,k]$ = PGDEigenvalue($\mathcal{A}$,$\mathcal{B}$)

$\mathcal{A} := [\mathcal{A}_1^x, ..., \mathcal{A}_A^x; \mathcal{A}_1^y, ..., \mathcal{A}_A^y]$

$\mathcal{B} := [\mathcal{B}_1^x, ..., \mathcal{B}_B^x; \mathcal{B}_1^y, ..., \mathcal{B}_B^y]$

Initialization: $\ell = 0$, $E_k^\ell = 1$, $E_\phi^\ell = 1$, $\sigma^\ell = 1/k_{\max}$

Define initial eigenvector: Φ^ℓ

Compute eigenvalue: $k^\ell \leftarrow$ Eq. (6b)

while $E_k^\ell < \epsilon_k$ and $E_\phi^\ell < \epsilon_\phi$ **do**

Build linear operator: $\mathcal{L}^\ell \leftarrow$ Eq. (12a)

Build linear source: $q^\ell \leftarrow$ Eq. (12b)

Define enrichment tolerance: $\epsilon_{\text{enr}}^\ell \leftarrow$ Eq. (23)

Solve linear system: $\Phi^{\ell+1/2} := \sum_{i=1}^{N} X_i(x)Y_i(y) \leftarrow$Fig. 2

Compress eigenvector: $\Phi^{\ell+1/2} := \sum_{i=1}^{\tilde{N}\leq N} X_i(x)Y_i(y) \leftarrow$ Eq. (14)

Project solution and compute eigenvalue: $k^{\ell+1}$, $\Phi^{\ell+1} \leftarrow$ Eq. (18)

Compute shift: $\sigma^{\ell+1} \leftarrow$ Eq. (8)

$\ell = \ell + 1$

end while

end procedure

$$-\nabla \cdot D^g(x,y)\nabla = -\sum_{j=1}^{N_d}\left[\frac{\partial}{\partial x}D_j^{g,x}(x)\frac{\partial}{\partial x} + \frac{\partial}{\partial y}D_j^{g,y}(y)\frac{\partial}{\partial y}\right], \tag{24b}$$

$$\Sigma_t^g(x,y) = \sum_{j=1}^{N_t}\Sigma_{t,j}^{g,x}(x)\Sigma_{t,j}^{g,y}(y), \tag{24c}$$

$$\Sigma_s^{g\leftarrow g'}(x,y) = \sum_{j=1}^{N_s}\Sigma_{s,j}^{g\leftarrow g',x}(x)\Sigma_{s,j}^{g\leftarrow g',y}(y), \tag{24d}$$

$$\nu\Sigma_f^g(x,y) = \sum_{j=1}^{N_f}\nu\Sigma_{f,j}^{g,x}(x)\nu\Sigma_{f,j}^{g,y}(y). \tag{24e}$$

The decomposed description of the material properties can be generated using numerical or analytical techniques; for this work, we use the method described

in [21]. Note that the number of terms in each expansion (N, N_d, N_s, N_f) is independent of the group or group pair. Therefore, the number of terms for each group property description is equal to the number of terms of the group that has the largest expansion. In this paper, we assumed that the fission spectrum χ^g was independent of the material zone.

The one-dimensional variables for this work are discretized using first-order continuous finite elements (CFEM) [54]. With this framework, the multigroup flux and operators can be represented in matrix notation:

$$X_i^g(x)Y_i^g(y) \rightarrow (\boldsymbol{X}_i)_g \otimes (\boldsymbol{Y}_i)_g \,, \tag{25a}$$

$$\frac{\partial}{\partial x} D_j^{g,x}(x) \frac{\partial}{\partial x} \rightarrow (\boldsymbol{K}_j^x)_g \otimes \boldsymbol{M}^y \,, \quad \frac{\partial}{\partial y} D_j^{g,y}(y) \frac{\partial}{\partial y} \rightarrow \boldsymbol{M}^x \otimes (\boldsymbol{K}_j^y)_g \,, \tag{25b}$$

$$\Sigma_{t,j}^{g,x}(x)\Sigma_{t,j}^{g,y}(y) \rightarrow (\boldsymbol{M}_{t,j}^x)_{gg} \otimes (\boldsymbol{M}_{t,j}^y)_{gg} \,, \tag{25c}$$

$$\Sigma_{s,j}^{g\leftarrow g',x}(x)\Sigma_{s,j}^{g\leftarrow g',y}(y) \rightarrow (\boldsymbol{M}_{s,j}^x)_{gg'} \otimes (\boldsymbol{M}_{s,j}^y)_{gg'} \,, \tag{25d}$$

$$\chi^g \nu\Sigma_{f,j}^{g',x}(x)\nu\Sigma_{f,j}^{g',y}(y) \rightarrow (\boldsymbol{M}_{f,j}^x)_{gg'} \otimes (\boldsymbol{M}_{f,j}^y)_{gg'} \,. \tag{25e}$$

The group structure of the resulting matrices are equivalent to the ones described in Eq. (4). We can now easily define the eigenvalue problem operators described in Eq. (9):

$$\mathcal{A}_j^x = \begin{cases} \boldsymbol{K}_j^x & j = 1, ..., N_d \\ \boldsymbol{M}^x & j = N_d + 1, ..., 2N_d \\ \boldsymbol{M}_{t,j-2N_d}^x & j = 2N_d + 1, ..., 2N_d + N_t \\ \boldsymbol{M}_{s,j-2N_d-N_t}^x & j = 2N_d + N_t + 1, ..., 2N_d + N_t + N_s \end{cases} , \tag{26a}$$

$$\mathcal{A}_j^y = \begin{cases} \boldsymbol{M}^y & j = 1, ..., N_d \\ \boldsymbol{K}_{j-N_d}^y & j = N_d + 1, ..., 2N_d \\ \boldsymbol{M}_{t,j-2N_d}^y & j = 2N_d + 1, ..., 2N_d + N_t \\ \boldsymbol{M}_{s,j-2N_d-N_t}^y & j = 2N_d + N_t + 1, ..., 2N_d + N_t + N_s \end{cases} , \tag{26b}$$

$$\mathcal{B}_j^x = \boldsymbol{M}_{f,j}^x \,, \quad j = 1, ..., N_f \,, \tag{26c}$$

$$\mathcal{B}_j^y = \boldsymbol{M}_{f,j}^y \,, \quad j = 1, ..., N_f \,. \tag{26d}$$

Using these discretized operators for the multigroup neutron diffusion equation, the generalized eigenvalue problem can be evaluated using Algorithm 1.

4. Results

To illustrate the capability of our PGD algorithm for multigroup criticality problems, we apply it to four different problems. The first is a bare homogeneous cube with four-group cross sections, which is meant to verify the implementation with a analytic solution. The second example is a mildly heterogeneous 2-D two-group problem that Senecal & Wi analyzed as a fixed source problem in [42]. The third and final problems use the geometry and material properties of the BIBLIS benchmark [55], we utilize the one-group version from [41] and the two-group benchmark version.

The PGD method was implemented in MATLAB and utilizes first-order continuous finite elements (CFEM) for the 1D solutions. The linear systems in the PGD iteration process are evaluated using MATLAB's sparse LU solver. To produce a reference solution and compare performance with a full-order model (i.e., unseparated, multi-dimensional system evaluation), each problem was also implemented in the Multiphysics Object Oriented Simulation Environment (MOOSE) [56]. The MOOSE implementation also uses first-order CFEM and linear solvers are performed using algebraic multi-grid techniques from the HYPRE package with default numerical parameters. Both PGD and MOOSE solve each problem using the same sequence of regular rectangular grids. These grids are uniformly refined for each problem to gauge accuracy versus run-time for each method. These mesh refinements are identified by the number of elements in the x and y dimensions, the total number of elements is therefore this number squared.

