A low-rank power iteration scheme for neutron transport criticality problems

Jonas Kusch[a], Benjamin Whewell[b], Ryan McClarren[b], Martin Frank[a]

[a] *Karlsruhe Institute of Technology, Karlsruhe, Germany, {jonas.kusch, martin.frank}@kit.edu*
[b] *University of Notre Dame, Notre Dame, Indiana, USA, {bwhewell, rmcclarr}@nd.edu*

Abstract

Computing effective eigenvalues for neutron transport often requires a fine numerical resolution. The main challenge of such computations is the high memory effort of classical solvers, which limits the accuracy of chosen discretizations. In this work, we derive a method for the computation of effective eigenvalues when the underlying solution has a low-rank structure. This is accomplished by utilizing dynamical low-rank approximation (DLRA), which is an efficient strategy to derive time evolution equations for low-rank solution representations. The main idea is to interpret the iterates of the classical inverse power iteration as pseudo-time steps and apply the DLRA concepts in this framework. In our numerical experiment, we demonstrate that our method significantly reduces memory requirements while achieving the desired accuracy. Analytic investigations show that the proposed iteration scheme inherits the convergence speed of the inverse power iteration, at least for a simplified setting.

Keywords: Dynamical low-rank approximation, kinetic equations, neutron transport, unconventional integrator

Introduction

In analyzing nuclear systems the eigenvalue problem that describes the behavior of a neutron-induced fission chain reaction known as the k-eigenvalue problem is of fundamental importance. The magnitude of the dominant eigenvalue indicates whether the fission chain reaction will 1) ultimately diverge, 2) reach a constant, non-zero steady state, or 3) decay to zero. These cases are known as supercritical, critical, or subcritical systems, respectively.
Given that the dominant (i.e., maximal) eigenvalue determines the character of the system, the inverse power iteration method (and its variations) is the most commonly applied method. In this method the neutron transport operator is repeatedly applied to an initial guess of the eigenvector in a similar manner to solving a steady-state problem where the righthand side changes with each iteration [26].
There are a variety of mathematical models for the transport of neutrons including Monte Carlo [25], discrete ordinates (S_N) [22], spherical harmonics (P_N) [1], and simplified P_N methods [27]. The workhorse model for nuclear systems is the multigroup diffusion method [34] where the neutron energies are discretized into finite energy ranges known as groups, and an elliptic, diffusion operator is used to approximate the migration of neutrons. Such a model will require storage of a solution that has a size that is the product of the energy and spatial degrees of freedom. When dealing with a highly heterogeneous system such as a nuclear reactor, the number of spatial degrees of freedom can be enormous [36], and to correctly approximate the reaction rates the number of energy degrees of freedom can easily reach several hundred [14]. For this reason, many problems are approximated by coarsened descriptions in space and energy to enable exploration of design space in engineering applications. These coarse problem descriptions can be accurate for systems that have either historical data to calibrate to or where past high-fidelity calculations can be used to inform discretizations. Nevertheless, the promise of small modular nuclear systems, interplanetary exploration scale reactors, and other emerging technologies may not be able to rely on these techniques.

To enable high-fidelity calculations we seek numerical techniques that have reduced memory and computational costs. An efficient method with low memory requirements for time-dependent problems is dynamical low-rank approximation (DLRA). This method has been introduced for matrix-valued solutions in [16] and an extension to tensors is given in [17]. Its main idea is to represent and evolve the solution on a manifold of rank r functions. DLRA yields evolution equations for the individual factors of the solution [16], which can be solved numerically with standard methods. Robust integrators for these equations are the matrix projector-splitting integrator [23] and the unconventional integrator [3]. The original use of dynamical low-rank approximation focuses on matrix ordinary differential equations. Note that partial differential equations can be brought into such a form by performing a discretization of the phase-space except for time. In [7], the derivation of DLRA has been extended to function spaces, i.e., the evolution equations can be derived before discretizing the problem. This approach does not only provide the ability to derive stable discretizations [19], but in the radiation transfer context allows an efficient implementation of scattering [19, 20]. Problems in which dynamical low-rank is successfully applied to reduce memory and computational costs are, e.g., kinetic theory [7, 8, 31, 30, 9, 5, 6, 13, 20] as well as uncertainty quantification [10, 28, 29, 33, 18]. Furthermore, DLRA allows for adaptive model refinement [4, 2, 32], where the main idea is to pick the rank of the solution representation adaptively. Recently, an approach to employ DLRA for computing rightmost eigenpairs has been proposed in [12].

In this work, we combine dynamical low-rank approximation and the neutron diffusion equations to obtain an inverse power iteration scheme with low memory requirements. For this, we treat the updates of the power iteration as pseudo-timesteps. DLRA is then applied to the iteration scheme. I.e., the solution is represented as a product of rank r matrices and the resulting DLRA update equations yield an iteration scheme of each matrix. The memory requirement of the solution representation reduces from $N_x \times G$, where N_x is the number of spatial cells and G is the number of energy groups to $N_x \times r + G \times r$, where $r \ll N_x, G$.

This paper is structured as follows: After the introduction, we provide an overview over the used concepts in Section 1: Section 1.1 presents a discretization of the full k-eigenvalue problem for matrix solutions $\boldsymbol{\phi} \in \mathbb{R}^{N_x \times G}$. Section 1.2 shortly reviews important concepts of dynamical low-rank approximation used in this work. In Section 2 we employ dynamical low-rank approximation to derive a memory efficient iteration scheme for the k-eigenvalue problem. Section 3 gives a convergence proof for the scheme in a simplified setting. Numerical results are presented in Section 4 before concluding with Section 5.

1. Background

In the following, we give a short overview of the concepts employed in this work. Furthermore, we use this section to write the energy-dependent neutron diffusion equation in a compact matrix notation, which simplifies the derivation of DLRA evolution equations.

1.1. k-eigenvalue computation

Our main goal is to determine the maximum eigenvalue k_{eff} as well as its corresponding eigenfunction ϕ of the neutron transport k-eigenvalue problem. The corresponding multigroup approximation reads

$$-\nabla \cdot D_g(r)\nabla\phi_g(r) + \Sigma_{t,g}(r)\phi_g(r) = \frac{\chi_g}{k_{\text{eff}}} \sum_{g'} \nu\Sigma_{f,g'}(r)\phi_{g'}(r) + \sum_{g'} \Sigma_{s,g',g}(r)\phi_g(r), \tag{1}$$

where $r \in \Omega \subset \mathbb{R}^d$ is the spatial variable, $\Sigma_{t,g}(r)$ is the total cross section of energy group g at spatial position r, $\Sigma_{f,g}(r)$ is the fission cross-section and $\Sigma_{s,g',g}(r)$ is the scattering cross section between groups g and g'. The variable $D_g(r)$ denotes the material diffusion coefficient at position r and energy group g. Furthermore, ν is the mean number of particles produced per fission event and χ_g is the fission neutron distribution function for group g. We are interested in computing $\phi_g(r)$ which is the integral of the scalar flux over the energy range of group g at position r corresponding to the maximal eigenvalue k_{eff}. For sake of readability, we omit boundary conditions and state them whenever needed.

In the following, our aim is to state a numerical discretization of (1), which treats the scalar flux at given spatial points as a matrix. That is, for a spatial grid $r_1, \cdots, r_{N_x}$ and energy groups $g \in \{1, \cdots, G\}$ we have $\boldsymbol{\phi} = (\phi_{jg})_{j,g=1}^{N_x,G}$, where $\phi_{jg} = \phi_g(x_j)$. Such a formulation enables the use of dynamical low-rank approximation to reduce memory requirements. First, to later be able to derive efficient update formulations for the low-rank factors, we require all material coefficients to have a moderate rank. Note that the rank of these material coefficients equals the number of materials N_m, since

$$D_g(r) = \sum_{\ell=1}^{N_m} \rho_\ell(r) D_g^{(\ell)}, \tag{2}$$

where $D_g^{(\ell)}$ is the diffusion coefficient (or any other material coefficient) for material ℓ and energy group g. The functions $\rho_\ell(r)$ denote densities of material ℓ. In the following, we use the low-rank formulation of the material parameters which later will facilitate the derivation of the DLRA evolution equations. When evaluating D_g at cell interfaces $r_{j+1/2} = (r_j + r_{j+1})/2$, one needs to approximate the material coefficient through its harmonic mean, i.e.,

$$D_g(r_{j+1/2}) =: D_{g,j+1/2} = 2\frac{D_{g,j} D_{g,j+1}}{D_{g,j} + D_{g,j+1}} = 2\frac{\sum_{\ell,k} \rho_j^{(\ell)} D_g^{(\ell)} \rho_{j+1}^{(k)} D_g^{(k)}}{\sum_\ell (\rho_j^{(\ell)} + \rho_{j+1}^{(\ell)}) D_g^{(\ell)}}.$$

Noting that only two terms in the sums can be non-zero, we get

$$D_{g,j+1/2} = \sum_{\ell,k=1}^{N_m} \rho_j^{(\ell)} \rho_{j+1}^{(k)} \left(\rho_j^{(\ell)} + \rho_{j+1}^{(k)}\right) \frac{D_g^{(\ell)} D_g^{(k)}}{D_g^{(\ell)} + D_g^{(k)}}. \tag{3}$$

The diffusion operator in one dimension is discretized through

$$\nabla \cdot D_g(r) \nabla \phi_g(r)\Big|_{r=r_j} \approx (\boldsymbol{D}(g)\boldsymbol{\phi}_g)_j$$

where $\boldsymbol{\phi}_g \in \mathbb{R}^{N_x}$ collects the scalar flux at all spatial cells. The matrix $\boldsymbol{D}(g) \in \mathbb{R}^{N_x \times N_x}$ has values

$$\begin{aligned} D_{j,j\pm1}(g) &= \frac{1}{\Delta r \cdot V_j} D_{g,j\pm1/2} S_{j\pm1/2}, \\ D_{j,j}(g) &= -\frac{1}{\Delta r \cdot V_j} \left[D_{g,j+1/2} S_{j+1/2} + D_{g,j-1/2} S_{j-1/2}\right], \end{aligned}$$

where Δr is the size of each radial element. The surface area between cell j and $j \pm 1$ is denoted by $S_{j\pm1/2}$ and the volume of cell j is denoted by V_j. The choice of these terms defines the spatial geometry. In our numerical experiments, we look at spherical domains, where we have $V_j = \frac{4\pi}{3}(r_{i+1/2}^3 - r_{i-1/2}^3)$ and $S_{j\pm1/2} = 4\pi r_{i\pm1/2}^2$. Using (3) in the above definition of $\boldsymbol{D}(g)$, lets us write $\boldsymbol{D}(g)\boldsymbol{\phi}_g$ as

$$\boldsymbol{D}(g)\boldsymbol{\phi}_g = \sum_{\ell,k=1}^{N_m} \frac{D_g^{(\ell)} D_g^{(k)}}{D_g^{(\ell)} + D_g^{(k)}} \boldsymbol{D}^{(\ell,k)} \boldsymbol{\phi}_g,$$

where we use

$$\begin{aligned} D_{j,j\pm1}^{(\ell,k)} &= \pm \frac{\rho_j^{(\ell)} \rho_{j\pm1}^{(k)}}{\Delta r \cdot V_j} (\rho_\ell(r_j) + \rho_k(r_{j\pm1})) S_{j\pm1/2}, \\ D_{j,j}^{(\ell,k)} &= -\frac{1}{\Delta r \cdot V_j} \left[\rho_j^{(\ell)} \rho_{j+1}^{(k)} \left(\rho_j^{(\ell)} + \rho_{j+1}^{(k)}\right) S_{j+1/2} + \rho_j^{(\ell)} \rho_{j-1}^{(k)} \left(\rho_j^{(\ell)} + \rho_{j-1}^{(k)}\right) S_{j-1/2}\right]. \end{aligned}$$

Let us further use the diagonal matrix $\boldsymbol{\rho}^{(\ell)} \in \mathbb{R}^{N_x \times N_x}$ with entries $\rho^{(\ell)}_{jj} = \rho_\ell(r_j)$ to write

$$\frac{\chi_g \nu}{k} \sum_{g'} \Sigma_{f,g'}(r)\phi_{g'}(r) = \frac{\chi_g \nu}{k} \sum_{g',\ell} \rho_\ell(r_j)\Sigma^{(\ell)}_{f,g'}\phi_{g'}(r_j) = \frac{\chi_g \nu}{k} \sum_{g',\ell} \Sigma^{(\ell)}_{f,g'}\boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}_{g'},$$
$$\sum_{g'} \Sigma_{s,g',g}(r_j)\phi_g(r_j) = \sum_{g',\ell} \rho_\ell(r_j)\Sigma^{(\ell)}_{s,g',g}\phi_g(r) = \sum_{g',\ell} \Sigma^{(\ell)}_{s,g',g}\boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}_{g'}.$$

Then, for a given group g, the diffusion equation reads

$$-\sum_{\ell,k=1}^{N_m} \frac{D^{(\ell)}_g D^{(k)}_g}{D^{(\ell)}_g + D^{(k)}_g} \boldsymbol{D}^{(\ell,k)}\boldsymbol{\phi}_g + \sum_{\ell=1}^{N_m} \Sigma^{(\ell)}_{t,g}\boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}_g = \frac{\chi_g \nu}{k_{\text{eff}}} \sum_{g',\ell} \Sigma^{(\ell)}_{f,g'}\boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}_{g'} + \sum_{g',\ell} \Sigma^{(\ell)}_{s,g',g}\boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}_{g'}.$$

