Fast, Low-Memory Numerical Methods for Radiative Transfer via hp-Adaptive Mesh Refinement

Shukai Du\textsuperscript{a}, Samuel N. Stechmann\textsuperscript{a, b}

\textsuperscript{a}Department of Mathematics, University of Wisconsin–Madison
\textsuperscript{b}Department of Atmospheric and Oceanic Sciences, University of Wisconsin–Madison

Abstract

Numerical solutions to the radiative transfer equation are typically computationally expensive. The large expense arises because the solution has a high dimensionality with $NM$ degrees of freedom, where the $N$ and $M$ arise from spatial and angular degrees of freedom, respectively. Here, a numerical method is presented that aims for fast and low-memory calculations, in the sense of computational cost and memory requirements of only $O(N)$. The method uses a discontinuous Galerkin (DG) spectral element method and hp-adaptive mesh refinement to reduce the number of spatial degrees of freedom from $N$ to $n$, thereby reducing the total cost and memory to $nM$, with the aim of achieving $nM$ approximately equal to $N$. After this reduction in memory to $O(N)$, in order to ensure a computational cost of $O(N)$, a suitable preconditioner is identified and utilized. Numerical examples are presented in two spatial dimensions to allow calculation of high-resolution reference solutions for comparison, while the methodology is general and applies in either two or three spatial dimensions. The numerical examples show large memory reduction ratios $N/n$ and fast $O(N)$ computational cost. A variety of examples is shown, including smooth spatial variations or steep gradients, and Rayleigh (isotropic) or Mie (anisotropic) scattering. The methods could enable more tractable computations for many applications, such as medical imaging and weather and climate prediction.

Keywords: spectral element | discontinuous Galerkin | hp-adaptivity | three-dimensional (3D) radiative transfer

1. Introduction

1.1. Background and motivation

The radiative transfer equation is important in a variety of applications, such as astrophysics [87, 76, 92], medical imaging [54, 56, 84, 2], neutron transport [81, 60, 61], underwater imaging [77, 48], satellite remote sensing [7, 15, 83], and weather and climate prediction [10, 98, 44, 45].

A large computational expense arises in radiative transport because the radiative intensity, $I(x, y, z, \theta, \phi)$, is a function of five or more coordinates, including spatial coordinates $(x, y, z)$ and angular coordinates $(\theta, \phi)$ that characterize different directions of photon propagation. Additional coordinates may also arise, for example, from different frequencies $\nu$ of electromagnetic radiation or from evolution in time $t$. In any of these cases, radiative transfer is a high-dimensional problem. In discrete form for numerical calculations, storage of $I(x, y, z, \theta, \phi)$ would typically require $NM$ degrees of freedom, where $N$ and $M$ are the number of spatial $(x, y, z)$ and angular $(\theta, \phi)$ degrees of freedom, respectively. The value of $M$ can also be even larger, for example, due to different wavelengths of electromagnetic radiation.

In many applications, such as weather and climate prediction, the computational cost of radiative transfer is so large that a comprehensive treatment is abandoned. To decrease the computational cost, simplified treatments are often adopted (see, e.g., the ECMWF model [44, 45] and the RRTMG model [65, 11, 75]). For example, one common simplification is the two-stream approximation (also called plane-parallel or one-dimensional), which resolves photon propagation in only two directions or streams.

In the present paper, however, the aim is to avoid these traditional simplifications and to instead present methods to solve for the full quantity $I(x, y, z, \theta, \phi)$. Such an approach is often called three-dimensional (3D) radiative transfer [7] to emphasize that photons may propagate in any direction. Here, we investigate numerical methods for the partial differential equation (PDE) of radiative transfer, with
the aim of a cost that is not the prohibitive $O(NM)$ but is instead only $O(N)$. In order to achieve such a savings, a cost reduction factor of $O(M)$ is needed in order to counteract the increased cost of $O(M)$ from the angular degrees of freedom. To do so, the goal is to reduce the number of spatial degrees of freedom from $N$ to $n$, so the overall cost, $nM$, is approximately $N$.