Table 1: Material property values for homogeneous bare reactor

Group	D^g	Σ_r^g	$\nu\Sigma_f^g$	χ^g	$\Sigma_s^{g\leftarrow 1}$	$\Sigma_s^{g\leftarrow 2}$	$\Sigma_s^{g\leftarrow 3}$	$\Sigma_s^{g\leftarrow 4}$
1	2.500	0.0668	0.00835	0.75	0	0	0	0
2	1.050	0.0623	7.89e-4	0.25	0.668	0	0	0
3	0.676	0.0707	0.0101	0	4.77e-4	0.0630	0	0.00161
4	0.379	0.0909	0.117	0	0	3.0e-6	0.0482	0

4.1. Four-Group Bare Homogeneous Reactor

This first example is a 3-D four-group, homogeneous reactor, which has an derivable analytical solution. The reactor is $300\times300\times300$ cm with cross-section values described in Table 1. The analytical k_{eff} is 1.13530390 and eigenfunction is

$$\phi^g(x,y,z) = C^g \cos\left(\frac{\pi}{300}x\right)\cos\left(\frac{\pi}{300}y\right)\cos\left(\frac{\pi}{300}z\right) , \tag{27}$$

where C^g are group-dependent constants.

The full-order model, evaluated using MOOSE, was run with 10 to 40 elements per dimension, and the PGD model was run with 10 to 160 elements per dimension. The eigenvalue and eigenvector convergence tolerance (ϵ_k,ϵ_ϕ) were set to 10^{-9}. Fig. 3 shows the resulting eigenvalue and eigenvector error for MOOSE with power iteration and PGD with and without adaptive Wielandt shift. These errors show that both the full-order and PGD models have proper error convergence when the spatial mesh is refined. Fig. 4 shows the resulting run-time for each simulation. For this problem, PGD shows far superior performance compared to the multi-D evaluation using MOOSE. Since the exact solution is analytically separable, only one or two enrichments are needed as the power iteration converges, making this a computationally trivial problem for the PGD method. Table 2 shows the relative computational efficiency of linear system evaluation for each spatial refinement for the unshifted PGD model and full-order MOOSE model. The efficiency of a numerical scheme is defined as

$$\text{Efficiency} = \frac{1}{k_{\text{eff}}\,\text{Error}\times\text{CPU Time per PI}} . \tag{28}$$

Efficiency is higher for methods with low error and low CPU time. Because we employ unaccelerated and accelerated power iterations, the efficiency values use the average CPU time per power iteration. Table 2 shows the efficiency of the MOOSE solver and the PGD unshifted solver, relative to that of the PGD with shift approach. The efficiency of the unshifted PGD solver is greater than that of the shifted one is because the shifted operator appearing on the left-hand-side is more complex due to the Wiedlandt shift acting on the fission operator, making the unshifted PGD operator quicker to invert. The relative efficiency of the unshifted PGD scheme also increases with mesh refinement because the PGD solver spends less time in the tight enrichment tolerance range, due to the decrease in power iterations needed. When compared with the full multi-D solves performed using MOOSE, we note that the PGD approaches become significantly more efficient as the mesh is refined.

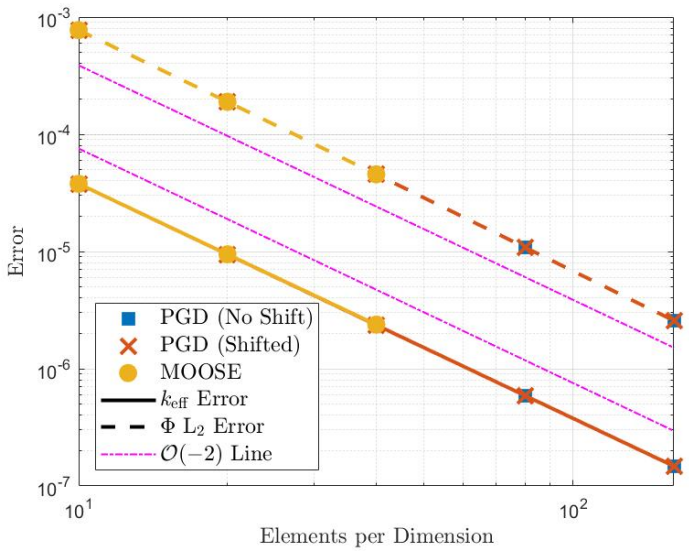


Figure 3: Relative eigenvalue and L_2 eigenvector error for the homogeneous four-group problem

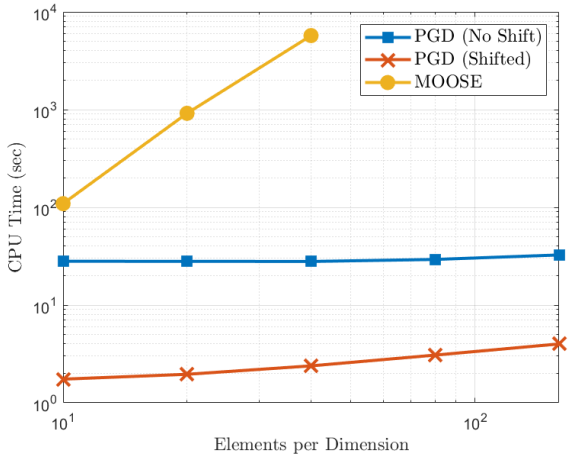


Figure 4: Run-time in seconds for each simulation of the homogeneous four-group problem

Table 2: Relative computational efficiency for unshifted PGD and MOOSE, homogeneous four-group problem (baseline is the PGD method with shift; values < 1 denote a less efficient technique)

Elements per Dim.	PGD	MOOSE
10	1.3002	0.3919
20	1.3826	0.0528
40	1.5803	0.0103
80	1.7943	—
160	1.9274	—

4.2. 2-D Two-Group Problem

This next example is a simple heterogeneous 2-D, two-group example based on Problem 5.7 from [57]; a fixed-source version of this problem was analyzed in [42]. Fig. 5 shows the geometry of the problem and the material properties can be found in Table 7 of [42], neglecting the fixed-source. The full-order model, evaluated using MOOSE, was run with 35 to 1,120 elements per dimension, and the PGD model was run with 35 to 2,240 elements per dimension. The eigenvalue convergence tolerance (ϵ_k) was set to 10^{-9}. A MOOSE model with

2,240 elements per dimension was used for the reference eigenvalue and eigenvector with $k_{\text{eff}} = 0.99010305$; this solution required high performance computing facilities, so the performance results for this simulation are not included.