We can write this term as a bigger system for $\boldsymbol{\phi} \in \mathbb{R}^{N_x \times G}$. For this, we define $\widetilde{\boldsymbol{\Sigma}}^{(\ell)}_f = \left(\chi_g \nu \Sigma^{(\ell)}_{f,g'}\right)_{g,g'=1}^{G}$ and the diagonal matrices $\boldsymbol{M}^{(\ell,k)} \in \mathbb{R}^{G\times G}$ with entries $M^{(\ell,k)}_{gg} = D^{(\ell,k)}_g$ as well as $\boldsymbol{\Sigma}^{(\ell)}_t \in \mathbb{R}^{G\times G}$ with $\Sigma^{(\ell)}_{t,gg} = \Sigma^{(\ell)}_{t,g}$. Then we have

$$-\sum_{\ell,k} \boldsymbol{D}^{(\ell,k)}\boldsymbol{\phi}\boldsymbol{M}^{(\ell,k)} + \sum_{\ell} \boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}\boldsymbol{\Sigma}^{(\ell)}_t = \frac{1}{k_{\text{eff}}} \sum_{\ell} \boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}\widetilde{\boldsymbol{\Sigma}}^{(\ell)}_f + \sum_{\ell} \boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}\boldsymbol{\Sigma}^{(\ell)}_s.$$

We solve the above equation for $\boldsymbol{\phi}$ and k_{eff} with an inverse power iteration. For this, an iteration index is assigned to $\boldsymbol{\phi}$ and an update of the scalar flux is given by

$$-\sum_{\ell,k} \boldsymbol{D}^{(\ell,k)}\widetilde{\boldsymbol{\phi}}^{n+1}\boldsymbol{M}^{(\ell,k)} + \sum_{\ell} \boldsymbol{\rho}^{(\ell)}\widetilde{\boldsymbol{\phi}}^{n+1}\left(\boldsymbol{\Sigma}^{(\ell)}_t - \boldsymbol{\Sigma}^{(\ell)}_s\right) = \sum_{\ell} \boldsymbol{\rho}^{(\ell)}\boldsymbol{\phi}^n\widetilde{\boldsymbol{\Sigma}}^{(\ell)}_f. \tag{4}$$

The iteration method to determine the eigenvalue k then reads

1. Start with initial guess $\boldsymbol{\phi}^0_g$ with $g = 1, \cdots, G$.
2. Compute $\widetilde{\boldsymbol{\phi}}^{n+1}$ from (4).
3. Set $k_{n+1} = \|\widetilde{\boldsymbol{\phi}}^{n+1}\|$ and $\boldsymbol{\phi}^{n+1} = \widetilde{\boldsymbol{\phi}}^{n+1}/k_{n+1}$
4. If $|k_{n+1} - k_n| \leq \varepsilon$ stop, else set $n \leftarrow n+1$ and repeat from step 2.

Note that the chosen norm $\|\cdot\|$ denotes the Frobenius norm.

1.2. Dynamical low-rank approximation

In the following, the dynamical low-rank approximation [16] will be reviewed. For a more detailed derivation of this method, we recommend reading the initial presentation [16] as well as the derivation of stable integrators to the derived evolution equations [23, 3]. A comprehensive discussion and summary of DLRA can for example be found in [35, Chapter 1]. In this work, DLRA will be employed to derive a memory efficient iteration scheme. DLRA is commonly derived for time dependent problems of the form

$$\dot{\mathbf{u}}(t) = \mathbf{F}(\mathbf{u}(t)), \tag{5}$$

where $\boldsymbol{u} \in \mathbb{R}^{N\times M}$ and $\boldsymbol{F} : \mathbb{R}^{N\times M} \to \mathbb{R}^{N\times M}$. To reduce computational complexity as well as memory requirements, DLRA represents and evolves the solution to (5) on a manifold of rank r functions. In our case, the solution is represented by an SVD-like decomposition of the form

$$\mathbf{u}(t) \approx \mathbf{X}(t)\mathbf{S}(t)\mathbf{W}(t)^T, \tag{6}$$

where $\mathbf{X} \in \mathbb{R}^{N\times r}$ and $\mathbf{W} \in \mathbb{R}^{M\times r}$ can be interpreted as basis matrices with orthogonal columns and $\boldsymbol{S} \in \mathbb{R}^{r\times r}$ is a (not necessarily diagonal) coefficient matrix. We denote the set of matrices of the form (6),

i.e., the set of rank r matrices, as $\mathcal{M}_r$. In order to derive evolution equations for the two basis matrices and their corresponding coefficient matrix, we wish to find $\mathbf{u}_r \in \mathcal{M}_r$ which fulfills

$$\dot{\mathbf{u}}_r(t) \in T_{\mathbf{u}_r(t)}\mathcal{M}_r \qquad \text{such that} \qquad \|\dot{\mathbf{u}}_r(t) - \mathbf{F}(\mathbf{u}(t))\| = \min. \tag{7}$$

Here, $T_{\mathbf{u}_r(t)}\mathcal{M}_r$ denotes the tangent space of $\mathcal{M}_r$ at $\mathbf{u}_r(t)$. Then, following [16, Proposition 2.1], the condition (7) leads to update equations

$$\dot{\boldsymbol{S}} = \boldsymbol{X}^T \mathbf{F}(\mathbf{u}(t))\boldsymbol{W}, \tag{8a}$$

$$\dot{\boldsymbol{X}} = (\boldsymbol{I} - \boldsymbol{X}\boldsymbol{X}^T)\mathbf{F}(\mathbf{u}(t))\boldsymbol{W}\boldsymbol{S}^{-1}, \tag{8b}$$

$$\dot{\boldsymbol{W}} = (\boldsymbol{I} - \boldsymbol{W}\boldsymbol{W}^T)\mathbf{F}(\mathbf{u}(t))^T\boldsymbol{X}\boldsymbol{S}^{-T}. \tag{8c}$$

Due to the inverse coefficient matrix on the right-hand side of the above equations, this formulation is not robust under small eigenvalues. Robust integrators for the low-rank factors are the *projector-splitting integrator* [23] as well as the *unconventional integrator* [3]. In this work, we make use of the latter, however a derivation for the projector-splitting integrator is possible as well. Let us assume that a factorized solution is available at time t_n as $\mathbf{X}^n$, $\mathbf{S}^n$ and $\mathbf{W}^n$. The unconventional integrator updates the factorized solution from time t_n to t_{n+1} via

1. K**-step**: Update $\mathbf{X}^n$ to $\mathbf{X}^{n+1}$ via

$$\dot{\mathbf{K}}(t) = \mathbf{F}(\mathbf{K}(t)\mathbf{W}^{n,T})\mathbf{W}^n, \qquad \mathbf{K}(t_n) = \mathbf{X}^n\mathbf{S}^n. \tag{9}$$

   Determine $\mathbf{X}^{n+1}$ with $\mathbf{K}(t_{n+1}) = \mathbf{X}^{n+1}\mathbf{R}$ and store $\mathbf{M} = \mathbf{X}^{n+1,T}\mathbf{X}^n$.
2. L**-step**: Update $\mathbf{W}^n$ to $\mathbf{W}^{n+1}$ via

$$\dot{\mathbf{L}}(t) = \mathbf{X}^{n,T}\mathbf{F}(\mathbf{X}^n\mathbf{L}(t)), \qquad \mathbf{L}(t_n) = \mathbf{S}^n\mathbf{W}^{n,T}. \tag{10}$$

   Determine $\mathbf{W}^{n+1}$ with $\mathbf{L}^{n+1} = \mathbf{W}^{n+1}\widetilde{\mathbf{R}}$ and store $\mathbf{N} = \mathbf{W}^{n+1,T}\mathbf{W}^n$.
3. S**-step**: Update $\mathbf{S}^n$ to $\mathbf{S}^{n+1}$ via

$$\dot{\mathbf{S}}(t) = \mathbf{X}^{n+1,T}\mathbf{F}(\mathbf{X}^1\mathbf{S}(t)\mathbf{W}^{n+1,T})\mathbf{W}^{n+1}, \qquad \mathbf{S}(t_n) = \mathbf{M}\mathbf{S}^n\mathbf{N}^T \tag{11}$$

   and set $\mathbf{S}^{n+1} = \mathbf{S}(t_{n+1})$.

The updated solution is then given as $\boldsymbol{u}(t_{n+1}) = \boldsymbol{X}^{n+1}\boldsymbol{S}^{n+1}\boldsymbol{W}^{n+1,T}$. When using a forward Euler time discretization we obtain the equations

$$\mathbf{K}^{n+1} = \boldsymbol{K}^n + \Delta t\mathbf{F}(\mathbf{K}^n\mathbf{W}^{n,T})\mathbf{W}^n, \qquad \mathbf{K}^n = \mathbf{X}^n\mathbf{S}^n, \tag{12a}$$

$$\mathbf{L}^{n+1} = \mathbf{L}^n + \Delta t\mathbf{X}^{n,T}\mathbf{F}(\mathbf{X}^n\mathbf{L}^n), \qquad \mathbf{L}^n = \mathbf{S}^n\mathbf{W}^{n,T}, \tag{12b}$$

$$\mathbf{S}^{n+1} = \mathbf{S}^n + \Delta t\mathbf{X}^{n+1,T}\mathbf{F}(\mathbf{X}^{n+1}\mathbf{X}^{n+1,T}\mathbf{X}^n\mathbf{S}^n\mathbf{W}^{n,T}\mathbf{W}^{n+1}\mathbf{W}^{n+1,T})\mathbf{W}^{n+1}. \tag{12c}$$

Hence, from a given initial condition $\mathbf{u}(t=0)$, we can compute $\mathbf{X}^0$,$\mathbf{S}^0$ and $\mathbf{X}^0$ through a singular value decomposition truncated at a given rank r and use (12) to time update the facrized solution in time. In the following, we use the above scheme to define a low-rank inverse power iteration for the k-eigenvalue problem.

2. Dynamical low-rank approximation for the inverse power iteration

2.1. Iteration equations for low-rank factors

In the following, we derive the DLRA evolution equations of the unconventional integrator. The equations of the matrix projector-splitting integrator take a similar form. Let us choose a rank r representation for $\boldsymbol{\phi}^n$, which reads $\boldsymbol{\phi}^n \approx \boldsymbol{X}^n\boldsymbol{S}^n\boldsymbol{W}^n$. Here, $\boldsymbol{X}^n \in \mathbb{R}^{N_x \times r}$, $\boldsymbol{S}^n \in \mathbb{R}^{r\times r}$ and $\boldsymbol{W}^n \in \mathbb{R}^{G\times r}$. The main idea

of our derivation is to interpret the iteration index n as a pseudo-time, i.e., we can use (12) to define an iteration on the factoring matrices. Let us first plug the rank r representation into (4). Furthermore, we define $\boldsymbol{\Sigma}^{(\ell)} := \boldsymbol{\Sigma}_t^{(\ell)} - \boldsymbol{\Sigma}_s^{(\ell)}$. Then we have

$$-\sum_{\ell,k} \boldsymbol{D}^{(\ell,k)} \widetilde{\boldsymbol{\phi}}^{n+1} \boldsymbol{M}^{(\ell,k)} + \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \widetilde{\boldsymbol{\phi}}^{n+1} \boldsymbol{\Sigma}^{(\ell)} = \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{X}^n \boldsymbol{S}^n \boldsymbol{W}^{n,T} \widetilde{\boldsymbol{\Sigma}}_f^{(\ell)}. \tag{13}$$

This form simplifies the presentation of the following K, L and S-step derivations:
K-step: We define $\boldsymbol{K}^n = \boldsymbol{X}^n \boldsymbol{S}^n$, set $\widetilde{\boldsymbol{\phi}}^{n+1} = \boldsymbol{K}^{n+1} \boldsymbol{W}^{n,T}$ and multiply (13) with $\boldsymbol{W}^n$ from the right. Then, we have

$$-\sum_{\ell,k} \boldsymbol{D}^{(\ell,k)} \boldsymbol{K}^{n+1} \boldsymbol{W}^{n,T} \boldsymbol{M}^{(\ell,k)} \boldsymbol{W}^n + \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^{n+1} \boldsymbol{W}^{n,T} \boldsymbol{\Sigma}^{(\ell)} \boldsymbol{W}^n = \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^n \boldsymbol{W}^{n,T} \widetilde{\boldsymbol{\Sigma}}_f^{(\ell)} \boldsymbol{W}^n.$$

The terms

$$\widehat{\boldsymbol{\Sigma}}_{f,n}^{(\ell)} = \boldsymbol{W}^{n,T} \widetilde{\boldsymbol{\Sigma}}_f^{(\ell)} \boldsymbol{W}^n, \qquad \widehat{\boldsymbol{M}}_n^{(\ell,k)} = \boldsymbol{W}^{n,T} \boldsymbol{M}^{(\ell,k)} \boldsymbol{W}^n, \qquad \widehat{\boldsymbol{\Sigma}}_n^{(\ell)} = \boldsymbol{W}^{n,T} \boldsymbol{\Sigma}^{(\ell)} \boldsymbol{W}^n$$

can be computed in $O(r^2 \cdot G^2)$ operations. With the above definitions, the K-step reads

$$-\sum_{\ell,k} \boldsymbol{D}^{(\ell,k)} \boldsymbol{K}^{n+1} \widehat{\boldsymbol{M}}_n^{(\ell,k)} + \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^{n+1} \widehat{\boldsymbol{\Sigma}}_n^{(\ell)} = \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^n \widehat{\boldsymbol{\Sigma}}_{f,n}^{(\ell)}. \tag{14}$$