If such a reduction were enacted naively—for instance, by simply increasing the grid spacing of a finite difference or finite volume method—then the result of the reduction from $N$ to $n$ would be a substantial loss of accuracy. However, there are other computational strategies that can represent functions with greater flexibility and reduced cost. Here it is shown that a discontinuous Galerkin (DG) spectral element method can be used to obtain the goal of adequate accuracy with only $n$ spatial degrees of freedom in a variety of scenarios. A suitable preconditioner is also needed in order to ensure that the overall cost is $O(N)$. In addition, in complex cases of Mie (anisotropic) scattering or steep spatial gradients in the absorption or scattering coefficients, an $hp$-adaptive refinement strategy brings further improvement, and facilitates more general applicability, as described in more detail below.

The use of $hp$-adaptivity here is one of few examples of this technique for the radiative transfer equation, or for related equations of kinetic theory such as the Boltzmann equation or neutron transport equation [39, 95, 32]. While adaptivity in $h$ has provided sound results for radiative transfer [52, 30, 74], we find here that $hp$-adaptive refinement is important, rather than $h$-adaptive refinement or $p$-refinement alone, for achieving a method that is both low-memory and fast, for a range of scenarios including steep gradients and Mie (anisotropic) scattering. The technique of $hp$-adaptivity has also been used on an approximation of the radiative transfer equation called the simplified $P_N$ approximation [59, 35, 36], and on other equations such as the convection–diffusion equation [90, 104, 34].

There are other memory-saving techniques which can be used for numerically solving radiative transfer equations, such as sparse grid [99], dynamical low-rank approximation [57], $P_N$ approximation [35, 36], or truncated approximation to the scattering phase function [31, 85]. These techniques can provide accurate results in certain cases, and could potentially benefit from combined use with $hp$-adaptivity.

Without $hp$-adaptivity, the use of high-$p$ methods such as spectral methods or DG methods have been advanced in recent years and can provide accurate solutions in many scenarios [53, 102, 103, 3, 42, 23, 29, 49]. However, it is known that steep gradients can present difficulties for non-adaptive high-$p$ methods. Since steep gradients are an important aspect in many applications (such as medical imaging or clouds in the atmosphere), here we investigate $hp$-adaptive methods for general use.

1.2. The radiative transfer equation

In the paper we consider the time-independent radiative transfer equation,

$$\mathbf{s} \cdot \nabla I(\mathbf{x}, \mathbf{s}) + \beta_e(\mathbf{x}) I(\mathbf{x}, \mathbf{s}) = \beta_e(\mathbf{x})(1 - \tilde{\omega}(\mathbf{x})) B(\mathbf{x}, \mathbf{s}) + \frac{\beta_e(\mathbf{x}) \tilde{\omega}(\mathbf{x})}{|S|} \int_S p(s, s') I(\mathbf{x}, s') ds', \quad (1a)$$

for $x \in \Omega \subset \mathbb{R}^d$. In Eq. (1), $I(\mathbf{x}, \mathbf{s})$ represents the radiation intensity at $\mathbf{x}$ and along the direction $\mathbf{s} \in S := S^d$, where $S^d$ denotes the unit sphere in $\mathbb{R}^d$, and $|S|$ denotes the area of $S$. The parameters $\beta_e(\mathbf{x})$, $\tilde{\omega}(\mathbf{x})$, and $B(\mathbf{x}, \mathbf{s})$ represent the extinction parameter, the single-scattering albedo, and the black body radiation, respectively. Finally, $p(s, s')$ denotes the scattering phase function which has the following form:

$$p(s, s') = \frac{1 - g^2}{c(1 + g^2 - 2g \cos \text{ang}(s, s'))^{3/2}}. \quad (1b)$$

This is the Henyey-Greenstein phase function and is most widely used as a model phase function for anisotropic scattering [73]. In Eq. (1b), $\text{ang}(s, s')$ represents the angle between the directions $s$ and $s'$, while $g$ is the asymmetric parameter which typically takes values in $[0, 1]$. For isotropic scattering we have $g = 0$, while for strongly forward scattering case such as the scattering of short-wave (solar) radiation in water clouds, $g$ can takes values from 0.8 to 0.9 [91].