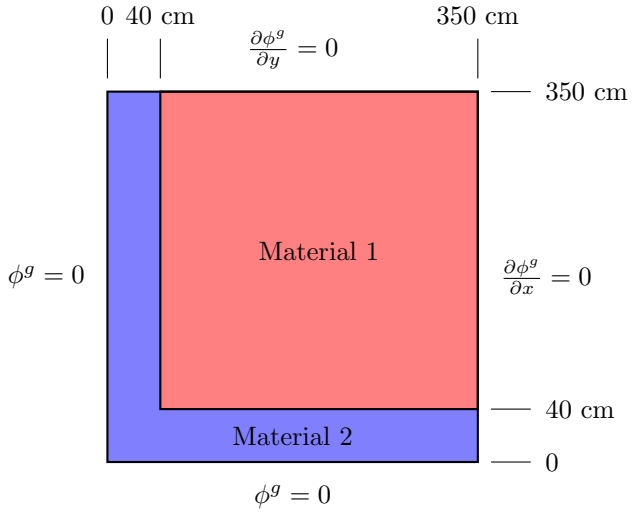


Figure 5: Geometry of 2-D two-group problem (figure not to scale)

Fig. 6 shows the resulting eigenvalue and eigenvector error for MOOSE with power iteration and PGD with and without adaptive Wielandt shift. These errors show that both the full-order and PGD models have proper error convergence when the spatial mesh is refined. The eigenvalue errors in the last two refinements begin to deviate slightly between each method. This is most likely due to the fact that the error at these points are relatively close to the tolerance set for the power iteration. Fig. 7 shows the resulting run-time for each simulation. For this problem, PGD shows far superior performance compared to multi-D MOOSE solves. Since this problem has very little heterogeneity, it is another relatively simple problem for PGD. Fig. 8 shows the contribution of each enrichment for the PGD runs at 560 elements per dimension and the magnitude of each term of the SVD of the full-order model solution. Both the SVD and PGD enrichment convergence shows that a relatively few number of terms are needed to represent the converged solution to the iteration tolerance. The

shifted PGD requires more enrichments than the unshifted simulation because the last power iteration had a tighter enrichment tolerance for the shifted simulation. This is due to the adaptive enrichment tolerance procedure defined by Eq. (23). Table 3 shows the relative computational efficiency for each refinement of the unshifted PGD model and full-order MOOSE model. The unshifted PGD efficiency shows the same trend as in the previous example. As noted earlier, the efficiency of the full multi-D solves (using MOOSE) decreases significantly as the mesh is refined, compare to that of PGD-based approaches.

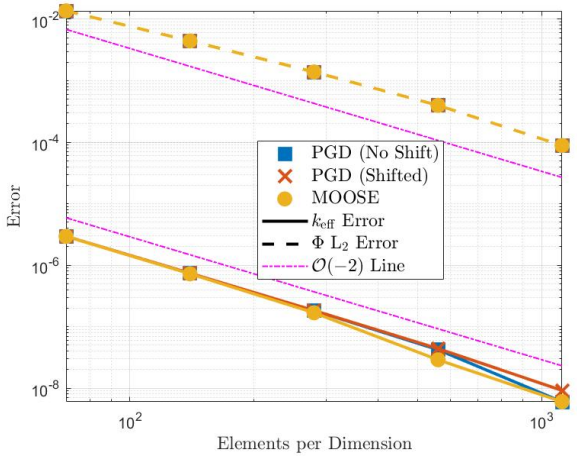


Figure 6: Relative eigenvalue and L_2 eigenvector error for the 2-D two-group problem

Table 3: Relative computational efficiency for unshifted PGD and MOOSE, 2-D two-group problem (baseline is the PGD method with shift; values < 1 denote a less efficient technique)

Elements per Dim.	PGD	MOOSE
70	0.3901	0.2598
140	0.4544	0.0447
280	0.6578	0.0260
560	0.5371	0.0135
1120	0.7804	0.0072

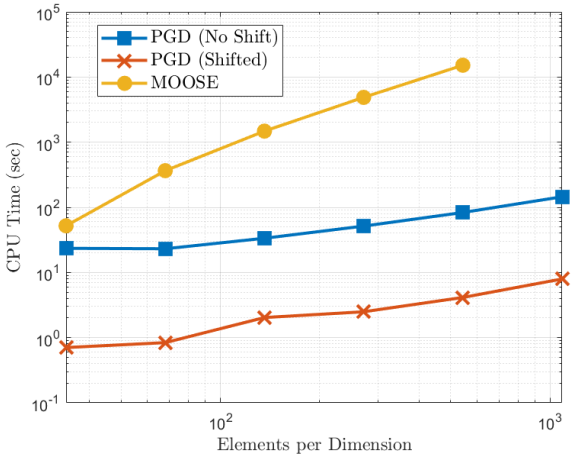


Figure 7: Run-time in seconds for each simulation of the 2-D two-group problem

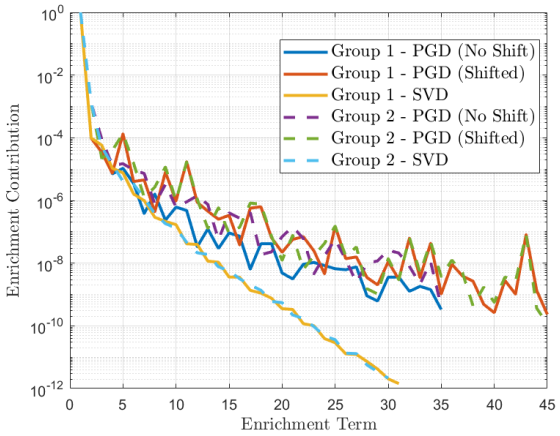


Figure 8: Enrichment contribution of converged PGD solutions and SVD of the full-order solution 2-D two-group problem

4.3. BIBLIS Benchmark Problem

These next two examples use a geometry based on the BIBLIS benchmark problem, which is shown in Fig. 9. This problem is meant to illustrate PGD performance for highly heterogeneous geometries and compare between mono-energetic and multigroup systems. The first of these examples is the single-group version and second is the two-group version.

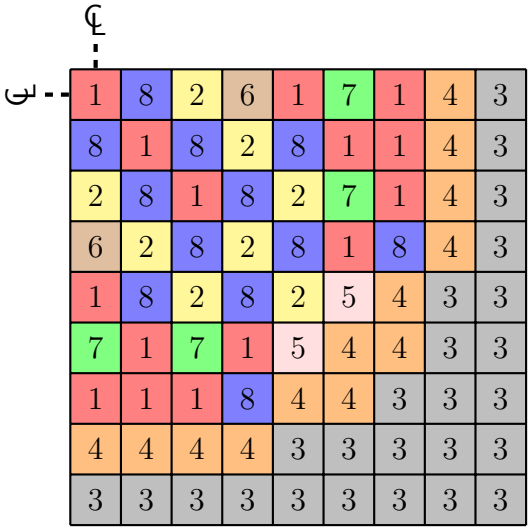


Figure 9: BIBLIS benchmark geometry: each block is 23.1226 cm × 23.1226 cm

4.3.1. One-Group Version

The material property values for one-group version of the BIBLIS benchmark can be found in Table 1 of [41]. The full-order model, evaluated using MOOSE, was run with 34 to 1,088 elements per dimension, and the PGD model was run with 34 to 2,176 elements per dimension. The eigenvalue convergence tolerance (ϵ_k) were set to 10^{-9}. A MOOSE model with 2,176 elements per dimension was used for the reference eigenvalue and eigenvector with $k_{\text{eff}} = 0.618502168$; this solution required high performance computing facilities, so the performance results for this simulation are not included.

Fig. 10 shows the resulting eigenvalue and eigenvector error for MOOSE with power iteration and PGD with and without adaptive Wielandt shift. These errors show that both the full-order and PGD models, again, have proper error convergence when the spatial mesh is refined. To compare the solution between each model, Fig. 11 shows the full-order solution at 272 elements per dimension with the difference from the PGD models at the same refinement. These plots show that the MOOSE and PGD evaluations converge to the same solution in the power iteration process.