L-step: We define $\boldsymbol{L}^n = \boldsymbol{S}^n \boldsymbol{W}^{n,T}$, set $\widetilde{\boldsymbol{\phi}}^{n+1} = \boldsymbol{X}^n \boldsymbol{L}^{n+1}$ and multiply (13) with $\boldsymbol{X}^{n,T}$ from the left. Then, we get

$$-\sum_{\ell,k} \boldsymbol{X}^{n,T} \boldsymbol{D}^{(\ell,k)} \boldsymbol{X}^n \boldsymbol{L}^{n+1} \boldsymbol{M}^{(\ell,k)} + \sum_{\ell} \boldsymbol{X}^{n,T} \boldsymbol{\rho}^{(\ell)} \boldsymbol{X}^n \boldsymbol{L}^{n+1} \boldsymbol{\Sigma}^{(\ell)} = \sum_{\ell} \boldsymbol{X}^{n,T} \boldsymbol{\rho}^{(\ell)} \boldsymbol{X}^n \boldsymbol{L}^n \widetilde{\boldsymbol{\Sigma}}_f^{(\ell)}.$$

The terms

$$\widehat{\boldsymbol{\rho}}_n^{(\ell)} = \boldsymbol{X}^{n,T} \boldsymbol{\rho}^{(\ell)} \boldsymbol{X}^n, \qquad \widehat{\boldsymbol{D}}_n^{(\ell,k)} = \boldsymbol{X}^{n,T} \boldsymbol{D}^{(\ell,k)} \boldsymbol{X}^n$$

can be computed in $O(r^2 \cdot N_x^2)$ operations. With the above definitions, the L-step reads

$$-\sum_{\ell,k} \widehat{\boldsymbol{D}}_n^{(\ell,k)} \boldsymbol{L}^{n+1} \boldsymbol{M}^{(\ell,k)} + \sum_{\ell} \widehat{\boldsymbol{\rho}}_n^{(\ell)} \boldsymbol{L}^{n+1} \boldsymbol{\Sigma}^{(\ell)} = \sum_{\ell} \widehat{\boldsymbol{\rho}}^{(\ell)} \boldsymbol{L}^n \widetilde{\boldsymbol{\Sigma}}_f^{(\ell)}. \tag{15}$$

S-step: We define $\boldsymbol{S} = \boldsymbol{X}^{n+1,T} \boldsymbol{X}^n \boldsymbol{S}^n \boldsymbol{W}^{n,T} \boldsymbol{W}^{n+1}$, set $\widetilde{\boldsymbol{\phi}}^{n+1} = \boldsymbol{X}^{n+1} \boldsymbol{S}^{n+1} \boldsymbol{W}^{n+1,T}$ and multiply (13) with $\boldsymbol{X}^{n+1,T}$ from the left and $\boldsymbol{W}^{n+1}$ from the right. Then, we get

$$\begin{aligned} &-\sum_{\ell,k} \boldsymbol{X}^{n+1,T} \boldsymbol{D}^{(\ell,k)} \boldsymbol{X}^{n+1} \boldsymbol{S}^{n+1} \boldsymbol{W}^{n+1,T} \boldsymbol{M}^{(\ell,k)} \boldsymbol{W}^{n+1} \\ &\quad + \sum_{\ell} \boldsymbol{X}^{n+1,T} \boldsymbol{\rho}^{(\ell)} \boldsymbol{X}^{n+1} \boldsymbol{S}^{n+1} \boldsymbol{W}^{n+1,T} \boldsymbol{\Sigma}^{(\ell)} \boldsymbol{W}^{n+1} = \sum_{\ell} \boldsymbol{X}^{n+1,T} \boldsymbol{\rho}^{(\ell)} \boldsymbol{X}^{n+1} \boldsymbol{S} \boldsymbol{W}^{n+1,T} \widetilde{\boldsymbol{\Sigma}}_f^{(\ell)} \boldsymbol{W}^{n+1}. \end{aligned}$$

Let us reuse our above definitions, but evaluate them at iteration $n+1$ instead of n. Then, the S-step reads

$$-\sum_{\ell} \widehat{\boldsymbol{D}}_{n+1}^{(\ell)} \boldsymbol{S}^{n+1} \widehat{\boldsymbol{M}}_{n+1}^{(\ell)} + \sum_{\ell} \widehat{\boldsymbol{\rho}}_{n+1}^{(\ell)} \boldsymbol{S}^{n+1} \widehat{\boldsymbol{\Sigma}}_{n+1}^{(\ell)} = \sum_{\ell} \widehat{\boldsymbol{\rho}}_{n+1}^{(\ell)} \boldsymbol{S} \widehat{\boldsymbol{\Sigma}}_{f,n+1}^{(\ell)}. \tag{16}$$

2.2. Inversions

In the presented method, we frequently need to solve systems of the form

$$-\sum_{\ell} \boldsymbol{A}^{(\ell)} \boldsymbol{x} \boldsymbol{B}^{(\ell)} + \sum_{\ell} \boldsymbol{C}^{(\ell)} \boldsymbol{x} \boldsymbol{D}^{(\ell)} = \boldsymbol{y}.$$

Let us assume general dimensions N and M. Then $\boldsymbol{A}^{(\ell)}, \boldsymbol{C}^{(\ell)} \in \mathbb{R}^{N\times N}$, $\boldsymbol{B}^{(\ell)}, \boldsymbol{D}^{(\ell)} \in \mathbb{R}^{M\times M}$, and we wish to determine $\boldsymbol{x} \in \mathbb{R}^{N\times M}$ for a right-hand-side $\boldsymbol{y} \in \mathbb{R}^{N\times M}$. Written in index notation, we obtain

$$-\sum_{\ell}\sum_{j,\alpha} A^{(\ell)}_{ij} x_{j\alpha} B^{(\ell)}_{\alpha\beta} + \sum_{\ell}\sum_{j,\alpha} C^{(\ell)}_{ij} x_{j\alpha} D^{(\ell)}_{\alpha\beta} = y_{i\beta}.$$

Let us rearrange this to a matrix-vector product:

$$\sum_{j,\alpha}\sum_{\ell} \left(-A^{(\ell)}_{ij} B^{(\ell)}_{\alpha\beta} + C^{(\ell)}_{ij} D^{(\ell)}_{\alpha\beta}\right) \tilde{x}_{\alpha+(j-1)r} = \tilde{y}_{\beta+(i-1)r},$$

where $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}} \in \mathbb{R}^{N\cdot M}$ are the matrices $\boldsymbol{x}$ and $\boldsymbol{y}$ rearranged to vectors. Hence, we can define the matrix $\boldsymbol{\mathcal{E}} \in \mathbb{R}^{N\cdot M\times N\cdot M}$ with entries

$$\mathcal{E}_{\beta+(i-1)r,\alpha+(j-1)r} = \sum_{\ell} \left(-A^{(\ell)}_{ij} B^{(\ell)}_{\alpha\beta} + C^{(\ell)}_{ij} D^{(\ell)}_{\alpha\beta}\right)$$

and then solve the linear system $\boldsymbol{\mathcal{E}}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{y}}$. Note that for the K-step (14), we have $N = N_x$ and $M = r$. The L-step (15) has $N = G$ and $M = r$ and the S-step (18) has dimensions $N = M = r$. Opposed to the original problem, which inverts a matrix $\boldsymbol{E} \in \mathbb{R}^{N_x\cdot G\times N_x\cdot G}$, we now need to invert three significantly smaller subproblems (assuming $r \ll \max\{N_x, G\}$). It should be noted that many implementations of the multigroup diffusion equations solve the equations on a group-by-group iteration based on the Gauss-Seidel method. These methods require the storage of a matrix of dimensions $N_x \times N_x$ and a storage of a solution vector of size $N_x \cdot G$. Nevertheless, our implementation is able to avoid the building of matrices at each step and because it is not necessarily matrix free, it could utilize a wider variety of preconditioning strategies.

2.3. Algorithm

The full algorithm takes the following form:

1. K-**step**: Update $\mathbf{X}^n$ to $\mathbf{X}^{n+1}$ via

$$-\sum_{\ell,k} \boldsymbol{D}^{(\ell,k)} \boldsymbol{K}^{n+1} \widehat{\boldsymbol{M}}^{(\ell,k)}_n + \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^{n+1} \widehat{\boldsymbol{\Sigma}}^{(\ell)}_n = \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^{n} \widehat{\boldsymbol{\Sigma}}^{(\ell)}_{f,n}.$$

   Determine $\mathbf{X}^{n+1}$ with $\mathbf{K}^{n+1} = \mathbf{X}^{n+1}\mathbf{R}$ and store $\mathbf{N}_x = \mathbf{X}^{n+1,T}\mathbf{X}^n$.

2. L-**step**: Update $\mathbf{W}^n$ to $\mathbf{W}^{n+1}$ via

$$-\sum_{\ell,k} \widehat{\boldsymbol{D}}^{(\ell,k)}_n \boldsymbol{L}^{n+1} \boldsymbol{M}^{(\ell,k)} + \sum_{\ell} \widehat{\boldsymbol{\rho}}^{(\ell)}_n \boldsymbol{L}^{n+1} \boldsymbol{\Sigma}^{(\ell)} = \sum_{\ell} \widehat{\boldsymbol{\rho}}^{(\ell)} \boldsymbol{L}^{n} \widetilde{\boldsymbol{\Sigma}}^{(\ell)}_f.$$

   Determine $\mathbf{W}^{n+1}$ with $\mathbf{L}^{n+1} = \mathbf{W}^{n+1}\widetilde{\mathbf{R}}$ and store $\mathbf{N}_E = \mathbf{W}^{n+1,T}\mathbf{W}^n$.

3. S-**step**: Update $\mathbf{S}^n$ to $\mathbf{S}^{n+1}$ via

$$-\sum_{\ell} \widehat{\boldsymbol{D}}^{(\ell,k)}_{n+1} \widetilde{\boldsymbol{S}}^{n+1} \widehat{\boldsymbol{M}}^{(\ell,k)}_{n+1} + \sum_{\ell} \widehat{\boldsymbol{\rho}}^{(\ell)}_{n+1} \widetilde{\boldsymbol{S}}^{n+1} \widehat{\boldsymbol{\Sigma}}^{(\ell)}_{n+1} = \sum_{\ell} \widehat{\boldsymbol{\rho}}^{(\ell)}_{n+1} \boldsymbol{S} \widehat{\boldsymbol{\Sigma}}^{(\ell)}_{f,n+1}. \tag{17}$$

   with $\boldsymbol{S} = \boldsymbol{N}_x \boldsymbol{S}^n \boldsymbol{N}_E^T$. Set $k^{n+1} = \|\widetilde{\boldsymbol{S}}^{n+1}\|$ and $\boldsymbol{S}^{n+1} = \widetilde{\boldsymbol{S}}^{n+1}/k^{n+1}$.

4. If $|k_{n+1} - k_n| \leq \varepsilon$ stop, else set $n \leftarrow n+1$ and repeat.

An extension of this algorithm to rank adaptivity according to [2] is straight forward. We state the rank adaptive algorithm here, even though our numerical computations have all been done with the fixed rank integrator.

1. K**-step**: Update $\mathbf{X}^n \in \mathbb{R}^{N_x \times r_n}$ to $\mathbf{X}^{n+1} \in \mathbb{R}^{N_x \times 2r_n}$ via

$$-\sum_{\ell,k} \boldsymbol{D}^{(\ell,k)} \boldsymbol{K}^{n+1} \widehat{\boldsymbol{M}}_n^{(\ell,k)} + \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^{n+1} \widehat{\boldsymbol{\Sigma}}_n^{(\ell)} = \sum_{\ell} \boldsymbol{\rho}^{(\ell)} \boldsymbol{K}^n \widehat{\boldsymbol{\Sigma}}_{f,n}^{(\ell)}.$$

   Determine $\mathbf{X}^{n+1}$ with $[\mathbf{K}^{n+1}, \boldsymbol{X}^n] = \mathbf{X}^{n+1}\mathbf{R}$ and store $\mathbf{N}_x = \mathbf{X}^{n+1,T}\mathbf{X}^n \in \mathbb{R}^{2r_n \times r_n}$.

2. L**-step**: Update $\mathbf{W}^n \in \mathbb{R}^{G \times r_n}$ to $\mathbf{W}^{n+1} \in \mathbb{R}^{G \times 2r_n}$ via

$$-\sum_{\ell,k} \widehat{\boldsymbol{D}}_n^{(\ell,k)} \boldsymbol{L}^{n+1} \boldsymbol{M}^{(\ell,k)} + \sum_{\ell} \widehat{\boldsymbol{\rho}}_n^{(\ell)} \boldsymbol{L}^{n+1} \boldsymbol{\Sigma}^{(\ell)} = \sum_{\ell} \widehat{\boldsymbol{\rho}}^{(\ell)} \boldsymbol{L}^n \widetilde{\boldsymbol{\Sigma}}_f^{(\ell)}.$$

   Determine $\mathbf{W}^{n+1}$ with $[\mathbf{L}^{n+1}, \mathbf{W}^n] = \mathbf{W}^{n+1}\widetilde{\mathbf{R}}$ and store $\mathbf{N}_E = \mathbf{W}^{n+1,T}\mathbf{W}^n \in \mathbb{R}^{2r_n \times r_n}$.