Equation Eq. (1a) is not uniquely solvable until boundary conditions are included. Here we consider the inflow boundary condition

$$I(s) = I^{bd}(s) \quad \text{for } s \cdot \mathbf{n}_f \leq 0 \quad \text{on } \Gamma := \partial \Omega, \quad (1c)$$

where $\mathbf{n}_f$ is the normal vector pointing outward from the domain $\Omega$. For a priori estimates of solutions of Eq. (1), we refer to [27, 28] and the references therein.
1.3. Considerations for Memory, Cost, and Accuracy

In this section we aim to give a heuristic estimate for the memory reduction ratio by using DG spectral element method, compared to traditional low-order methods such as finite-volume/finite-difference. The strategy here involves a reduction in the number of spatial degrees of freedom from $N$ to $n$, where $N$ can be taken as the number of grid cells from a uniform grid if one were to use a typical finite difference or finite volume method. With such a reduction, it is important to ensure that adequate accuracy can still be maintained. As a preliminary consideration, the balance between cost and accuracy can be estimated heuristically as follows.

Suppose that polynomial accuracy is desired with respect to $N$, so that the error $E$ scales as

$$E \sim CN_1^{-q},$$  \hspace{1cm} (2)

where $q$ is the polynomial order, $C$ is a constant, and $N_1$ is the number of degrees of freedom in one spatial dimension, so that $N_1^d = N$, where $d$ is the number of spatial dimensions. For instance, $d = 3$ for three-dimensional space $(x, y, z)$, and $q = 1$ for first-order accuracy, $q = 2$ for second-order accuracy, etc. The error is then $CN_1^{-1}$ for first-order accuracy, $CN_1^{-2}$ for second-order accuracy, etc.

Note that the desired accuracy in Eq. (2) is only polynomial accuracy, even though a DG spectral element method is proposed here. As is well known, typical use of the spectral element method would result in a much smaller error with decay which could be as fast as exponential decay rather than only polynomial decay [72, 4]. Here, however, the high accuracy of the DG spectral element method will be used in a different way, not because it provides minuscule errors, but instead to provide an adequate accuracy for a greatly reduced cost and reduced degrees of freedom.

In terms of the reduced degrees of freedom, $n$, by using a DG spectral element method, it is possible to achieve errors that decay exponentially, proportional to

$$E \sim \exp(-cn_1),$$  \hspace{1cm} (3)

for constant $c$, and where $n_1$ is the number of degrees of freedom in one dimension, so that $n_1^d = n$ [72, 4]. Note that this is an idealized case which might not be true in certain cases, such that when the solution has poor regularity. However, it suffices as a heuristic estimate of the potential of high-order DG spectral element methods in reducing memory.

By comparing the estimates above in terms of $N$ and $n$, from Eq. (2) and Eq. (3), one can estimate how small $n$ can be, or how large $M$ can be. The value of $n$ can be estimated by comparing the $n$-based error from DG spectral elements with the desired error,

$$\exp(-cn_1) \sim CN_1^{-q},$$

which would ensure the polynomial accuracy with respect to $N$. By taking a logarithm, one can rewrite this as

$$n_1 \sim \frac{q}{c} \log N_1,$$  \hspace{1cm} (4)

or

$$n \sim \left(\frac{q}{cd} \log N\right)^d,$$  \hspace{1cm} (5)

where $C$ was assumed to be small compared to $N_1^q$. If $n$ is chosen to follow this scaling, or to be larger, then the desired accuracy of order $q$ should be achieved. The value of $M$ can then be as large as

$$M \sim \frac{N}{n} \sim \left(\frac{cd}{q}\right)^d \frac{N}{(\log N)^d}.$$  \hspace{1cm} (6)

The above estimate suggests that the memory reduction ratio ($N/n$, or $M$) should be more significant if we have a larger system size $N$, or if a higher precision level is desired (by Eq. (2)). In addition to the system size and the precision level, the savings of the memory could also depend on the factors such as the types of the error norms (e.g., $L^2$ or $L^\infty$) being used, the regularity of the solution, the quality of the error estimators and the refinement strategies (for AMR methods), and so on. A systematic investigation of this topic exceeds the scope here. In this introductory section we aim for an approximate estimate of the memory reduction ratio with respect to the size of the system.