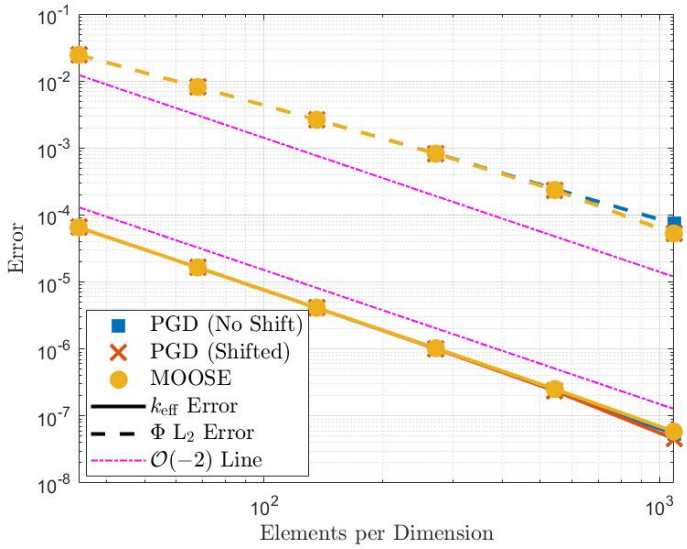
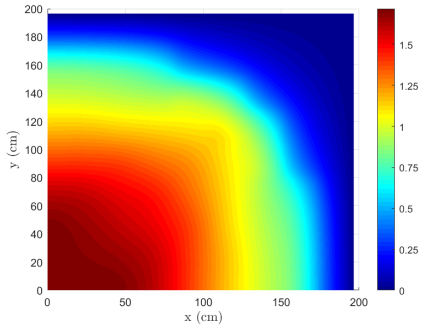
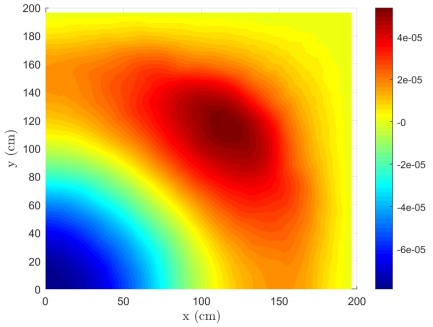


Figure 10: Relative eigenvalue and L_2 eigenvector error for one-group BIBLIS problem

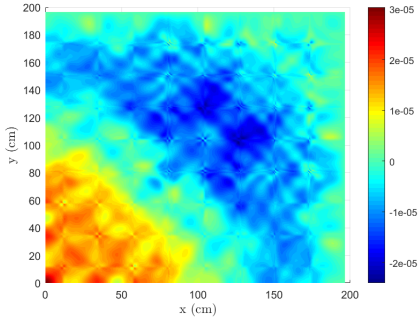
Fig. 12 shows the resulting run-time for each simulation. These run-time results show that PGD has more marginal performance than the previous problem, which is mainly due to the increase in the number of enrichment terms needed to represent the converged solution. Table 4 shows the relative computational efficiency for each refinement of the unshifted PGD model and full-order MOOSE model, definition of these numbers are described in Section 4.1. We see here that the unshifted PGD efficiency is far smaller than the shifted one, this due to the fact that there is no group coupling and inverting the shifted operator is essentially the same as inverting the unshifted one. As noted previously, the multi-D solve efficiency gets worse with mesh refinement. Fig. 13 shows the contribution of each enrichment for the PGD runs at 272 elements per dimension and the magnitude of each term of the SVD of the full-order model solution. The PGD enrichment contributions follow very closely with the SVD reference, but significantly more terms are needed fully characterize the solution when compared to the previous problem.



(a) Full-order solution (MOOSE)



(b) Difference between the PGD unshifted solution and the full-order solution



(c) Difference between the PGD shifted solution and the full-order solution

Figure 11: Full-order (MOOSE) solution and difference from PGD solutions at 272 elements per dimension, one-group BIBLIS problem

Table 4: Relative computational efficiency for unshifted PGD and MOOSE, one-group BIBLIS problem (baseline is the PGD method with shift; values < 1 denote a less efficient technique)

Elements per Dim.	PGD	MOOSE
34	0.3737	6.0264
68	0.3384	1.7264
136	0.3699	0.4959
272	0.3441	0.0613
544	0.2971	0.0282
1088	0.2633	0.0089

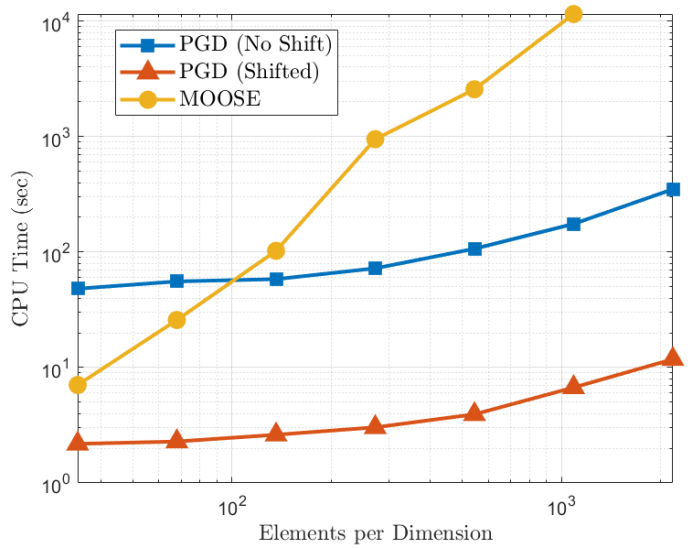


Figure 12: Run-time in seconds for each simulation of the one-group BIBLIS problem

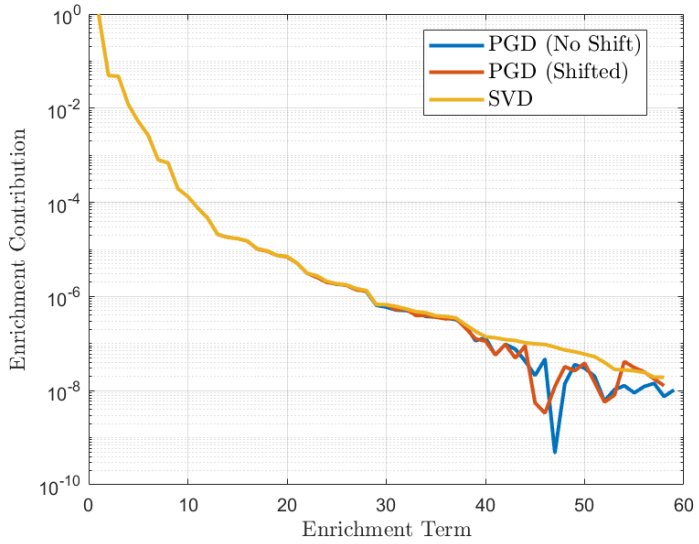


Figure 13: Enrichment contribution of converged PGD solutions and SVD of the full-order solution, one-group BIBLIS problem

4.3.2. Two-Group Version

To illustrate the performance of PGD for a multigroup example with more realistic reactor heterogeneities, we apply the PGD algorithm to the two-group version of the BIBLIS benchmark. The material property values for this problem can be found in Table 2 of [55]. The full-order model, evaluated using MOOSE, was run with 34 to 1,088 elements per dimension, and the PGD model was run with 34 to 2,176 elements per dimension. The eigenvalue convergence tolerance (ϵ_k) were set to 10^{-9}. A MOOSE model with 2,176 elements per dimension was used for the reference eigenvalue and eigenvector with $k_{\text{eff}} = 1.02510305$; this solution required high performance computing facilities, so the performance results for this simulation are not included.

Fig. 14 shows the resulting eigenvalue and eigenvector error for MOOSE with power iteration and PGD with and without adaptive Wielandt shift. These errors show that both the full-order and PGD models, again, have proper error convergence when the spatial mesh is refined. To compare the solution between each model, Fig. 15 shows the full-order solutions at 272 elements per dimension with the difference from the PGD models at the same refinement. These plots show that the MOOSE and PGD evaluations converge to approximately the same solution in the power iteration process.