3. S**-step**: Update $\mathbf{S}^n \in \mathbb{R}^{r_n \times r_n}$ to $\widehat{\mathbf{S}}^{n+1} \in \mathbb{R}^{2r_n \times 2r_n}$ via

$$-\sum_{\ell,k} \widehat{\boldsymbol{D}}_{n+1}^{(\ell,k)} \widehat{\boldsymbol{S}}^{n+1} \widehat{\boldsymbol{M}}_{n+1}^{(\ell,k)} + \sum_{\ell} \widehat{\boldsymbol{\rho}}_{n+1}^{(\ell)} \widetilde{\boldsymbol{S}}^{n+1} \widehat{\boldsymbol{\Sigma}}_{n+1}^{(\ell)} = \sum_{\ell} \widehat{\boldsymbol{\rho}}_{n+1}^{(\ell)} \boldsymbol{S} \widehat{\boldsymbol{\Sigma}}_{f,n+1}^{(\ell)}. \tag{18}$$

   with $\boldsymbol{S} = \boldsymbol{N}_x \boldsymbol{S}^n \boldsymbol{N}_E^T \in \mathbb{R}^{2r_n \times 2r_n}$.

4. **Truncation**: Determine the SVD $\widehat{\boldsymbol{S}}^{n+1} = \widehat{\boldsymbol{P}}\widehat{\boldsymbol{\Sigma}}\widehat{\boldsymbol{Q}}^\top$ where $\widehat{\boldsymbol{\Sigma}} = \mathrm{diag}(\sigma_j)$. For a given tolerance ϑ, choose the new rank $r_{n+1} \leq 2r_n$ such that

$$\left(\sum_{j=r_{n+1}+1}^{2r} \sigma_j^2 \right)^{1/2} \leq \vartheta.$$

   Compute the new factors for the approximation of ϕ^{n+1} as follows: Let $\boldsymbol{S}_1$ be the $r_1 \times r_1$ diagonal matrix with the r_1 largest singular values and let $\boldsymbol{P}_1 \in \mathbb{R}^{2r \times r_1}$ and $\boldsymbol{Q}_1 \in \mathbb{R}^{2r \times r_1}$ contain the first r_1 columns of $\widehat{\boldsymbol{P}}$ and $\widehat{\boldsymbol{Q}}$, respectively. Finally, set $\boldsymbol{U}_1 = \widehat{\boldsymbol{U}}\boldsymbol{P}_1 \in \mathbb{R}^{m \times r_1}$ and $\boldsymbol{V}_1 = \widehat{\boldsymbol{V}}\boldsymbol{Q}_1 \in \mathbb{R}^{n \times r_1}$.

5. Set $k^{n+1} = \|\widetilde{\boldsymbol{S}}^{n+1}\|$ and $\boldsymbol{S}^{n+1} = \widetilde{\boldsymbol{S}}^{n+1}/k^{n+1}$. If $|k_{n+1} - k_n| \leq \varepsilon$ stop, else set $n \leftarrow n+1$ and repeat.

3. Convergence in a simplified setting

Let us remark the following properties of the presented iteration method: If the iteration converges to a so-called steady state, the error of the maximal eigenvalue depends on the low-rank structure of the steady state solution. I.e., the accuracy of the method depends on the low-rank structure of the full problem. To investigate convergence, we investigate a simplified setting for the power iteration. Assume that we wish to determine the maximal λ such that $\boldsymbol{A}\boldsymbol{\phi}\boldsymbol{B} = \lambda \boldsymbol{C}\boldsymbol{\phi}\boldsymbol{D}$, i.e.,

$$\boldsymbol{\phi} = \lambda \boldsymbol{A}^{-1}\boldsymbol{C}\boldsymbol{\phi}\boldsymbol{D}\boldsymbol{B}^{-1}. \tag{19}$$

Note that this iteration gives the maximal eigenvalue of the operator $\mathcal{T}(\boldsymbol{\phi}) := \boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{\phi}\boldsymbol{B}\boldsymbol{D}^{-1}$. Moreover, note that in this setting, we obtain eigenmatrices rather than eigenvectors. Then, the power iteration for the full problem becomes

$$\boldsymbol{\phi}^{n+1} = \frac{\boldsymbol{A}^{-1}\boldsymbol{C}\boldsymbol{\phi}^n\boldsymbol{D}\boldsymbol{B}^{-1}}{\|\boldsymbol{A}^{-1}\boldsymbol{C}\boldsymbol{\phi}^n\boldsymbol{D}\boldsymbol{B}^{-1}\|} = \frac{\widehat{\boldsymbol{C}}\boldsymbol{\phi}^n\widehat{\boldsymbol{D}}}{\|\widehat{\boldsymbol{C}}\boldsymbol{\phi}^n\widehat{\boldsymbol{D}}\|}$$

where $\widehat{\boldsymbol{C}} := \boldsymbol{A}^{-1}\boldsymbol{C}$ and $\widehat{\boldsymbol{D}} := \boldsymbol{D}\boldsymbol{B}^{-1}$. Let us assume that $\widehat{\boldsymbol{C}} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}$ and $\widehat{\boldsymbol{D}} = \boldsymbol{U}^{-1}\boldsymbol{\Sigma}\boldsymbol{U}$. In this case, the eigenmatrix corresponding to the maximal eigenvalue of $\mathcal{T}$ is of rank 1, namely $\boldsymbol{v}_1\boldsymbol{u}_1^T$. Then, we write

our initial iterate as

$$\phi^0_{jg} = \sum_{\ell,k} V_{j\ell}\alpha_{\ell k}U_{kg}$$

or in terms of matrices $\boldsymbol{\phi}^n = \boldsymbol{V}\boldsymbol{\alpha}^n\boldsymbol{U}$. Plugging this into the iteration scheme yields

$$\boldsymbol{\phi}^1 = \frac{\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}\boldsymbol{\phi}^0\boldsymbol{U}^{-1}\boldsymbol{\Sigma}\boldsymbol{U}}{\|\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}\boldsymbol{\phi}^0\boldsymbol{W}^{-1}\boldsymbol{\Sigma}\boldsymbol{U}\|} = \frac{\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{\alpha}\boldsymbol{\Sigma}\boldsymbol{U}}{\|\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{\alpha}\boldsymbol{\Sigma}\boldsymbol{U}\|}.$$

Applying this multiple times gives

$$\boldsymbol{\phi}^{n+1} = \frac{\boldsymbol{V}\boldsymbol{\Lambda}^n\boldsymbol{\alpha}\boldsymbol{\Sigma}^n\boldsymbol{U}}{\|\boldsymbol{V}\boldsymbol{\Lambda}^n\boldsymbol{\alpha}\boldsymbol{\Sigma}^n\boldsymbol{U}\|}.$$

Let us collect columns of $\boldsymbol{V}$ and $\boldsymbol{U}$ in vectors $\boldsymbol{v}_i$ and $\boldsymbol{u}_i$. Furthermore, let us define $\mu_{ij} = \frac{\lambda_i\sigma_j}{\lambda_1\sigma_1}$. Then, we can rewrite the above expression as

$$\boldsymbol{\phi}^{n+1} = \frac{\sum_{i,j}\lambda_i^n\sigma_j^n\alpha_{ij}\boldsymbol{v}_i\boldsymbol{u}_j^T}{\|\sum_{i,j}\lambda_i^n\sigma_j^n\alpha_{ij}\boldsymbol{v}_i\boldsymbol{u}_j^T\|} = \frac{\lambda_1^n\sigma_1^n}{|\lambda_1^n\sigma_1^n|}\frac{\sum_{i,j}\mu_{ij}^n\alpha_{ij}\boldsymbol{v}_i\boldsymbol{u}_j^T}{\|\sum_{i,j}\mu_{ij}^n\alpha_{ij}\boldsymbol{v}_i\boldsymbol{u}_j^T\|}.$$

Since $\lim_{n\to\infty}\mu_{ij}^n = \delta_{i1}\delta_{j1}$, we have that $\lim_{n\to\infty}\phi^{(n+1)} = \boldsymbol{v}_1\boldsymbol{u}_1^T$. Hence

$$\lim_{n\to\infty}\|\boldsymbol{A}^{-1}\boldsymbol{C}\boldsymbol{\phi}^n\boldsymbol{D}\boldsymbol{B}^{-1}\| = \lambda_1\sigma_1.$$

Now we come to the dynamical low-rank algorithm. Here we need to assume that the directions $\boldsymbol{v}_1$ and $\boldsymbol{u}_1$ lie in the initial basis matrices. Let us take a closer look at what this means for the spatial basis. For this, we collect the i-th column of $\boldsymbol{X}$ in the vector $\boldsymbol{X}_i$. We say that $\boldsymbol{v}_1$ is contained in $\boldsymbol{X}$ if for any $i \in \{1,\cdots,r\}$ we have a representation $\boldsymbol{X}_i = \sum_{j=1}^{N_x} T_{ij}\boldsymbol{v}_j$ with $\boldsymbol{T}_i = (T_{ij})_{j=1}^{N_x} \in \mathbb{R}^{N_x}$ and $T_{i1} \neq 0$. Note that $\mathbf{T}$ defines the basis transform between $\boldsymbol{V}$ and the basis matrix retrieved by our dynamical low-rank iteration $\boldsymbol{X}$. Hence, for $\boldsymbol{v}_1$ being contained in $\boldsymbol{X}$ we do not require $\boldsymbol{v}_i$ to be spanned by the r columns of $\boldsymbol{X}$, we only require that $\boldsymbol{v}_i$ lies in the representation of any column. In this setting, the convergence speed of the DLRA power iteration is given by

Theorem 3.1. *The inverse DLRA power iteration scheme as proposed in Section 2 converges for a problem of the form* (19) *to the eigenvector corresponding to the maximal eigenvalue* $\lambda_1\sigma_1$*. Moreover, we have*

$$|\|\boldsymbol{\phi}^n\| - \lambda_1\sigma_1| \leq C_1\left(\frac{\lambda_2}{\lambda_1}\right)^n + C_2\left(\frac{\sigma_2}{\sigma_1}\right)^n.$$

Proof. See Appendix A. □

4. Numerical Results

In the following, we investigate the proposed algorithm for different material and geometric settings. Results will be compared against the full code framework [37]. Note that the memory requirements of this code framework do not allow the computation of a finely resolved solution, which is why we show that the DLRA approach converges to the same solution as the full problem for sufficiently large rank. The dynamical low-rank code that has been used in this work can be found in [21]. Note that the material data used in this work cannot be made publicly available. To compare against a finely resolved reference solution, a DLRA solution with high rank will be used.

4.1. *Stainless-steel reflected uranium sphere*

The first problem we consider is the IEU-MET-FAST-005 criticality benchmark from the OECD/NEA suite [11]. This problem has a sphere of 36% enriched uranium surrounded by a neutron reflector comprised of stainless steel. The problem has an overall radius of 21.486 cm and the uranium sphere has a radius of 13.213 cm. The stainless steel is divided into shells with two different densities: one with radius 1.758 cm and the other in the remainder of the total size.

The dynamical low-rank approximation of this setting is derived according to Section 2. A spatial discretization with 400 spatial cells is chosen. The energy domain is represented by 87 energy groups. Figure 1 shows

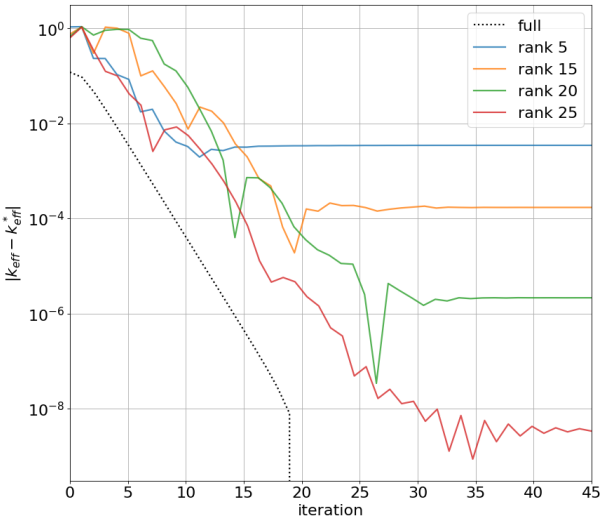
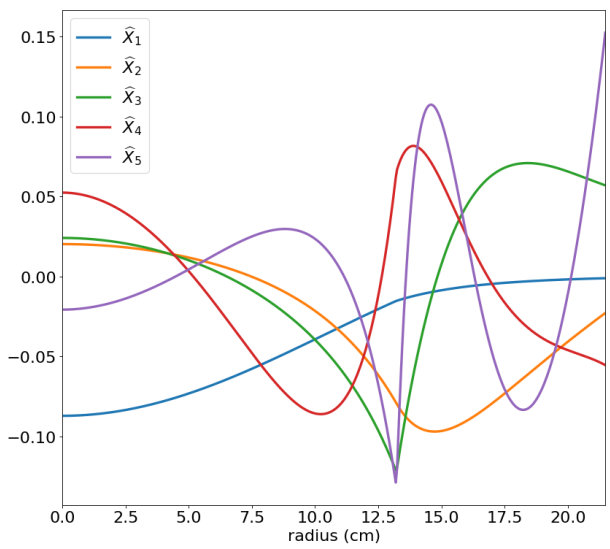


Figure 1: Convergence of the effective eigenvalue for the uranium sphere testcase. As reference effective eigenvalue k^*_{eff} for the 87 group problem, the converged solution of the full inverse power iteration is used. The number of spatial cells is $N_x = 400$.