Consider an example of how large $M$ could be, based on Eq. (6). For instance, for a three-dimensional case with first-order accuracy, suppose $d = 3$, $q = 1$, and $c = 1$. If the solution is desired on a grid with
100 grid points per spatial dimension, then \( N_1 = 100 \) and \( N = N_1^d = 10^6 \). Inserting these values into Eq. (6) leads to

\[
M \sim 3^3 \frac{10^6}{(\log 10^6)^3} \approx 10^4.
\] (7)

If, instead, second-order accuracy is desired, then this \( M \) value would be reduced by a factor of \( q^d = 2^3 = 8 \) but would still be a large value of over 1,000.

While the estimates above are for accuracy and memory, a further consideration is cost. At best, for a numerical solution with \( N \) degrees of freedom, one can aim for a cost of \( O(N) \) in terms of the number of floating-point operations. Additional cost can potentially arise, depending on the numerical methods and the matrix solver for the corresponding linear system. For the present paper, DG spectral element methods are used with \( hp \) refinement, and the cost is expected to be larger than but close to \( O(N) \). Numerical experiments will be used below to examine the cost in several test cases.

For practical applications, the values of \( M \) from Eq. (6) and Eq. (7) are large enough for many demanding scenarios. For example, consider an application in weather prediction. One contribution to \( M \) is the number of wavelength or frequency bands for the \( \nu \) coordinate of \( I(x, y, z, \theta, \phi, \nu) \), and this number can be as large as 200 or 300 [44, 45]. Another contribution to \( M \) is the number of angular dimensions, or streams. If a ten-stream method is used [50, 51], then the total value of \( M \) would be \( M = 2000 \) to 3000; or, if the angular dimensions are better resolved with 20 or 30 streams, then the value of \( M \) could become nearly 10,000. These challenging scenarios are within the realm of the estimates from Eq. (6) and Eq. (7), and they suggest that the present methods could be applicable in practical settings.

1.4. Organization of the paper

In Section 2, we introduce the numerical methods, including the choices of the spectral element (SE) approximation spaces, the discontinuous Galerkin (DG) schemes, and the iterative solvers. The discussion on \( hp \)-adaptivity is put in Section 3, including the error and smoothness estimator, and the refinement strategy. Finally in Section 4, we present numerical experiments, in both the Rayleigh and Mie scattering cases.

2. Numerical Methods

In this section, we describe the numerical approach that is motivated by the goals of being fast and low-memory. For a low-memory representation of the solution, a DG spectral element method is first introduced. Then, for a fast solver, options for iterative methods and preconditioners are discussed.

2.1. Low-memory representation by DG spectral element

The solution \( I \) of Eq. (1) can present both spatial and angular inhomogeneity. Taking the cloud radiation model as an example, while the spatial inhomogeneity is caused by the inhomogeneous distribution of clouds in the atmosphere, the angular inhomogeneity is a result of the collimated property of the sunbeam and the large asymmetry parameter of cloud droplets. As a consequence, from the perspective of finding optimal approximation spaces, adopting spectral elements (SE) is ideal in the sense that it combines the advantages of the spectral methods and the finite element (FE) methods, allowing an efficient low-memory representation of the solution at the places where the solution is smooth, while keeping the flexibility of finite element at resolving local features of the solution.

Here we demonstrate the main procedures of constructing the spectral element DG space for Eq. (1). For simplicity, we consider the two-dimensional radiative transfer model used in [55, 62, 38], for which we have \( I = I(x, y, \theta) \) with \( (x, y) \in \Omega \) and \( \theta \in [0, 2\pi] \), and the radiation direction vector \( s = (\cos \theta, \sin \theta) \). Let \( \mathcal{T}_h \) be a non-conforming triangulation of \( \Omega \) by rectangular cells. We require that \( \mathcal{T}_h \) is 1-irregular, i.e., for any element \( K \in \mathcal{T}_h \), there are at most two neighbour elements connecting to \( K \) through each edge. Here \( h \) is both an index for the triangulations \( \mathcal{T}_h \) and also the mesh-size, which is defined as \( h := \max_{K \in \mathcal{T}_h} h_K \) where \( h_K \) represents the diameter of \( K \). For each \( K \in \mathcal{T}_h \), we associate it with an angular discretization, denoted by \( \mathcal{T}_{h_{\nu}} \), of the angular space \([0, 2\pi]\). Then we have

\[
\Omega \times [0, 2\pi] = \cup_{K \in \mathcal{T}_h} \cup_{K_{\nu} \in \mathcal{T}_{h_{\nu}}} K \times K_{\nu}.
\]