Fig. 16 shows the resulting run-time for each simulation. These run-time results show that PGD has even more marginal performance than the one-group version. Table 5 shows the relative computational efficiency for each refinement of the unshifted PGD model and full-order MOOSE model. Fig. 17 shows the contribution of each enrichment for the PGD runs at 272 elements per dimension and the magnitude of each term of the SVD of the full-order model solution. SVD has similar enrichment convergence as the one-group version, but PGD does not show the same convergence. From the poorer convergence of the shifted PGD model, the difference from SVD appears to be caused by the coupling between group fluxes in the PGD linear system evaluation. The unshifted operator only has scattering, coupling from group 1 to 2, for a given

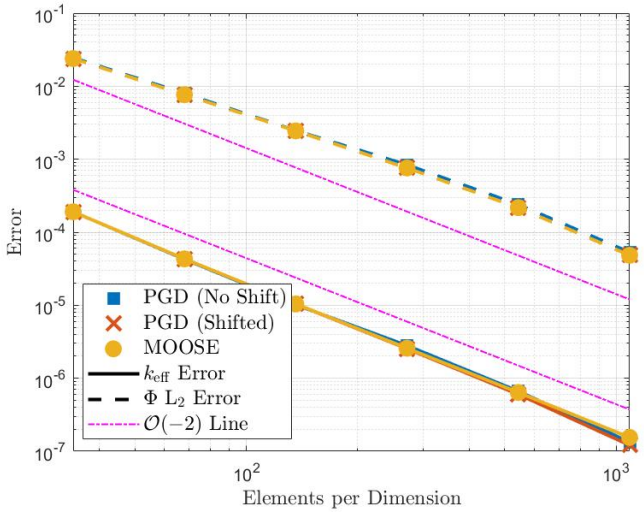


Figure 14: Relative eigenvalue and L_2 eigenvector error for two-group BIBLIS problem

source term; while the shifted operator has scattering and fission, coupling both groups together. The group-1 convergence is better than the group-2 for unshifted operator because group 1 is independent, but they are approximately the same for the shifted operator. Therefore, a combination of the heterogeneity making the solution difficult to decompose as a separated representation and the multigroup coupling makes this PGD approach a less effective technique for solving this problem.

Table 5: Relative computational efficiency for unshifted PGD and MOOSE, two-group BIBLIS problem (baseline is the PGD method with shift; values < 1 denote a less efficient technique)

Elements per Dim.	PGD	MOOSE
34	1.2906	138.3303
68	0.8220	25.4305
136	0.5271	5.9563
272	0.9528	6.5240
544	1.4271	2.4053
1088	1.2823	0.9485

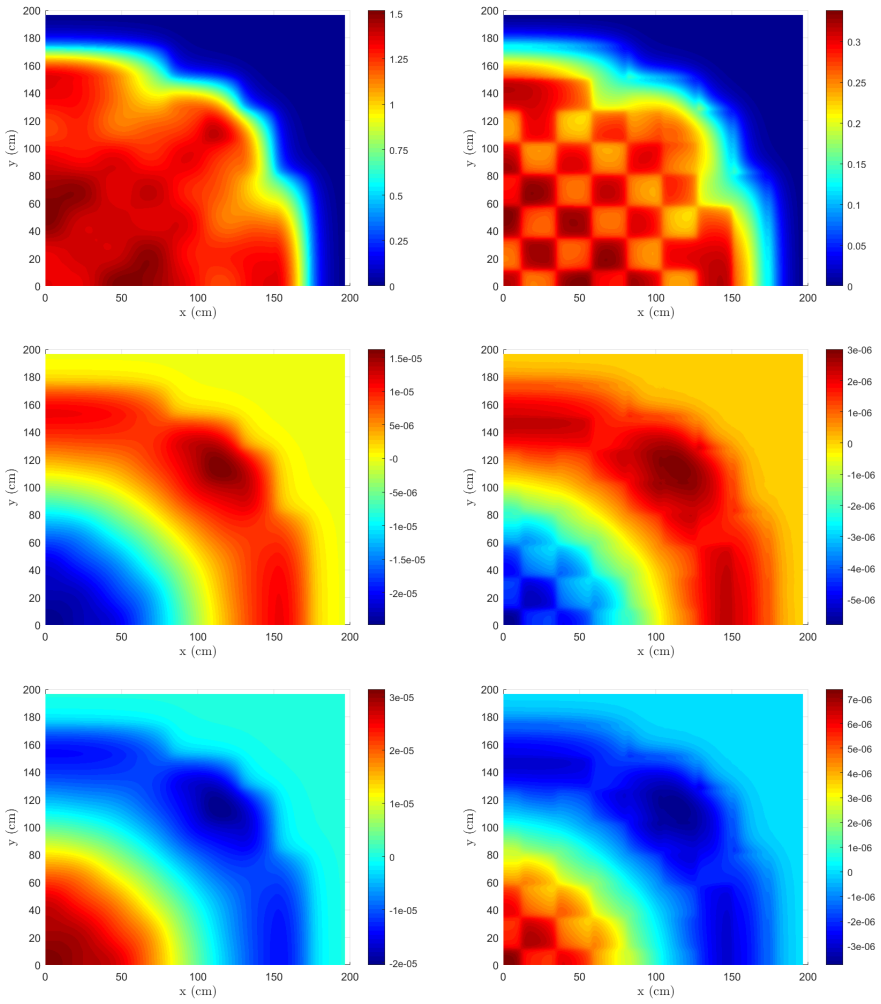


Figure 15: Full-order (MOOSE) solution and difference from PGD solutions at 272 elements per dimension, two-group BIBLIS problem. Left column: group 1 flux. Right column: group 2 flux. Top row: full-order solution. Middle row: difference with unshifted PGD solution. Bottom Row: difference with shifted PGD solution

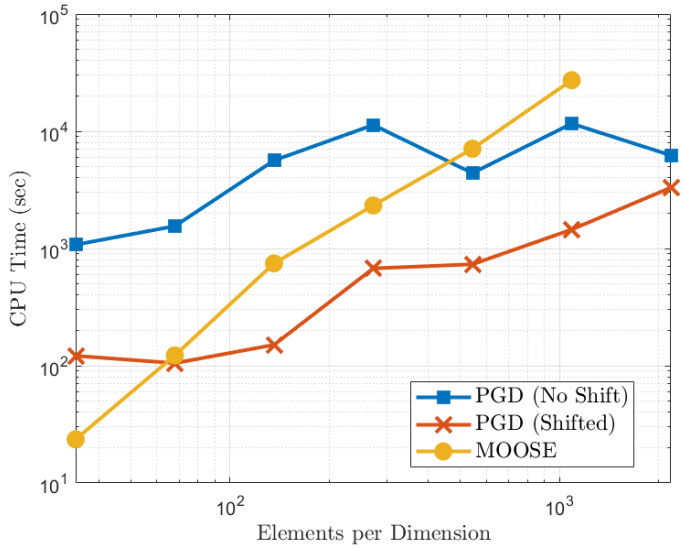


Figure 16: Run-time in seconds for each simulation of the two-group BIBLIS problem

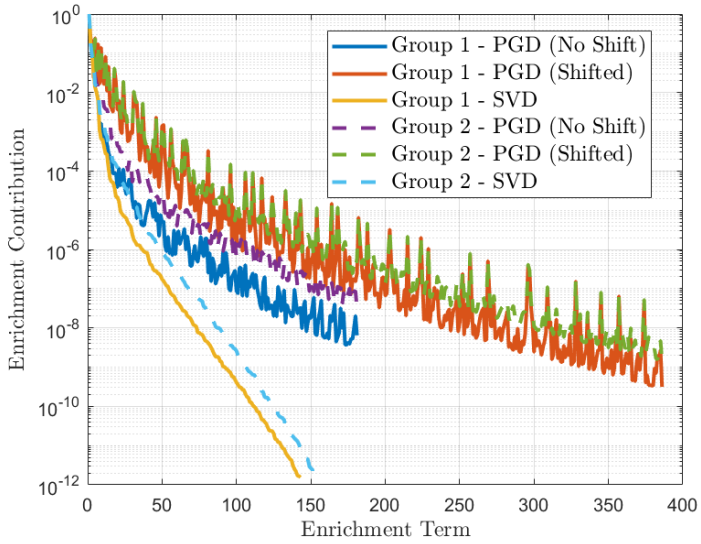


Figure 17: Enrichment contribution of converged PGD solutions and SVD of the full-order solution, two-group BIBLIS problem

4.4. Analysis of PGD Algorithm

The purpose of this section is to analyze some of the features of the implemented PGD algorithm. These include the shifting parameter, solution compression, eigenvalue projection, and the adaptive enrichment tolerance. The two-group version of BIBLIS example from Section 4.3 with 34 elements per dimension will be used for this analysis.