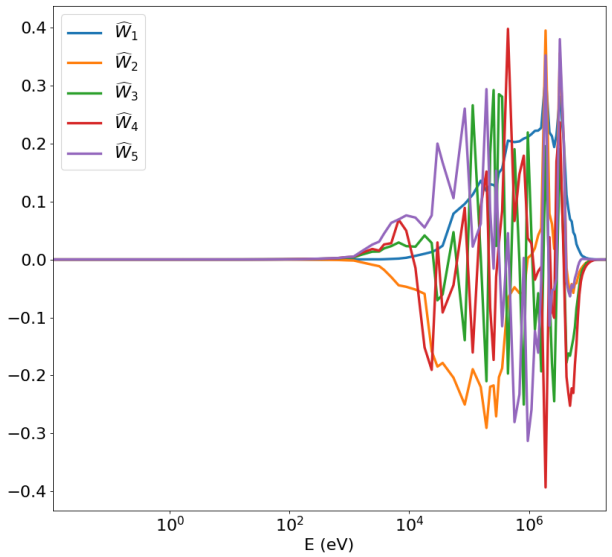
the convergence of the effective eigenvalue for the full problem as well as the DLRA approximation. It is observed that the solutions for rank 20 and 25 show satisfactory convergence properties. Lower ranks result in unsatisfactory approximations of the effective eigenvalue. Furthermore, it should be noted that the DLRA method requires an increased number of iterations to reach a converged state.

From the resulting DLRA factorization, we can investigate the dynamics of the system. For this, we compute an SVD decomposition of the coefficient matrix $\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ and plot the vectors $\widehat{\boldsymbol{X}}_i := \boldsymbol{X}\boldsymbol{U}_i \in \mathbb{R}^{N_x}$ as well as $\widehat{\boldsymbol{W}}_i := \boldsymbol{W}\boldsymbol{V}_i \in \mathbb{R}^G$, where for sake of representation we only look at the first five basis functions. Basis functions and the corresponding eigenvalues of the diagonal matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{r\times r}$ are shown in Figure 2. The basis functions encode both the spatial geometry as well as physical effects. First, the spatial basis captures the change of the background material, especially the transition from uranium to stainless steel. Second, the energy basis encodes the appearance of mostly high-energy particles. The corresponding eigenvalues decay rather slowly as their index increases, which is an indicator for the effectiveness of the method only when the chosen rank is sufficiently high.

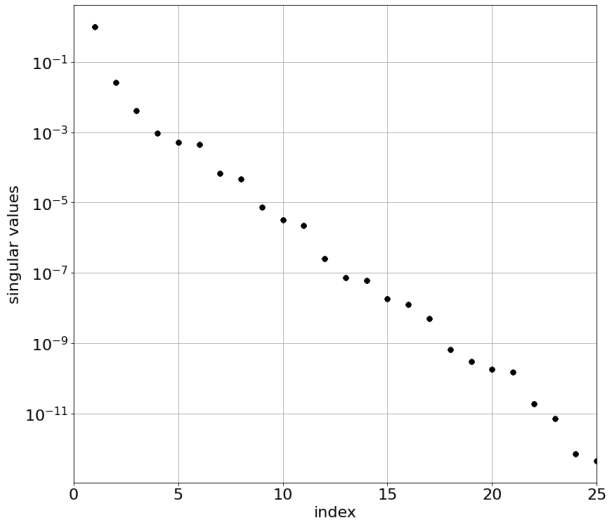
In Figure 3 we look further at the properties of the solution. The spatial variation in the solution integrated over different energy ranges is shown in Figure 3a. In this figure the fast energy range is all groups above 0.5 MeV, epithermal is above 5 eV up to 0.5 MeV, and everything at 5 eV and below is the thermal range. For this problem we observe that the problem is dominated by fast neutrons, as is to be expected because the stainless steel reflector does not moderate neutrons particularly well. To look at the shape of the solution in energy, Figure 3b plots ϕ as a function of energy at different spatial points: one near the outer radius of the sphere and one in the interior. Because ϕ_g in our solution is the integral over the group energy range, in this and subsequent plots we display the average value $\phi(r, E) = \phi_g(r)/\Delta E_g$ for E in group g where ΔE_g is the width of group g. From Figure 3b we can see that there is a significant shift in the energy spectrum at different spatial points in the system.



(a) Basis function in radius

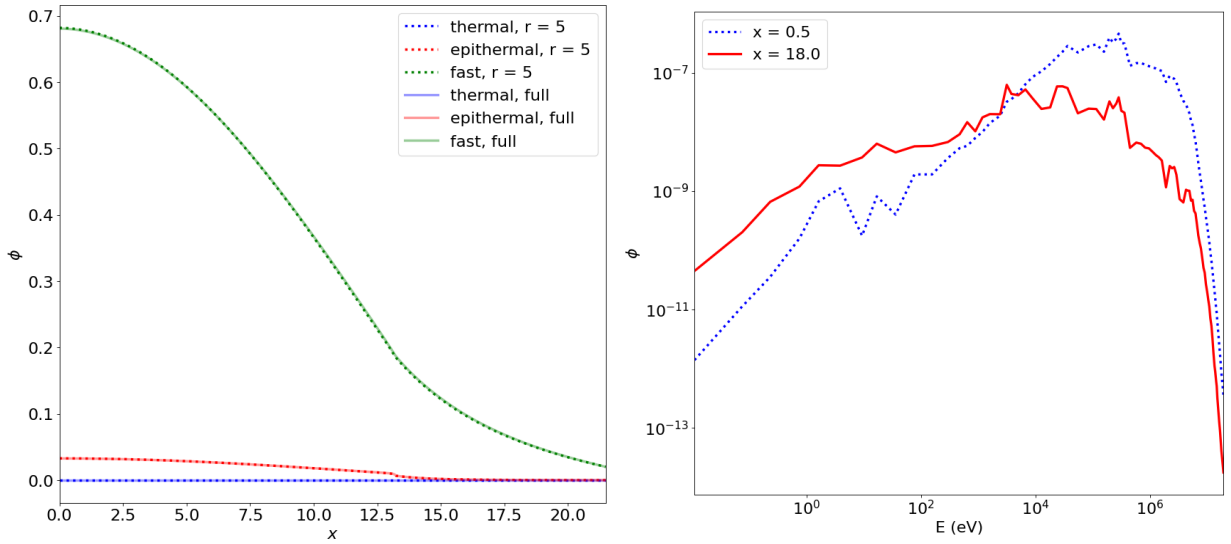


(b) Basis functions in energy



(c) Singular values of the solution matrix

Figure 2: First five DLRA basis functions in (a) radius and (b) energy as well as (c) singular values for the uranium sphere problem $N_x = 400$, $G = 87$ and rank $r = 25$.



(a) Thermal, epithermal and fast neutrons

(b) ϕ at fixed spatial positions.

Figure 3: Left: Thermal ($E \in [0, 5]$), epithermal ($E \in (5, 0.5 \cdot 10^6]$) and fast ($E \in (0.5 \cdot 10^6, \infty)$) neutrons with a rank $r = 5$ approximation and $N_x = 400$, $G = 87$ as well as the full solution. Right: ϕ at fixed spatial positions computed with a rank 25 approximation.

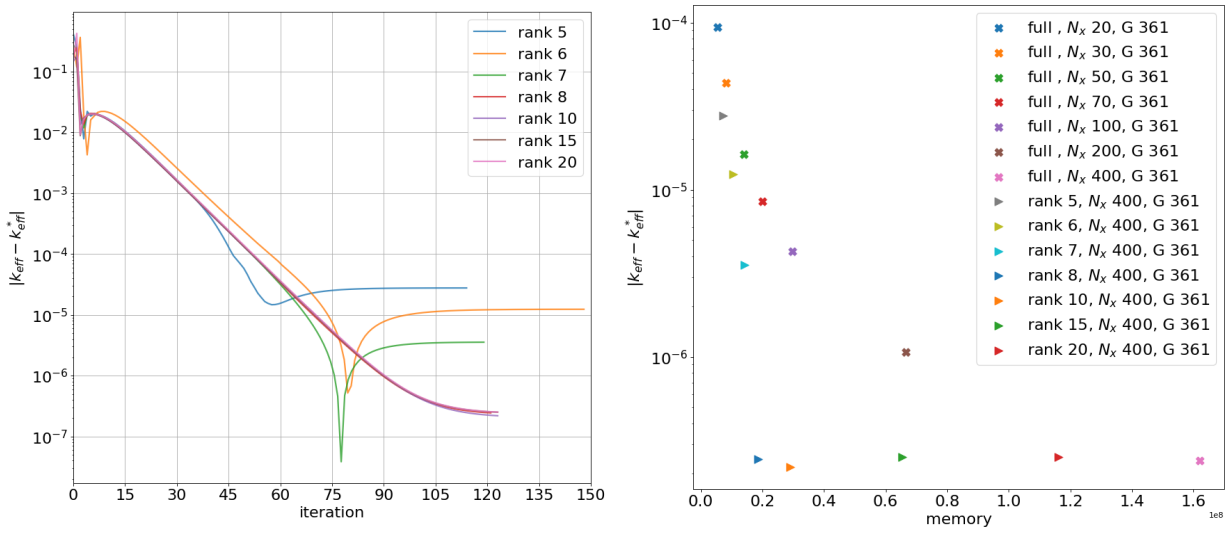
4.2. Spherical, Thermal Reactor

In this problem we look at the solution for a problem of a homogenized mixture of water fuel using the SHEM 361-group energy group structure [15]. The problem consists of this homogenized material in a sphere of radius 79.06925 cm. We expect the solution to have more neutrons in the thermal energy range than the previous problem. Furthermore, this problem has a large number of energy groups. From Figure 4a we see that the eigenvalue k_{eff} converges within 1 pcm (1 percent-mille $= 10^{-5}$) with a rank of 7 when compared to a full reference calculation with $N_x = 1000$ spatial cells. The reduced memory requirement of implementations is depicted in Figure 4b, where we observe that the memory of the full method grows significantly faster then the DLRA method. Note that our implementation of the full method uses Gauss-Seidel iterations, which is not employed in the DLRA implementation. Hence, the depicted memory for dynamical low-rank can further be reduced.

We also note that the number of iterations required is much larger for this problem. This indicates a large dominance ratio, i.e., the ratio of the fundamental eigenvalue to the first harmonic., as would be expected in a problem with a large spatial extent.

The spatial and energy basis functions are plotted in Figure 5. The spatial basis indicates that the leading mode peaks at the center of the problem and decays toward the boundary. The higher spatial basis functions account for the fact that different energy groups will decay at different rates as the edge of the sphere is approached. In this problem the energy basis is especially interesting because the fine group structure is able to give details on many of the resonances in the energy spectrum. Additionally, we observe in Figure 5c that the singular values of the system decay more rapidly in the problem, reaching smaller than 10^{-13} by rank 25.

The spatial variation in the solution and the energy spectrum are plotted in Figure 6. In this figure we observe that for this single material problem, the energy spectrum shape is approximately constant with a magnitude that shifts upward as the center of the sphere is approached. We also observe that DLRA is able to capture the energy self-shielding in the epithermal range as evidences by the characteristic dips in the energy spectrum.



(a) (b)

Figure 4: (a) Convergence of the effective eigenvalue for the light water reactor testcase. (b) Theoretical memory requirement of the full algorithm ($N_x^2 \cdot G + 2 \cdot G^2 \cdot N_x$) vs. dynamical low-rank approximation ($r^2 \cdot N_x^2 + r^2 \cdot G^2$) and corresponding error. Note that we assume that DLRA is not using Gauss-Seidel, which would result in a further significant compression. Since the DLRA code does not make use of this technique, we provide the less memory efficient memory estimate. As reference effective eigenvalue k^*_{eff}, the converged solution of the $N_x = 1000$ cells full method is used. The number of spatial cells for DLRA is $N_x = 400$.