Since \( K \) is rectangular, for each spatial–angular element \( K \times K_{\nu} \), we have the following decomposition:

\[
K \times K_{\nu} = [x_0^K, x_1^K] \times [y_0^K, y_1^K] \times [\theta_0^K, \theta_1^K]
\]
The above decomposition allows us to construct a basis on $K \times K^a$ based on tensor products:

$$
\phi_m^{K \times K^a}(x, s) = \phi_{m_1}^{K}(x) \phi_{m_2}^{K^a}(y) \phi_{m_3}^{K^a}(\theta),
$$
where

$$m_1 = 1, ..., p_x^K + 1, \quad m_2 = 1, ..., p_y^{K^a} + 1, \quad m_3 = 1, ..., p_\theta^{K^a} + 1.
$$

Let $V(K \times K^a)$ be the space spanned by the above basis. The global approximation space is defined as

$$V_h := \prod_{K \in T_h} \prod_{K^a \in T_h^a} V(K \times K^a),$$

Then the numerical solution $I_h$ can be written as

$$I_h = \sum_{K \in T_h} \sum_{K^a \in T_h^a} \sum_m I_m^{K \times K^a} \phi_m^{K \times K^a}(x, s),$$

where $I_m^{K \times K^a}$ represents the DOFs of $I_h$ on the element $K \times K^a$.

Note that we use piece-wise polynomials to approximate the angular variation of the radiation intensity $I$. This approximation setting was known as angular finite element in some references [9]. Here we shall call it angular discontinuous Galerkin (DG) to emphasize the fact that no continuity is enforced among any neighbour angular elements. If the polynomial degree is fixed to be zero, then the angular DG becomes the (angular) finite volume method [66], which is equivalent to the discrete ordinate method with a special choice of the quadrature rules [33, 58]. The angular discretization can also have an effect on the choice of the best iterative solver. The current paper focuses on the spatial $hp$-AMR techniques, and it could be combined with any choice of angular discretization.

When $d = 3$, a direct discretization based on the polar and azimuth angles can lead to a non-uniform and inefficient meshing of the angular sphere due to the singularity presented at the north/south poles. In this case, it could be better to consider the discretization based on a different parameterization rule of the angular sphere. For instance, the cubed sphere approach [89, 86] parameterizes the unit sphere without introducing any singularities at the poles. Alternatively, one can also discretize directly the unit sphere [66, Section 16.6].

### 2.2. Discontinuous Galerkin

Discontinuous Galerkin (DG) methods were originally proposed for neutron transport in [82]. Then they were applied to a wide range of problems including convection-diffusion [14, 8], Stokes and Navier-Stokes [12, 5], elasticity [41, 78, 25], and Maxwell’s equations [13, 43, 26]. The versatility of DG originates from the fact that it combines the advantages of both finite volume and finite element methods. To be more specific, DG can be thought as a generalization of finite volume methods to arbitrary high-order, while on the other hand, it extends the classical class of finite element methods by relaxing or adjusting their-inter-element conformity, by its flexible choices of the numerical traces.

We next derive a DG method for Eq. (1) based on the DG spectral element approximation space $V_h$ introduced above. To begin with, we multiply Eq. (1a) with a test function $v$ and integrate it on a spatial-angular element $K \times K^a$:

$$
\int_{K^a} \int_K (\mathbf{s} \cdot \nabla I)v = - \int_{K^a} \int_K \beta_e I v + \int_{K^a} \int_K \beta_s (1 - \tilde{\omega}) B v + \int_K \frac{\beta_s \tilde{\omega}}{|S|} \int_{K^a} \int_S p(s, s') v(s) I(s') ds' ds.
$$