4.4.1. Wielandt Shift

As stated in Section 2.2, there are many different choices for the Wielandt shift parameter σ. In the previous sections, an adaptive shift was utilized, defined by Eq. (8). Here we compare the adaptive shift results with fixed shifts:

- No shift: $\sigma_0 = 0$
- Adaptive shift: $\sigma_{\text{adapt}} \longleftarrow$ Eq. (8)
- Maximum k_{eff}: $\sigma_{\min} = 1/k_{\max} = 1/3$
- Maximum infinite medium: $\sigma_\infty = 1/\max(k_\infty) = 0.8525$

$\max(k_\infty)$ is the determined by the material zone with the largest k_∞. Table 6 shows the number of iterations required for a 10^{-7} tolerance in eigenvalue and eigenvector errors, total run-time, and number of enrichments required for the last power iteration. These results show that the adaptive shift is the best method for this problem, with a tenth of the number of power iterations compared to no shift. However, these results also indicate that having a shift for this problem increases the run-time per power iteration significantly, due to the increase in the number of enrichment evaluations per power iteration. It is also worth noting that the number of enrichments increases as the efficiency of the shift improves.

This is by no means an exhaustive list of all the possible shifting parameters. See [43] and [31] for more options. We would also like to note that we attempted using a shift based on the Rayleigh quotient, explained in Section 2.2, but the PGD systems became severely ill conditioned after a few power iterations.

Table 6: Performance results of PGD with various shifting parameters for two-group BIBLIS problem

Shift Type	P.I.	k_{eff} Error	Φ Error	Run-time (s)	No. of Enrichments
σ_0	117	0.0001906	0.005983	1031	73
σ_{adapt}	11	0.0001906	0.005003	149.9	175
σ_{min}	96	0.0001907	0.006334	1523	89
σ_∞	28	0.0001906	0.005100	423.9	108

4.4.2. Solution Compression, Eigenvalue Projection, and Adaptive Enrichment Tolerance

In Sections 3.3 and 3.4, several methods are proposed to improve PGD's performance for generalized eigenvalue problems. Here we analyze the effectiveness of these methods. Five different combinations were run (with no Wiedlandt shift):

- Type 1 - All methods implemented
- Type 2 - No compression, projection, or adaptive tolerance
- Type 3 - No compression
- Type 4 - No projection
- Type 5 - No adaptive tolerance

Table 7 shows the number of iterations required for a 10^{-7} tolerance in eigenvalue and eigenvector errors, difference in run-time from Type 1, and number of enrichments required for the last power iteration. These results show that each of the methods implemented improve PGD performance significantly, without affecting the final error in the solution. Most significant is the adaptive enrichment tolerance, which shows that the method prevents "over-solving".

Table 7: Performance results of PGD with various efficiency methods turned off for two-group BIBLIS problem

Type	P.I.	k_{eff} Error	Φ Error	Δ Run-time (s)	No. of Enrichments
1	117	0.0001906	0.005983	0	73
2	139	0.0001906	0.005823	+7586	143
3	117	0.0001906	0.006157	+260	73
4	139	0.0001906	0.005957	+769	77
5	117	0.0001906	0.005970	+4143	143

5. Conclusions

In this paper, we have investigated the implementation of Proper Generalized Decomposition (PGD) to multigroup neutron diffusion criticality problems. The eigenproblems are solved using standard power iteration and accelerated power iteration with an adaptive Wielandt shift. PGD is utilized as a linear system solver to obtain the eigenvector at each iteration. In the theory sections, we discuss the PGD solution process and its implementation within the power iteration process. In order to avoid re-evaluating the PGD operators for each spatial dimension, a PGD-favorable decomposition of the multigroup diffusion operators is discussed for an efficient implementation of the technique. This is particularly needed for highly heterogeneous geometries, such as the ones of found in modern nuclear cores. Because the iterative PGD process is located in the innermost loop (power iteration being the traditional outer loop), several techniques to reduce the computational burden of the PGD iterations have been proposed, including compression of the current PGD iteration, solution projection to optimize the current PGD, and adaptive tolerance criteria.

The results for this paper involve four different criticality problems. For comparison purposes, a non-PGD approach is also employed, solved using the MOOSE FEM library. The first test case is a four-group 3D homogeneous bare cube. PGD shows far superior performance compared to MOOSE; this problems admits an (analytic) solution that is separable in space, so the PGD

evaluation requires very few terms to converge. The second problem is a two-group 2D two-material geometry. The domain is large is mostly consist of one material. Again, PGD shows superior performance compared to MOOSE for this problem because relatively few number of enrichments are needed to represent each power iteration's eigenvector; this problem was used in previous PGD applications [42]. The third and fourth problems have the same highly heterogeneous 2D geometry, representative of a more realistic core layout. For these problems, PGD shows a much more marginal performance compared to the previous examples, especially for the two-group version (fourth problem).

The difference in the PGD performance for these more heterogeneous problems highlights that the effectiveness of PGD may hinge on the problem's heterogeneity. The material-layout dependence is evident when comparing the results from the two-material problem and the two-group BIBLIS problem. Both of the eigenvectors have a similar enrichment convergence, as assessed by an Singular Value Decomposition analysis. This is compounded in the multigroup case, for example when comparing the 1-g and 2-g BIBLIS results where about 4 times more enrichments are needed in the PGD process than the ideal value (computed by performing an SVD of the solution). This increase in the number of enrichments renders PGD slower because added enrichments require more alternating direction iterations, each one being more computationally demanding than the previous one (residual evaluation with more terms).

We conclude that PGD can be an effective technique for a certain class of neutron diffusion problems. Some intuition must be held on the nature of the resulting solution in order to determine if PGD will have better performance than a full multi-D model evaluation. In general, this PGD approach performs more poorly for highly heterogeneous geometries. For more homogeneous geometries, we have found PGD to always outperform its full-order modeling counterpart in a significant manner.

For future work, more analysis and verification is needed on PGD for reactor-physics problems including more complex geometries and problems with finer group structures. Although, we expect the PGD approach to struggle in solving

multigroup criticality problems for highly heterogeneous geometries. Some recent advances in PGD techniques may mitigate this difficulty; see, for instance, the adaptive subspace methods [58], multigrid algorithms for canonical products [59], and domain decomposition [60]. Furthermore, PGD algorithms should be compared with more traditional reactor physics codes, such as TRIVAC and PARCS. Finally, we note that the PGD approach can be extended beyond the spatial coordinates to create space-energy decompositions; see, for instance, [61]

6. Acknowledgments

This material is based upon work supported under an Integrated University Program Graduate Fellowship of the Department of Energy, Office of Nuclear Energy (contract reference DE-NE0000112). Any opinions, findings, conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the Department of Energy, Office of Nuclear Energy.