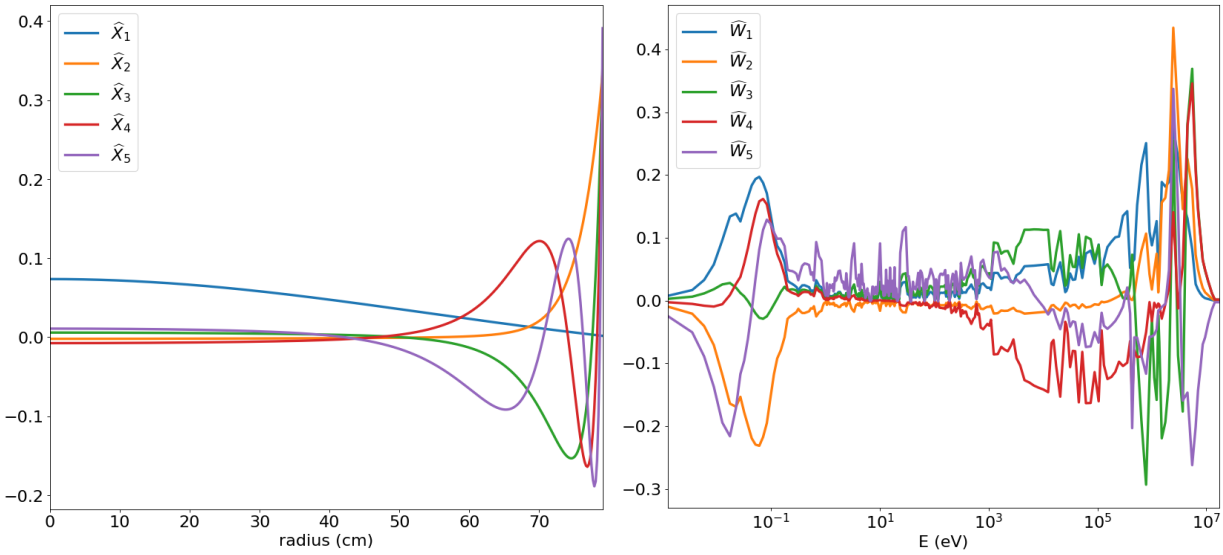
5. Conclusion

In this work, we presented a dynamical low-rank iteration to compute effective eigenvalues in criticality problems. The method treats the iteration index as a pseudo-time and thereby allows deriving update equations for the factorized scalar flux with the DLRA method. Consequently, the memory requirements are decreased significantly, permitting the use of fine discretizations. Our numerical experiments show that the method yields satisfactory approximations of effective eigenvalues for sufficiently high ranks. Physically relevant characteristics are captured by chosen basis functions, which encode resonance regions and relevant energy ranges.

In future work, we aim at investigating applying our techniques alongside different acceleration strategies for power iteration, e.g., coarse-mesh finite difference or Anderson acceleration. Moreover, we aim at including transport terms in the problem formulation. In this case, the solution becomes a tensor and we need employ tensor integrators as presented in [3] or [24].

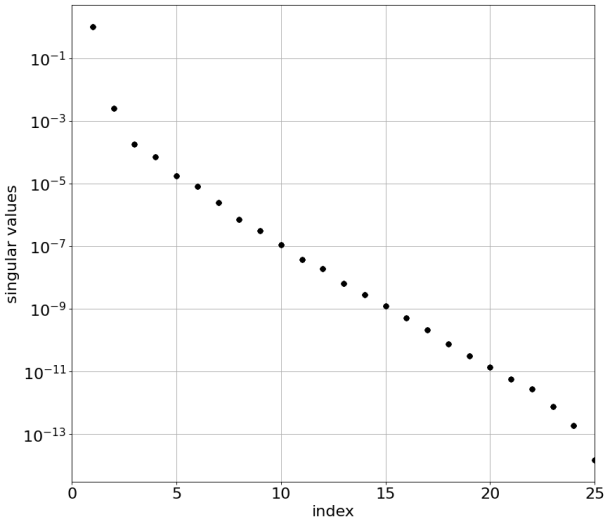
Acknowledgments

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) — Project-ID 258734477 — SFB 1173. This work was also partially funded by the Center for Exascale Monte-Carlo Neutron Transport (CEMeNT) a PSAAP-III project funded by the Department of Energy, grant number DE-NA003967.



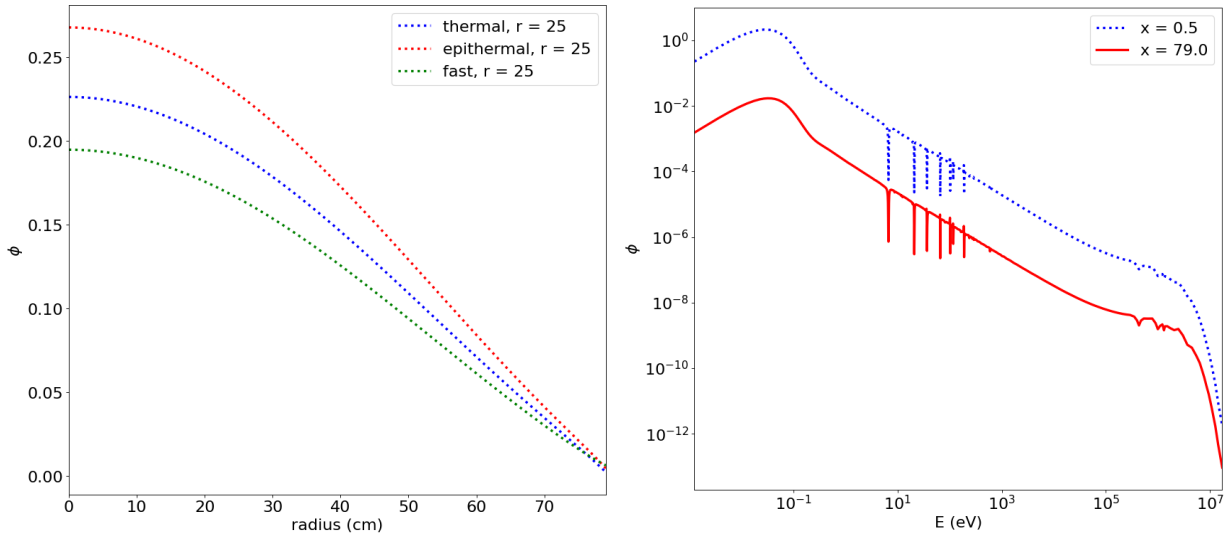
(a) Basis function in radius

(b) Basis functions in energy



(c) Singular values of the solution matrix

Figure 5: First five DLRA basis functions in (a) radius and (b) energy as well as (c) singular values for the light water reactor problem $N_x = 400$, $G = 361$ and rank $r = 25$.



(a) $N_x = 400$, $G = 361$

(b) ϕ at fixed spatial positions

Figure 6: Left: Thermal, epithermal and fast neutrons with a rank $r = 25$ approximation. Right: ϕ at fixed spatial positions.

Appendix A. Convergence in a simplified setting

We wish to prove Theorem 3.1. The idea is to first show that the basis matrices $\mathbf{X}$ and $\mathbf{W}$ can represent the correct basis of the eigenfunction belonging to the largest eigenvalue in Lemma Appendix A.1. Once this result is established, we need to ensure that the coefficient matrix $\mathbf{S}$ will only pick these eigenfunctions. For this, we reformulate the S-step in Lemma Appendix A.2 which then yields the main result in Theorem Appendix A.3. Throughout the proof, we assume the following

Remark 1. *We assume that the eigenvectors $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ have unit norm. Furthermore, it is assumed that $\boldsymbol{A}^{-1}\boldsymbol{C}$ as well as $\boldsymbol{D}\boldsymbol{B}^{-1}$ exist and have full rank.*

Moreover, we employ the following notation:

Remark 2. *The Frobenius norm is denoted by $\|\cdot\|$, whereas the spectral norm is denoted by $\|\cdot\|_2$. Recall that for matrices $\boldsymbol{Y}$ and $\boldsymbol{Z}$ we have $\|\boldsymbol{Y}\boldsymbol{Z}\| \leq \|\boldsymbol{Y}\|_2 \cdot \|\boldsymbol{Z}\| \leq \|\boldsymbol{Y}\| \cdot \|\boldsymbol{Z}\|$.*

Let us start with with a look at the K-step

Lemma Appendix A.1. *Assume that we have a problem of the form (19) and there exist N_x linear independent eigenvectors $\boldsymbol{v}_j$ of $\boldsymbol{A}^{-1}\boldsymbol{C}$. If the direction $\boldsymbol{v}_1$ of the maximal eigenvalue λ_1 is contained in any column of $\boldsymbol{X}^0$, then there exists an $i \leq r$ such that*

$$\|\boldsymbol{X}_i^n - \boldsymbol{v}_1\| \leq C \cdot |\lambda_2/\lambda_1|^n,$$

i.e., the eigenvector $\boldsymbol{v}_1$ lies in the column range of $\boldsymbol{X}^n$ for $n \to \infty$. Furthermore, for $j \neq i$, we have

$$|\boldsymbol{v}_1^T \boldsymbol{X}_j^n| \leq C \cdot |\lambda_2/\lambda_1|^n, \qquad |\boldsymbol{e}_1^T \boldsymbol{T}_j^n| \leq \widetilde{C} \cdot |\lambda_2/\lambda_1|^n. \tag{A.1}$$

Proof. Our derivation of the K-step yields

$$\boldsymbol{A}\boldsymbol{K}^{n+1}\boldsymbol{W}^{n,T}\boldsymbol{B}\boldsymbol{W}^n = \boldsymbol{C}\boldsymbol{K}^n\boldsymbol{W}^{n,T}\boldsymbol{D}\boldsymbol{W}^n,$$

where superscripts again denote the iteration step. Let us define $\widetilde{\boldsymbol{B}}^n := \boldsymbol{W}^{n,T}\boldsymbol{B}\boldsymbol{W}^n$ and $\widetilde{\boldsymbol{D}}^n := \boldsymbol{W}^{n,T}\boldsymbol{D}\boldsymbol{W}^n$. Furthermore, we write $\boldsymbol{X}^n = \boldsymbol{V}\boldsymbol{T}^n$, where $\boldsymbol{T}^n \in \mathbb{R}^{N_x \times r}$. Then with $\boldsymbol{K}^{n+1} = \boldsymbol{X}^{n+1}\widetilde{\boldsymbol{S}}$ we have

$$\boldsymbol{X}^{n+1}\widetilde{\boldsymbol{S}} = \boldsymbol{A}^{-1}\boldsymbol{C}\boldsymbol{K}^n\widetilde{\boldsymbol{D}}^n(\widetilde{\boldsymbol{B}}^n)^{-1} = \widehat{\boldsymbol{C}}\boldsymbol{X}^n\boldsymbol{S}^n\widetilde{\boldsymbol{D}}^n(\widetilde{\boldsymbol{B}}^n)^{-1} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{T}^n\boldsymbol{S}^n\widetilde{\boldsymbol{D}}^n(\widetilde{\boldsymbol{B}}^n)^{-1}.$$

Let us define the diagonal matrix $\boldsymbol{\Lambda}_c := \mathrm{diag}(1, \lambda_2/\lambda_1, \cdots, \lambda_{N_x}/\lambda_1)$, which gives

$$\boldsymbol{X}^{n+1}\widetilde{\boldsymbol{S}} = \boldsymbol{V}\boldsymbol{\Lambda}_c\boldsymbol{T}^n\boldsymbol{S}^n\widetilde{\boldsymbol{D}}^n(\widetilde{\boldsymbol{B}}^n)^{-1}\cdot\lambda_1.$$

From the S-step as well as Remark 1, we know that $\boldsymbol{S}^n\widetilde{\boldsymbol{D}}^n(\widetilde{\boldsymbol{B}}^n)^{-1}$ is of rank r for all n. Hence, the column range of $\boldsymbol{X}^{n+1}$, which we denote by $\mathrm{range}(\boldsymbol{X}^{n+1})$ is

$$\mathrm{range}(\boldsymbol{X}^{n+1}) = \mathrm{range}(\boldsymbol{V}\boldsymbol{\Lambda}_c\boldsymbol{T}^n) = \mathrm{range}(\boldsymbol{V}\boldsymbol{\Lambda}_c\boldsymbol{V}^{-1}\boldsymbol{V}\boldsymbol{T}^n) = \mathrm{range}(\boldsymbol{V}\boldsymbol{\Lambda}_c\boldsymbol{V}^{-1}\boldsymbol{X}^n) = \mathrm{range}(\boldsymbol{V}\boldsymbol{\Lambda}_c^{n+1}\boldsymbol{T}^0).$$

Therefore, there exist r coefficient vectors $\boldsymbol{\alpha}_i \in \mathbb{R}^{N_x}$ such that with the normalization factor $\gamma_{i,n} := \|\boldsymbol{V}\boldsymbol{\Lambda}_c^n\boldsymbol{\alpha}_i\|$ we have $\boldsymbol{X}_i^{n+1} = \frac{1}{\gamma_{i,n}}\boldsymbol{V}\boldsymbol{\Lambda}_c^{n+1}\boldsymbol{\alpha}_i$. Note that

$$\left|\gamma_{i,n} - |\alpha_{1i}|\right| \leq \left(\sum_{\ell=2}^{N_x}\left(\frac{\lambda_\ell}{\lambda_1}\right)^{2n}\alpha_{\ell i}^2\right)^{1/2}. \tag{A.2}$$

Hence, when assuming $\alpha_{1i} > 0$, we have

$$\begin{aligned}\|\boldsymbol{X}_i^n - \boldsymbol{v}_1\|^2 =& \frac{1}{\gamma_{i,n}^2}\|\boldsymbol{V}(\boldsymbol{\Lambda}_c^n\boldsymbol{\alpha}_i - \boldsymbol{e}_1\gamma_{i,n})\|^2 \leq \frac{\|\boldsymbol{V}\|_2^2}{\gamma_{i,n}^2}\|\boldsymbol{\Lambda}_c^n\boldsymbol{\alpha}_i - \boldsymbol{e}_1\gamma_{i,n}\|^2 \\ =& \frac{\|\boldsymbol{V}\|_2^2}{\gamma_{i,n}^2}\left((\alpha_{1i}-\gamma_{i,n})^2 + \sum_{\ell=2}^{N_x}\left(\frac{\lambda_\ell}{\lambda_1}\right)^{2n}\alpha_{\ell i}^2\right) \leq C|\lambda_2/\lambda_1|^n.\end{aligned}$$