By an integration by parts of the left term of the above equation, and then symbolically replacing $I$ in $K$ and $I$ on $\partial K$ by the approximation $I_h$ and $\tilde{I}_h$, respectively, we obtain the following general form of DG methods: find $I_h \in V_h$ such that

$$a_h(I_h; v_h) = F(v_h) \quad \forall v_h \in V_h,
$$

$$\tag{8a}$$
where

\[
a_h(I; v) := \sum_{K \in T_h} \sum_{K^\ast \in T_{h^k}} \left( \int_{K^\ast} \int_{\partial K} \hat{I} v (s \cdot n) \right) \quad (8b)
\]

\[
- \int_{K^\ast} \int_{K} I s \cdot \nabla v + \int_{K^\ast} \int_{K} \beta_e I v \\
- \frac{\beta_e \tilde{\omega}}{|S|} \int_{K^\ast} \int_{S} p(s, s') I(s') v(s) ds' ds,
\]

\[
F(v) := \sum_{K \in T_h} \sum_{K^\ast \in T_{h^k}} \int_{K^\ast} \int_{K} \beta_e (1 - \tilde{\omega}) B v. \quad (8c)
\]

The numerical trace \( \hat{I} \) needs to be specified to complete the definition. Here we consider the upwinding form:

\[
\hat{I}(s) = \begin{cases} 
I^{\text{out}}(s) & \text{if } s \cdot n_{\partial K} < 0, \\
I^{\text{in}}(s) & \text{if } s \cdot n_{\partial K} \geq 0,
\end{cases} \quad (8d)
\]

where \( I^{\text{in}} \) is the restriction of \( I \) on \( \partial K \) from inside, and \( I^{\text{out}} \) comes from either the neighbouring elements or the boundary data \( I^{\text{bd}} \). Other choices of the numerical fluxes exist, such as those introduced in [100, 101, 79], which can give positive and asymptotic-preserving schemes.

For many atmospheric models, the coefficients such as \( \beta_e, \tilde{\omega} \) are provided only on a grid of points. Therefore, to solve Eq. (8), we will have to first interpolate the data on the uniform grid to the \( hp \)-AMR grid \( \mathcal{T}_h \). Then, after solving Eq. (8) and obtaining the desired quantities (for instance, the radiative heating rate), we interpolate these data back to the original grid. Note that the computational cost of the interpolation is usually \( \mathcal{O}(N) \) while the computational cost of solving Eq. (8) is \( \mathcal{O}(nM) \). Since we aim for reducing \( N \) to \( n \) such that \( nM \approx N \), the cost of the interpolation is at most the cost of solving the reduced system. In practice, considering that solving a linear system of size \( N \) usually has a cost of more than \( N \), the cost of the interpolation should be small compared to the cost of solving Eq. (8) in most cases.

2.3. Iterative method and preconditioner

Equation Eq. (8) can be rewritten into the following compact form:

\[
A[I_h] = F + S[I_h],
\]

where \( [I_h] \) is an array representing the DOFs of \( I_h \), the matrix \( A \) represents the summation of the advection and the extinction, the matrix \( S \) represents the scattering operator, and the array \( F \) represents the black-body radiation; see the appendix for details on how \( A, S, \) and \( F \) are calculated.

Since \( [I_h] \) usually involves a large number of DOFs and the operator \( A \) is sparse by the locality of the advection operator, Eq. (9) are solved by iterative methods in most cases. A popular choice (although not our choice) of solving Eq. (9) is by source iteration, which starts with an initial guess \( I_h^0 \), and then repeatedly calculates

\[
[I_h^{n+1}] = A^{-1} (F + S[I_h^n]),
\]

where each iteration step requires an inversion of \( A \) usually manifested by a sweeping algorithm combined with an element-reordering strategy [70, 39]. Sweeping becomes difficult to implement if the triangulation of the spatial-angular domain is adapted differently throughout space [19].