References

References

[1] K. Pearson, Liii. on lines and planes of closest fit to systems of points in space, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2 (11) (1901) 559–572.

[2] R. Pinnau, Model Reduction via Proper Orthogonal Decomposition, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, Ch. 5, pp. 95–109. doi:10.1007/978-3-540-78841-6_5.

[3] S. Volkwein, Proper orthogonal decomposition: Theory and reduced-order modelling, Lecture Notes, University of Konstanz 4 (4) (2013).

[4] J. Lumley, A. Yaglom, V. Tatarski, Atmospheric turbulence and radio wave propagation, Jounal of Computational Chemistry 23 (13) (1967) 1236–1243.

[5] G. Berkooz, P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annual review of fluid mechanics 25 (1) (1993) 539–575.

[6] C. W. Rowley, Model reduction for fluids, using balanced proper orthogonal decomposition, International Journal of Bifurcation and Chaos 15 (03) (2005) 997–1013.

[7] Y. Cao, J. Zhu, Z. Luo, I. Navon, Reduced-order modeling of the upper tropical pacific ocean model using proper orthogonal decomposition, Computers & Mathematics with Applications 52 (8) (2006) 1373 – 1386.

[8] R. Ştefănescu, I. Navon, Pod/deim nonlinear model order reduction of an adi implicit shallow water equations model, Journal of Computational Physics 237 (2013) 95 – 114.

[9] G. Kerschen, J.-C. Golinval, A. F. Vakakis, L. A. Bergman, The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview, Nonlinear Dynamics 41 (1) (2005) 147–169.

[10] A. Corigliano, M. Dossi, S. Mariani, Model order reduction and domain decomposition strategies for the solution of the dynamic elastic-plastic structural problem, Computer Methods in Applied Mechanics and Engineering 290 (2015) 127 – 155.

[11] A. Sartori, D. Baroli, A. Cammi, D. Chiesa, L. Luzzi, R. Ponciroli, E. Previtali, M. E. Ricotti, G. Rozza, M. Sisti, Comparison of a modal method and a proper orthogonal decomposition approach for multi-group time-dependent reactor spatial kinetics, Annals of Nuclear Energy 71 (2014) 217 – 229.

[12] Y. Bang, H. S. Abdel-Khalik, J. M. Hite, Hybrid reduced order modeling applied to nonlinear models, International Journal for Numerical Methods in Engineering 91 (9) (2012) 929–949. doi:10.1002/nme.4298.

[13] A. Buchan, A. Calloo, M. Goffin, S. Dargaville, F. Fang, C. Pain, I. Navon, A pod reduced order model for resolving angular direction in neutron/photon transport problems, Journal of Computational Physics 296 (2015) 138 – 157.

[14] D. Huang, H. Abdel-Khalik, C. Rabiti, F. Gleicher, Dimensionality reducibility for multi-physics reduced order modeling, Annals of Nuclear Energy 110 (2017) 526 – 540. doi:10.1016/j.anucene.2017.06.045.

[15] H. S. Abdel-Khalik, Y. Bang, C. Wang, Overview of hybrid subspace methods for uncertainty quantification, sensitivity analysis, Annals of Nuclear Energy 52 (2013) 28 – 46, nuclear Reactor Safety Simulation and Uncertainty Analysis. doi:10.1016/j.anucene.2012.07.020.

[16] A. G. Buchan, C. C. Pain, F. Fang, I. M. Navon, A pod reduced-order model for eigenvalue problems with application to reactor physics, International Journal for Numerical Methods in Engineering 95 (12) (2013) 1011–1032.

[17] P. German, J. C. Ragusa, Reduced-order modeling of parameterized multigroup diffusion k-eigenvalue problems, Annals of Nuclear Energy 134 (2019) 144 – 157. doi:10.1016/j.anucene.2019.05.049.

[18] R. Everson, L. Sirovich, Karhunen-loeve procedure for gappy data, J. Opt. Soc. Am. A 12 (1995) 1657–1664.

[19] F. Chinesta, R. Keunings, A. Leygue, The Proper Generalized Decomposition for Advanced Numerical Simulations, Vol. 1, Springer, 2014.

[20] F. Chinesta, A. Ammar, E. Cueto, Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models, Archives of Computational Methods in Engineering 17 (4) (2010) 327–350. doi:10.1007/s11831-010-9049-y.

[21] Z. M. Prince, J. C. Ragusa, Parametric uncertainty quantification using proper generalized decomposition applied to neutron diffusion, Interna-

tional Journal for Numerical Methods in Engineering (2019) 1–23doi:10.1002/nme.6077.

[22] J. J. Duderstadt, L. J. Hamilton, Nuclear reactor analysis, Wiley, 1976.

[23] W. Stacey, Nuclear Reactor Physics, John Wiley & Sons, 2007.

[24] E. L. Wachspress, Iterative Solution of Elliptic Systems and Application to the Neutron Diffusion Equations of Reactor Physics, Prentice-Hall, 1966.

[25] H. Park, D. A. Knoll, C. K. Newman, Nonlinear acceleration of transport criticality problems, Nuclear Science and Engineering 172 (1) (2012) 52–65. doi:10.13182/NSE11-81.

[26] Y. Saad, Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems, Mathematics of Computation 42 (166) (1984) 567–588.

[27] T. Rivlin, Chebyshev Polynomials: From Approximation Theory to Algebra and Number Theory, Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts, Wiley, 1990.

[28] I. S. Duff, J. A. Scott, Computing selected eigenvalues of sparse unsymmetric matrices using subspace iteration, ACM Trans. Math. Softw. 19 (2) (1993) 137–159. doi:10.1145/152613.152614.

[29] Y. Saad, Numerical solution of large nonsymmetric eigenvalue problems, Computer Physics Communications 53 (1) (1989) 71 – 90. doi:10.1016/0010-4655(89)90149-5.

[30] J. H. Wilkinson (Ed.), The Algebraic Eigenvalue Problem, Oxford University Press, Inc., New York, NY, USA, 1988.

[31] B. C. Yee, B. Kochunas, E. W. Larsen, Y. Xu, Space-dependent wielandt shifts for multigroup diffusion eigenvalue problems, Nuclear Science and Engineering 188 (2) (2017) 140–159. doi:10.1080/00295639.2017.1350001.

[32] V. Vidal, G. Verdú, D. Ginestar, J. L. Muñoz Cobo, Variational acceleration for subspace iteration method. application to nuclear power reactors, International Journal for Numerical Methods in Engineering 41 (3) (1998) 391–407.

[33] G. Verdú, R. Miro, D. Ginestar, V. Vidal, The implicit restarted arnoldi method, an efficient alternative to solve the neutron diffusion equation, Annals of Nuclear Energy 26 (7) (1999) 579 – 593. doi:10.1016/S0306-4549(98)00077-2.

[34] J. Munoz-Cobo, R. Miró, A. Wysocki, A. Soler, 3d calculation of the lambda eigenvalues and eigenmodes of the two-group neutron diffusion equation by coarse-mesh nodal methods, Progress in Nuclear Energy 110 (2019) 393 – 409. doi:10.1016/j.pnucene.2018.10.008.

[35] R. B. Morgan, Generalizations of davidson's method for computing eigenvalues of large nonsymmetric matrices, Journal of Computational Physics 101 (2) (1992) 287 – 291. doi:10.1016/0021-9991(92)90006-K.

[36] G. Verdú, D. Ginestar, R. Miró, V. Vidal, Using the jacobi-davidson method to obtain the dominant lambda modes of a nuclear power reactor, Annals of Nuclear Energy 32 (11) (2005) 1274 – 1296. doi:10.1016/j.anucene.2005.03.002.