For $\alpha_{1i} \leq 0$ we have that $\|\boldsymbol{X}_i^n - (-\boldsymbol{v}_1)\| \leq C|\lambda_2/\lambda_1|^n$. Hence, $\boldsymbol{X}_i^n$ converges to $\pm\boldsymbol{v}_1$ and the eigenvector $\boldsymbol{v}_1$ lies in the range of $\boldsymbol{X}^n$ for $n \to \infty$. For $j \neq i$, we have

$$|\boldsymbol{v}_1^T\boldsymbol{X}_j^n| = |(\boldsymbol{v}_1 - \boldsymbol{X}_i^n)^T\boldsymbol{X}_j^n| \leq \|\boldsymbol{v}_1 - \boldsymbol{X}_i^n\|\cdot\|\boldsymbol{X}_j^n\| \leq C\cdot|\lambda_2/\lambda_1|^n.$$

Furthermore, since we assume that eigenvectors are linearly independent, we have

$$|\boldsymbol{e}_1^T\boldsymbol{T}_j^n| = |(\boldsymbol{e}_1 - \boldsymbol{T}_i^n)^T\boldsymbol{T}_j^n| \leq \|\boldsymbol{V}^{-1}\boldsymbol{V}(\boldsymbol{e}_1 - \boldsymbol{T}_i^n)\|\cdot\|\boldsymbol{T}_j^n\| \leq \frac{C}{\|\boldsymbol{V}^{-1}\|_2}\cdot|\lambda_2/\lambda_1|^n.$$

□

Remark 3. *The same holds true for $\boldsymbol{W}^n$: If $\boldsymbol{W}^0$ contains $\boldsymbol{u}_1$, this direction lies in the range of $\boldsymbol{W}^n$ for $n \to \infty$. The proof is straightforward and simply applies the above derivation to the L-step.*

Now, we know that in the limit, the eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{u}_1$ form one of the columns of $\boldsymbol{X}$ and $\boldsymbol{W}$, respectively. If we have $\boldsymbol{X}_i = \boldsymbol{v}_1$ and $\boldsymbol{W}_\ell = \boldsymbol{u}_1$ it remains to show that $S_{jk}^n \to \delta_{ji}\delta_{k\ell}$, such that $\boldsymbol{\phi}^n \to \lambda_1\boldsymbol{v}_1\boldsymbol{u}_1^T\sigma_1$ as $n \to \infty$. For this, we need to take a closer look at the S-step, which reads

$$\boldsymbol{X}^{n+1,T}\boldsymbol{A}\boldsymbol{X}^{n+1}\widetilde{\boldsymbol{S}}^{n+1}\boldsymbol{W}^{n+1,T}\boldsymbol{B}\boldsymbol{W}^{n+1} = \boldsymbol{X}^{n+1,T}\boldsymbol{C}\boldsymbol{X}^{n+1}\boldsymbol{M}\boldsymbol{S}^n\boldsymbol{N}^T\boldsymbol{W}^{n+1,T}\boldsymbol{D}\boldsymbol{W}^{n+1}$$

Therefore,

$$\widetilde{\boldsymbol{S}}^{n+1} = (\widetilde{\boldsymbol{A}}^{n+1})^{-1}\widetilde{\boldsymbol{C}}^{n+1}\boldsymbol{M}\boldsymbol{S}^n\boldsymbol{N}^T\widetilde{\boldsymbol{D}}^{n+1}(\widetilde{\boldsymbol{B}}^{n+1})^{-1}.$$

Let us represent our basis in space and energy as $\boldsymbol{X}^n = \boldsymbol{V}\boldsymbol{T}_x^n$ and $\boldsymbol{W}^n = \boldsymbol{U}\boldsymbol{T}_e^n$. Then, the S-step can be

rewritten in the following way:

Lemma Appendix A.2. *With* $\boldsymbol{\Delta}^{(x)} := \boldsymbol{I} - \boldsymbol{T}_x\boldsymbol{T}_x^T \in \mathbb{R}^{N_x\times N_x}$ *and* $\boldsymbol{\Delta}^{(e)} := \boldsymbol{I} - \boldsymbol{T}_e\boldsymbol{T}_e^T \in \mathbb{R}^{G\times G}$*, let us define*

$$\begin{aligned}\boldsymbol{\mathcal{E}}_x^{(n+1)} :=& (\widetilde{\boldsymbol{A}}^{n+1})^{-1}\boldsymbol{X}^{n+1,T}\boldsymbol{A}\boldsymbol{V}\boldsymbol{\Delta}^{(x)}\boldsymbol{\Lambda}_c\boldsymbol{T}_x^{n+1},\\ \boldsymbol{\mathcal{E}}_e^{(n+1)} :=& \boldsymbol{T}_e^{n+1,T}\boldsymbol{\Sigma}_c\boldsymbol{\Delta}^{(e)}\boldsymbol{U}^T\boldsymbol{B}\boldsymbol{W}^{n+1}(\widetilde{\boldsymbol{B}}^{n+1})^{-1}.\end{aligned}$$

Then, the S-step takes the form

$$\widetilde{\boldsymbol{S}}^{n+1} = \lambda_1\sigma_1\left(\boldsymbol{T}_x^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}_x^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\boldsymbol{N}^T\left(\boldsymbol{\mathcal{E}}_e^{(n+1)} + \boldsymbol{T}_e^{n+1,T}\boldsymbol{\Sigma}_c\boldsymbol{T}_e^{n+1}\right). \tag{A.3}$$

Proof. Let us show the derivation for the spatial part only and drop the x index in the following. We know that $\boldsymbol{A}^{-1}\boldsymbol{C}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{\Lambda}$, hence $\boldsymbol{C}\boldsymbol{V} = \boldsymbol{A}\boldsymbol{V}\boldsymbol{\Lambda}$. Then, the matrix $\widetilde{\boldsymbol{C}}$ becomes

$$\widetilde{\boldsymbol{C}}^{n+1} = \boldsymbol{X}^{n+1,T}\boldsymbol{C}\boldsymbol{X}^{n+1} = \boldsymbol{T}^{n+1,T}\boldsymbol{V}^T\boldsymbol{C}\boldsymbol{V}\boldsymbol{T}^{n+1} = \boldsymbol{T}^{n+1,T}\boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{T}^{n+1}.$$

Let $\boldsymbol{\Delta} := \boldsymbol{I} - \boldsymbol{T}^{n+1}\boldsymbol{T}^{n+1,T} \in \mathbb{R}^{N_x\times N_x}$. Then, with $\boldsymbol{I} = \boldsymbol{T}^{n+1}\boldsymbol{T}^{n+1,T} + \boldsymbol{\Delta}$ we have

$$\begin{aligned}\widetilde{\boldsymbol{C}}^{n+1} =& \boldsymbol{T}^{n+1,T}\boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}(\boldsymbol{T}^{n+1}\boldsymbol{T}^{n+1,T} + \boldsymbol{\Delta})\boldsymbol{\Lambda}\boldsymbol{T}^{n+1}\\ =& \widetilde{\boldsymbol{A}}^{n+1}\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}\boldsymbol{T}^{n+1} + \boldsymbol{T}^{n+1,T}\boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{\Lambda}\boldsymbol{T}^{n+1}.\end{aligned}$$

Therefore,

$$\begin{aligned}(\widetilde{\boldsymbol{A}}^{n+1})^{-1}\widetilde{\boldsymbol{C}}^{n+1} &= \boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}\boldsymbol{T}^{n+1} + (\widetilde{\boldsymbol{A}}^{n+1})^{-1}\boldsymbol{X}^{n+1,T}\boldsymbol{A}\boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{\Lambda}\boldsymbol{T}^{n+1}\\ &= \lambda_1\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + (\widetilde{\boldsymbol{A}}^{n+1})^{-1}\boldsymbol{X}^{n+1,T}\boldsymbol{A}\boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1}\right)\\ &=: \lambda_1\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right).\end{aligned} \tag{A.4}$$

The energy parts can be derived analogously, which yields the S-step (A.3). □

Now we have all building block to show convergence: Recall Theorem 3.1.

Theorem Appendix A.3. *The inverse DLRA power iteration scheme as proposed in Section 2 converges for a problem of the form* (19) *to the eigenvector corresponding to the maximal eigenvalue* $\lambda_1\sigma_1$*. Moreover, we have*

$$|\|\boldsymbol{\phi}^n\| - \lambda_1\sigma_1| \le C_1\left(\frac{\lambda_2}{\lambda_1}\right)^n + C_2\left(\frac{\sigma_2}{\sigma_1}\right)^n.$$

Proof. Let us investigate the spatial part of the S-step (A.3), namely (A.4) and again conclude the terms for the energy part. We start by defining $\boldsymbol{P}_i := \boldsymbol{e}_i\boldsymbol{e}_i^T$ and $\boldsymbol{P}_i^\perp := \boldsymbol{I} - \boldsymbol{e}_i\boldsymbol{e}_i^T$. The idea of this proof is to show that $\boldsymbol{P}_i^\perp\boldsymbol{S}$ and $\boldsymbol{S}(\boldsymbol{P}_\ell^\perp)^T \to 0$ as $n\to\infty$. Let us start by noting that

$$\boldsymbol{P}_i^\perp\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1}\boldsymbol{M}\boldsymbol{S}^n = \boldsymbol{P}_i^\perp\boldsymbol{T}^{n+1,T}(\boldsymbol{P}_1 + \boldsymbol{P}_1^\perp)\boldsymbol{\Lambda}_c(\boldsymbol{P}_1 + \boldsymbol{P}_1^\perp)\boldsymbol{T}^{n+1}(\boldsymbol{P}_i + \boldsymbol{P}_i^\perp)\boldsymbol{M}\boldsymbol{S}^n. \tag{A.5}$$

With (A.1), we have $\|\boldsymbol{P}_i^\perp\boldsymbol{T}^{n+1,T}\boldsymbol{P}_1\| \le C(\lambda_2/\lambda_1)^{n+1}$ and $\|\boldsymbol{P}_1^\perp\boldsymbol{T}^{n+1}\boldsymbol{P}_i\| \le C(\lambda_2/\lambda_1)^{n+1}$, hence

$$\begin{aligned}\|\boldsymbol{P}_i^\perp\boldsymbol{T}^{n+1,T}\boldsymbol{P}_1\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1}\boldsymbol{M}\boldsymbol{S}^n\| \le \widetilde{C}(\lambda_2/\lambda_1)^{n+1},\\ \|\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{P}_1^\perp\boldsymbol{T}^{n+1}\boldsymbol{P}_i\boldsymbol{M}\boldsymbol{S}^n\| \le \widetilde{C}(\lambda_2/\lambda_1)^{n+1}.\end{aligned}$$

Together with $\boldsymbol{P}_1^\perp\boldsymbol{\Lambda}_c\boldsymbol{P}_1 = 0$, (A.5) becomes

$$\|\boldsymbol{P}_i^\perp\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1}\boldsymbol{M}\boldsymbol{S}^n\| \le \|\boldsymbol{P}_i^\perp\boldsymbol{T}^{n+1,T}\boldsymbol{P}_1^\perp\boldsymbol{\Lambda}_c\boldsymbol{P}_1^\perp\boldsymbol{T}^{n+1}\boldsymbol{P}_i^\perp\boldsymbol{M}\boldsymbol{S}^n\| + C(\lambda_2/\lambda_1)^{n+1}.$$

Let us observe that

$$\begin{aligned}\|\boldsymbol{P}_i^{\perp}\boldsymbol{M}\boldsymbol{P}_i\| =&\|\boldsymbol{P}_i^{\perp}\boldsymbol{T}^{n+1,T}\boldsymbol{V}^T\boldsymbol{V}\boldsymbol{T}^n\boldsymbol{P}_i\|\\ =&\|\boldsymbol{P}_i^{\perp}\boldsymbol{T}^{n+1,T}(\boldsymbol{V}(\boldsymbol{P}_1+\boldsymbol{P}_1^{\perp}))^T\boldsymbol{V}(\boldsymbol{P}_1+\boldsymbol{P}_1^{\perp})\boldsymbol{T}^n\boldsymbol{P}_i\|\\ \leq&\|\boldsymbol{P}_i^{\perp}\boldsymbol{T}^{n+1,T}(\boldsymbol{V}\boldsymbol{P}_1^{\perp})^T\boldsymbol{V}\boldsymbol{P}_1\boldsymbol{T}^n\boldsymbol{P}_i\| + C(\lambda_2/\lambda_1)^{n+1} = C(\lambda_2/\lambda_1)^{n+1},\end{aligned}$$

where in the last step we used $(\boldsymbol{V}\boldsymbol{P}_1^{\perp})^T\boldsymbol{V}\boldsymbol{P}_1 = \boldsymbol{0}$. Hence, with $\Lambda_0 := \mathrm{diag}(0, \lambda_2/\lambda_1, \cdots, \lambda_{N_x}/\lambda_1)$, we have

$$\begin{aligned}\|\boldsymbol{P}_i^{\perp}\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1}\boldsymbol{M}\boldsymbol{S}^n\| \leq&\|\boldsymbol{P}_i^{\perp}\boldsymbol{T}^{n+1,T}\boldsymbol{P}_1^{\perp}\boldsymbol{\Lambda}_c\boldsymbol{P}_1^{\perp}\boldsymbol{T}^{n+1}\boldsymbol{P}_i^{\perp}\boldsymbol{M}\boldsymbol{S}^n\| + C(\lambda_2/\lambda_1)^{n+1}\\ \leq&\|\boldsymbol{P}_i^{\perp}\boldsymbol{T}^{n+1,T}\boldsymbol{P}_1^{\perp}\boldsymbol{\Lambda}_c\boldsymbol{P}_1^{\perp}\boldsymbol{T}^{n+1}\boldsymbol{P}_i^{\perp}\boldsymbol{M}\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n\| + C(\lambda_2/\lambda_1)^{n+1}\\ \leq&\|\boldsymbol{P}_i^{\perp}\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_0\boldsymbol{T}^{n+1}\boldsymbol{M}\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n\| + C(\lambda_2/\lambda_1)^{n+1}.\end{aligned}$$