Therefore, here we consider an alternative approach by a (Preconditioned) Krylov subspace methods, which can be shown faster than source iteration in many cases. To be more specific, we adopt the restarted Generalized Minimal RESidual (GMRES(\( k \)) method [88] to solve Eq. (9). Here \( k \) is the number of iterations before a restart. Since the cost of the iterations grows fast in \( \mathcal{O}(n^2) \) with \( n \) as the iteration counts, the restarted GMRES helps save memory cost. To speed up the convergence of the GMRES, we consider a right preconditioner by \( A^{-1} \), where the inversion is calculated by a multifrontal LU factorization such as UMFPACK [21, 20]. Note that only \( A \) needs to be explicitly assembled for an inversion. The scattering operator \( S \) can be applied independently for each spatial SE quadrature point so the full matrix \( S \) does not need to be explicitly assembled. For more on the iterative methods, such as those acceleration techniques in the diffusion and the Fokker-Planck/angular-diffusion regimes, we refer to [1, 67, 97, 24] and the references therein.
3. \(hp\)-adaptivity

In order to allow more general applicability to problems with steep gradients, an \(hp\)-adaptive mesh refinement method is presented in this section. The aim is to obtain a mesh and polynomial approximation setting such that the solution can be most efficiently represented. To achieve this, we will need

1. An error estimator \(\eta_K\) which gives an estimate for the error on each element \(K \in T_h\).
2. A refinement strategy which utilizes the error estimator to instruct on which elements need to be refined.
3. A \(hp\)-steering criteria for determining whether to perform local \(h\) or \(p\) refinement.

For the rest of this section we discuss these three ingredients of \(hp\)-adaptivity with more details.

3.1. Error estimator

For the rest of this section we discuss these three ingredients of \(hp\)-adaptivity with more details.

Since DG methods are used for the discretization of the radiative transfer equation, the solution \(I_h\) has jumps across the interfaces between neighbour elements. As the mesh is refined, these jumps approach zero at about the same rate of convergence as the numerical solution. This fact makes these jumps a natural choice for error estimators:

\[
\eta_K := \left( \frac{1}{|\partial K|} \int_{\partial K} \|I_h\|^2 \right)^{1/2},
\]  

where the jump notation is defined as \([I_h] := I_h^+ n^+ + I_h^- n^-\), and \(|\partial K|\) represents the area (\(d = 3\)) or the length (\(d = 2\)) of \(\partial K\). The jump-based estimator is inexpensive to compute and stands out for its simplicity. Here we only use the intensity moment of \(I_h\), which is justified in [96]. In addition, using only the intensity moment of \(I_h\) helps us focus on spatial adaptivity.

There exist other types of estimators that could be more efficient, but they all require extra computational expense. For instance, many goal-oriented estimators are designed based on the observation that in many applications, only partial statistics of the solution are needed (e.g. solution on a small fraction of the domain, or solution in certain angles). This observation motivates the usage of duality-based estimators, which require one to first solve a dual system. We refer to [6] for more details. Another type of estimator requires the solution on a more refined mesh, such as \(T_{h/2}\), and then compare \(I_h\) with \(I_{h/2}\) to estimate the error. Apparently, this estimator can give a good estimate since \(I_{h/2}\) is a much better approximation to the exact solution \(I\) when it is compared to \(I_h\). But the extra computational cost of obtaining \(I_{h/2}\) is much more expensive than evaluating the jumps.

3.2. Refinement strategy

Now we present our \(hp\)-refinement algorithm. The given quantities are a tolerance \(\text{TOL} > 0\), a maximum iteration number \(N_{\text{iter}}\), an initial mesh \(T_h^{(0)}\), and an initial polynomial degree distribution \(p^{(0)}\). Setting \(l = 0\) to start, our refinement algorithm can be formulated as the following steps:

1. Calculate the solution \(I_{h,p}^{(l)}\) based on the \(hp\)-mesh \((T_h^{(l)}, p^{(l)})\).
2. Calculate the error indicator \(\eta_K\) and the global indicator \(\eta = \sum_{K \in T_h} \eta_K\).
3. Check whether the stopping criterion is satisfied. Namely, if \(\eta \leq \text{TOL}\) or \(l \geq N_{\text{iter}}\), then STOP the iteration.
4. Sort \(\eta_K\) in increasing order. Mark the largest \(r_{\text{ref}}\) (10\% \~ 20\%) percentage of the elements for refinement.
5. For the marked elements, use the \(hp\)-steering criteria (see Section 3.3) to determine whether an \(h\)-refinement or a \(p\)-refinement should be performed. This step gives us \((T_h^{(l+1)}, p^{(l+1)})\).
6. Set \(l = l + 1\) and GOTO step 1.