[37] C. Subramanian, S. V. Criekingen, V. Heuveline, F. Nataf, P. Havé, The davidson method as an alternative to power iterations for criticality calculations, Annals of Nuclear Energy 38 (12) (2011) 2818 – 2823. doi:10.1016/j.anucene.2011.08.018.

[38] D. A. Knoll, H. Park, C. Newman, Acceleration of k-eigenvalue/criticality calculations using the jacobian-free newton-krylov method, Nuclear Science and Engineering 167 (2) (2011) 133–140. doi:10.13182/NSE09-89.

[39] V. Mahadevan, J. Ragusa, Novel hybrid scheme to compute several domi-

nant eigenmodes for reactor analysis problems, in: Proc. Int. Conf. Physics of Reactors (PHYSOR), Interlaken, Switzerland, 2008, p. 8.

[40] D. F. Gill, Y. Y. Azmy, Newton's method for solving k-eigenvalue problems in neutron diffusion theory, Nuclear Science and Engineering 167 (2) (2011) 141–153. doi:10.13182/NSE09-98.

[41] S. González-Pintor, D. Ginestar, G. Verdú, Using proper generalized decomposition to compute the dominant mode of a nuclear reactor, Mathematical and Computer Modelling 57 (7) (2013) 1807 – 1815. doi:10.1016/j.mcm.2011.11.066.

[42] J. P. Senecal, W. Ji, Characterization of the proper generalized decomposition method for fixed-source diffusion problems, Annals of Nuclear Energy 126 (2019) 68 – 83.

[43] Y. Saad, Numerical Methods for Large Eigenvalue Problems, Society for Industrial and Applied Mathematics, 2011. doi:10.1137/1.9781611970739.

[44] B. N. Parlett, The Symmetric Eigenvalue Problem, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1998.

[45] F. Chinesta, P. Ladevèze, Separated Representations and PGD-Based Model Reduction: Fundamentals and Applications, Vol. 1, Springer, 2014.

[46] F. Chinesta, E. Cueto, PGD-Based Modeling of Materials, Structures and Processes, Vol. 554, Springer, 2014.

[47] F. Chinesta, A. Ammar, A. Leygue, R. Keunings, An overview of the proper generalized decomposition with applications in computational rheology, Journal of Non-Newtonian Fluid Mechanics 166 (11) (2011) 578 – 592. doi:10.1016/j.jnnfm.2010.12.012.

[48] F. Chinesta, P. Ladevèze, E. Cueto, A short review on model order reduction based on proper generalized decomposition, Archives of Computational Methods in Engineering 18 (4) (2011) 395.

[49] A. Dumon, C. Allery, A. Ammar, Simulation of heat and mass transport in a square lid-driven cavity with proper generalized decomposition (PGD), Numerical Heat Transfer, Part B: Fundamentals 63 (1) (2013) 18–43. doi:10.1080/10407790.2012.724991.

[50] A. Ammar, B. Mokdad, F. Chinesta, R. Keunings, A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids, Journal of non-Newtonian fluid Mechanics 139 (3) (2006) 153–176.

[51] D. González, A. Ammar, F. Chinesta, E. Cueto, Recent advances on the use of separated representations, International Journal for Numerical Methods in Engineering 81 (5) (2010) 637–659.

[52] L. A. Semenza, E. E. Lewis, E. C. Rossow, The application of the finite element method to the multigroup neutron diffusion equation, Nuclear Science and Engineering 47 (3) (1972) 302–310. doi:10.13182/NSE72-A22416.

[53] Y. Wang, S. Schunert, V. Laboure, Rattlesnake theory manual, Tech. rep., Idaho National Laboratory, Idaho Falls, ID (2018).

[54] J. N. Reddy, An introduction to the finite element method, McGraw-Hill Mechanical Engineering, 2005.

[55] E. Múller, Z. Weiss, Benchmarking with the multigroup diffusion high-order response matrix method, Annals of Nuclear Energy 18 (9) (1991) 535 – 544. doi:10.1016/0306-4549(91)90098-I.

[56] D. Gaston, C. Newman, G. Hansen, D. Lebrun-Grandie, Moose: A parallel computational framework for coupled systems of nonlinear equations, Nucl. Engrg. Design 239 (2009) 1768 – 1778.

[57] A. Hébert, Applied reactor physics, Presses inter Polytechnique, 2009.

[58] A. Nouy, A priori model reduction through proper generalized decomposition for solving time-dependent partial differential equations, Computer

Methods in Applied Mechanics and Engineering 199 (23) (2010) 1603 – 1626.

[59] H. D. Sterck, K. Miller, An adaptive algebraic multigrid algorithm for low-rank canonical tensor decomposition, SIAM Journal on Scientific Computing 35 (1) (2013) B1–B24.

[60] A. Huerta, E. Nadal, F. Chinesta, Proper generalized decomposition solutions within a domain decomposition strategy, International Journal for Numerical Methods in Engineering 113 (13) (2018) 1972–1994. doi:10.1002/nme.5729.

[61] Z. M. Prince, J. C. Ragusa, Space-energy separated representations for multigroup neutron diffusion using proper generalized decompositions, Annals of Nuclear Energy*In Press* (2019).

Appendix A. Material Properties of Chosen Problems

Table A.8: Material properties of 2-D two-group problem

Material	Group	D^g (cm)	Σ_a^g (cm^{-1})	$\nu\Sigma_f^g$ (cm^{-1})	$\Sigma_s^{g\leftarrow 1}$ (cm^{-1})
1	1	1.264	7.86e-4	4.562e-3	0.0
	2	0.9328	4.10e-3	0.0	7.368e-3
2	1	1.310	0.0	0.0	0.0
	2	0.8695	2.117e-4	0.0	1.018e-2

$\chi^1 = 1.0$, $\chi^2 = 0.0$

Table A.9: Material properties for one-group BIBLIS benchmark problem

Material	D (cm)	Σ_a (cm^{-1})	$\nu\Sigma_f$ (cm^{-1})
1	1.4360	0.0095042	0.0058708
2	1.4366	0.0096785	0.0061908
3	1.3200	0.0026562	0.0000000
4	1.4389	0.0103630	0.0074527
5	1.4381	0.0100030	0.0061908
6	1.4385	0.0101320	0.0064285
7	1.4389	0.0101650	0.0061908
8	1.4393	0.0102940	0.0061908

Table A.10: Material properties of two-group BIBLIS benchmark problem

Material	Group	D^g (cm)	Σ_a^g (cm^{-1})	$\nu\Sigma_f^g$ (cm^{-1})	$\Sigma_s^{g\leftarrow 1}$ (cm^{-1})
1	1	1.4360	0.0095042	0.0058708	0
	2	0.3635	0.0750580	0.0960670	0.017754
2	1	1.4366	0.0096785	0.0061908	0
	2	0.3636	0.0784360	0.1035800	0.017621
3	1	1.3200	0.0026562	0	0
	2	0.2772	0.0715960	0	0.023106
4	1	1.4389	0.0103630	0.0074527	0
	2	0.3638	0.0914080	0.1323600	0.017101
5	1	1.4381	0.0100030	0.0061908	0
	2	0.3665	0.0848280	0.1035800	0.017290
6	1	1.4385	0.0101320	0.0064285	0
	2	0.3665	0.0873140	0.1091100	0.017192
7	1	1.4389	0.0101650	0.0061908	0
	2	0.3679	0.0880240	0.1035800	0.017125
8	1	1.4393	0.0102940	0.0064285	0
	2	0.3680	0.0905100	0.1091100	0.017027

$\chi^1 = 1.0$, $\chi^2 = 0.0$