In the same manner we have with $\boldsymbol{\mathcal{E}}_{x,0}^{(n+1)} := (\widetilde{\boldsymbol{A}}^{n+1})^{-1}\boldsymbol{X}^{n+1,T}\boldsymbol{A}\boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{\Lambda}_0\boldsymbol{T}_x^{n+1}$ that

$$\|\boldsymbol{\mathcal{E}}_x^{(n+1)}\boldsymbol{M}\boldsymbol{S}^n\| \leq \|\boldsymbol{\mathcal{E}}_{x,0}^{(n+1)}\boldsymbol{M}\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n\| + C_2(\lambda_2/\lambda_1)^{n+1}.$$

$$\left\|\boldsymbol{P}_i^{\perp}\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\| \leq \left\|\boldsymbol{P}_i^{\perp}\left(\boldsymbol{T}^{n+1,T}\boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}\boldsymbol{\Lambda}_0\boldsymbol{T}^{n+1}\right)\boldsymbol{M}\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n\right\| + C(\lambda_2/\lambda_1)^{n+1}.$$

All together, this gives the estimate

$$\begin{aligned}&\frac{\lambda_1\left\|\boldsymbol{P}_i^{\perp}\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|}{\lambda_1\left\|\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|}\\ \leq&\frac{\|\boldsymbol{P}_i^{\perp}\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_0\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_{x,0}^{(n+1)}\right)\boldsymbol{M}\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n\|}{\left\|\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|} + C(\lambda_2/\lambda_1)^{n+1}\\ \leq&\widehat{C}\cdot\|\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n\| + C(\lambda_2/\lambda_1)^{n+1}.\end{aligned}$$

Since by the normalization step of our scheme $\|\boldsymbol{S}^n\| = \|\boldsymbol{S}^{n,-1}\| = 1$, we have

$$\begin{aligned}\widehat{C} :=&\frac{\left\|\boldsymbol{P}_i^{\perp}\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_0\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_{x,0}^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\boldsymbol{S}^{n,-1}\right\|}{\left\|\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|} \leq \frac{\|\boldsymbol{P}_i^{\perp}\|_2\left\|\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_0\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_{x,0}^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|\cdot\|\boldsymbol{S}^{n,-1}\|}{\left\|\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|}\\ =&\frac{\left\|\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_0\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_{x,0}^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|}{\left\|\left(\boldsymbol{T}^{n+1,T}\boldsymbol{\Lambda}_c\boldsymbol{T}^{n+1} + \boldsymbol{\mathcal{E}}_x^{(n+1)}\right)\boldsymbol{M}\boldsymbol{S}^n\right\|} \leq \frac{\lambda_2}{\lambda_1}.\end{aligned}$$

Including the term $\boldsymbol{N}^T\widetilde{\boldsymbol{D}}^{n+1}(\widetilde{\boldsymbol{B}}^{n+1})^{-1}$ and utilizing the same arguments as above yields

$$\begin{aligned}\|\boldsymbol{P}_i^{\perp}\boldsymbol{S}^{n+1}(\boldsymbol{P}_\ell^{\perp})^T\| \leq&\frac{\lambda_2\sigma_2}{\lambda_1\sigma_1}\|\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n(\boldsymbol{P}_\ell^{\perp})^T\| + C\left(\frac{\lambda_2\sigma_2}{\lambda_1\sigma_1}\right)^{n+1}\\ \|\boldsymbol{P}_i^{\perp}\boldsymbol{S}^{n+1}\boldsymbol{P}_\ell^T\| \leq&\frac{\lambda_2}{\lambda_1}\|\boldsymbol{P}_i^{\perp}\boldsymbol{S}^n\boldsymbol{P}_\ell^T\| + C\left(\frac{\lambda_2}{\lambda_1}\right)^{n+1}\\ \|\boldsymbol{P}_i\boldsymbol{S}^{n+1}(\boldsymbol{P}_\ell^{\perp})^T\| \leq&\frac{\sigma_2}{\sigma_1}\|\boldsymbol{P}_i\boldsymbol{S}^n(\boldsymbol{P}_\ell^{\perp})^T\| + C\left(\frac{\sigma_2}{\sigma_1}\right)^{n+1}\end{aligned}$$

Recursively, we get

$$\|\boldsymbol{P}_i^\perp \boldsymbol{S}^n (\boldsymbol{P}_\ell^\perp)^T\| \leq c \left(\frac{\lambda_2\sigma_2}{\lambda_1\sigma_1}\right)^n, \quad \|\boldsymbol{P}_i^\perp \boldsymbol{S}^n \boldsymbol{P}_\ell^T\| \leq c \left(\frac{\lambda_2}{\lambda_1}\right)^n, \quad \|\boldsymbol{P}_i \boldsymbol{S}^n (\boldsymbol{P}_\ell^\perp)^T\| \leq c \left(\frac{\sigma_2}{\sigma_1}\right)^n.$$

Since $\|\boldsymbol{S}^n\| = \|(\boldsymbol{P}_i + \boldsymbol{P}_i^\perp)\boldsymbol{S}^n(\boldsymbol{P}_\ell + \boldsymbol{P}_\ell^\perp)^T\| = 1$, we know that

$$\left|\|\boldsymbol{P}_i \boldsymbol{S}^n \boldsymbol{P}_\ell^T\| - 1\right| \leq C_0 \left(\frac{\lambda_2\sigma_2}{\lambda_1\sigma_1}\right)^n + C_1 \left(\frac{\lambda_2}{\lambda_1}\right)^n + C_2 \left(\frac{\sigma_2}{\sigma_1}\right)^n.$$

I.e., $S_{kj}^n \to \delta_{ik}\delta_{\ell j}$ and with Lemma Appendix A.1 and Remark 3 we conclude the theorem. □

References

[1] G. Bell and S. Glasstone. *Nuclear Reactor Theory*. Van Nostrand Reinhold Company, 1970.

[2] G. Ceruti, J. Kusch, and C. Lubich. A rank-adaptive robust integrator for dynamical low-rank approximation. *arXiv preprint arXiv:2104.05247*, 2021.

[3] G. Ceruti and C. Lubich. An unconventional robust integrator for dynamical low-rank approximation. *BIT Numerical Mathematics*, pages 1–22, 2021.

[4] A. Dektor, A. Rodgers, and D. Venturi. Rank-adaptive tensor methods for high-dimensional nonlinear PDEs. *Journal of Scientific Computing*, 88(2):1–27, 2021.

[5] L. Einkemmer, J. Hu, and L. Ying. An efficient dynamical low-rank algorithm for the Boltzmann-BGK equation close to the compressible viscous flow regime. *arXiv preprint arXiv:2101.07104*, 2021.

[6] L. Einkemmer and I. Joseph. A mass, momentum, and energy conservative dynamical low-rank scheme for the Vlasov equation. *arXiv preprint arXiv:2101.12571*, 2021.

[7] L. Einkemmer and C. Lubich. A low-rank projector-splitting integrator for the Vlasov–Poisson equation. *SIAM J. Sci. Comput.*, 40(5):B1330–B1360, 2018.

[8] L. Einkemmer and C. Lubich. A quasi-conservative dynamical low-rank algorithm for the Vlasov equation. *SIAM Journal on Scientific Computing*, 41(5):B1061–B1081, 2019.

[9] L. Einkemmer, A. Ostermann, and C. Piazzola. A low-rank projector-splitting integrator for the Vlasov–Maxwell equations with divergence correction. *Journal of Computational Physics*, 403:109063, 2020.

[10] F. Feppon and P. F. Lermusiaux. Dynamically orthogonal numerical schemes for efficient stochastic advection and Lagrangian transport. *SIAM Rev.*, 60(3):595–625, 2018.

[11] M. V. Gorbatenko, V. P. Gorelov, V. P. Yegorov, V. G. Zagrafov, A. N. Zakharov, V. I. Ilyin, M. I. Kuvshinov, A. A. Malinkin, and V. I. Yuferev. Steel-reflected spherical assembly of ^{235}U(36%). In *International Handbook of Evaluated Criticality Safety Benchmark Experiments*, number NEA/NSC/DOC/(95)03/III. Organisation for Economic Co-operation and Development Nuclear Energy Agency, September 2002.

[12] N. Guglielmi, D. Kressner, and C. Scalone. Computing low-rank rightmost eigenpairs of a class of matrix-valued linear operators. *Advances in Computational Mathematics*, 47(5):1–28, 2021.

[13] W. Guo and J.-M. Qiu. A Low Rank Tensor Representation of Linear Transport and Nonlinear Vlasov Solutions and Their Associated Flow Maps. *arXiv:2106.08834*, 2021.

[14] T. Hazama, G. Chiba, and K. Sugino. Development of a fine and ultra-fine group cell calculation code slarom-uf for fast reactor analyses. *Journal of nuclear science and technology*, 43(8):908–918, 2006.

[15] A. Hébert and A. Santamarina. Refinement of the Santamarina-Hfaiedh energy mesh between 22.5 eV and 11.4 keV. In *International Conference on the Physics of Reactors, Interlaken, Switzerland*, 2008.

[16] O. Koch and C. Lubich. Dynamical low-rank approximation. *SIAM J. Matrix Anal. Appl.*, 29(2):434–454, 2007.

[17] O. Koch and C. Lubich. Dynamical tensor approximation. *SIAM J. Matrix Anal. Appl.*, 31(5):2360–2375, 2010.

[18] J. Kusch, G. Ceruti, L. Einkemmer, and M. Frank. Dynamical low-rank approximation for Burgers' equation with uncertainty. *arXiv preprint arXiv:2105.04358*, 2021.

[19] J. Kusch, L. Einkemmer, and G. Ceruti. On the stability of robust dynamical low-rank approximations for hyperbolic problems. *arXiv preprint arXiv:2107.07282*, 2021.

[20] J. Kusch and P. Stammer. A robust collision source method for rank adaptive dynamical low-rank approximation in radiation therapy. *arXiv preprint arXiv:2111.07160*, 2021.

[21] J. Kusch, B. Whehell, R. McClarren, and M. Frank. *Numerical testcases for "A low-rank power iteration scheme for neutron transport criticality problems"*, 2021.

[22] E. Lewis and W. Miller. *Computational Methods of Neutron Transport*. Wiley, 1984.

[23] C. Lubich and I. V. Oseledets. A projector-splitting integrator for dynamical low-rank approximation. *BIT*, 54(1):171–188, 2014.

[24] C. Lubich, B. Vandereycken, and H. Walach. Time integration of rank-constrained Tucker tensors. *SIAM J. Numer. Anal.*, 56(3):1273–1290, 2018.

[25] I. Lux and L. Koblinger. *Monte Carlo particle transport methods: neutron and photon calculations*. CRC press, 2018.

[26] R. McClarren. *Computational nuclear engineering and radiological science using python*. Academic Press, 2017.
[27] R. G. McClarren. Theoretical aspects of the simplified Pn equations. *Transport Theory and Statistical Physics*, 39(2-4):73–109, 2010.
[28] E. Musharbash and F. Nobile. Dual dynamically orthogonal approximation of incompressible Navier–Stokes equations with random boundary conditions. *J. Comput. Phys.*, 354:135–162, 2018.
[29] E. Musharbash, F. Nobile, and E. Vidličková. Symplectic dynamical low rank approximation of wave equations with random parameters. *BIT Numer. Math.*, 60:1153–1201, 2020.
[30] Z. Peng and R. G. McClarren. A high-order/low-order (holo) algorithm for preserving conservation in time-dependent low-rank transport calculations. *Journal of Computational Physics*, 447:110672, 2021.
[31] Z. Peng, R. G. McClarren, and M. Frank. A low-rank method for two-dimensional time-dependent radiation transport calculations. *J. Comput. Phys.*, 421:109735, 2020.
[32] A. Rodgers, A. Dektor, and D. Venturi. Adaptive integration of nonlinear evolution equations on tensor manifolds. *arXiv preprint arXiv:2008.00155*, 2020.
[33] T. P. Sapsis and P. F. Lermusiaux. Dynamically orthogonal field equations for continuous stochastic dynamical systems. *Physica D*, 238(23-24):2347–2360, 2009.
[34] W. M. Stacey. *Nuclear reactor physics*. John Wiley & Sons, 2018.
[35] H. M. Walach. *Time integration for the dynamical low-rank approximation of matrices and tensors*. PhD thesis, Universität Tübingen, 2019.
[36] Y. Wang, W. Bangerth, and J. Ragusa. Three-dimensional h-adaptivity for the multigroup neutron diffusion equations. *Progress in Nuclear Energy*, 51(3):543–555, 2009.
[37] B. Whewell. *Neutron diffusion framework*, 2021. https://github.com/bwhewe-13/NeutronDiffusion.