If performing \(h\)-refinements naively, the 1-irregular criteria for the mesh \(T_h\) can be violated. An \(hp\)-mesh violating the 1-irregular criterion can potentially offer stronger representational capacity but will also suffer from an increased complexity of the inter-element communication. Therefore, a balance point between mesh representational capacity and inter-element communication efficiency is needed. Here we adopt the 1-irregular criterion for its simplicity and consider it as our initial step of the exploration. To guarantee that the 1-irregular criterion is met throughout the refinement procedure, we adopt a recursive refinement subroutine. Namely, if an element \(K\) is instructed to be refined, we shall also refine its neighbour elements if the child elements of \(K\) and the neighbour elements of \(K\) violate the 1-irregular...
criterion. This procedure will be recursively performed such that the whole mesh can be guaranteed to satisfy the 1-irregular criterion. See Figure 1 for a visualization.

For p-refinement, a similar trade-off exists between mesh representational capacity and inter-element communication efficiency. Suppose the maximal degree is $p_{\text{max}}$. Then the total number of the different types of the inter-element communication matrices is $O(p_{\text{max}}^3)$ for the 1-irregular criterion. In general, for an n-irregular mesh, this number is $O(p_{\text{max}}^{2n+1})$. When $p_{\text{max}}$ is large, it quickly becomes infeasible to store these communication matrices. In this case, these matrices have to be calculated on-the-fly, which can lead to a slow-down for the sweeping algorithm or the assembly of the mass or stiffness matrices. To avoid this issue, we adopt the even-degree criterion for the $hp$-mesh, which reduces the number to $O(p_{\text{max}}^2)$. To be more specific, if an element $K$ connects to multiple neighbour elements through an edge, then we require that these neighbour elements have the same degree. When implementing the p-refinement, if an element $K$ is marked to be refined, we check whether even-degree criterion is met between $K$ and its neighbours. If the criterion is not met, we perform p-refinement for the neighbour elements of $K$ which share the same mesh-size. This procedure is recursively carried out and repeated until the even-degree criterion is met for the whole mesh. See Figure 2 for a visualization.

3.3. hp-steering criteria

To complete the refinement strategy presented in the previous subsection, it remains to specify the $hp$-steering criteria so that we can determine whether $h$ or $p$ refinement should be performed. Here we consider a method estimating the local regularity of the solution by investigating the decaying pattern of the coefficients obtained from a Legendre expansion of $I_h$. Once an estimate of the solution regularity is obtained, p-refinement will be performed if the solution is indicated to be smooth enough, or otherwise we will adopt h-refinement. Here we briefly explain how this regularity estimate is done; more details can be found in [47]. For simplicity, we consider a solution $u$ defined on a reference element $[-1,1]^2$. Similar arguments apply for any physical element $[x_0,x_1] \times [y_0,y_1]$. On $[-1,1]^2$, $u$ can be expressed as a linear combination of the tensor basis $h_m(x)h_n(y)$:

$$u(x,y) = \sum_{m=1}^{p_x+1} \sum_{n=1}^{p_y+1} u_{mn} h_m(x)h_n(y).$$

Our goal is to estimate the regularity of $u$ based on the coefficients $u_{mn}$. To proceed, we let $\{L_k\}_{i=0}^{\infty}$ be the Legendre polynomial on $[-1,1]$. Then $u$ can have another expansion:

$$u(x,y) = \sum_{i,j=0}^{\infty} a_{ij} L_i(x)L_j(y).$$
Then we can calculate $a_{ij}$ as follows:

$$ a_{ij} = \frac{(2i + 1)(2j + 1)}{4} \int_{[-1,1]^2} u(x,y)L_i(x)L_j(y)dxdy $$

$$ = \frac{(2i + 1)(2j + 1)}{4} \sum_{m=1}^{p_x+1} \sum_{n=1}^{p_y+1} u_{mn} \int_{[-1,1]^2} h_m(x)h_n(y)L_i(x)L_j(y)dxdy. $$

Note that the spectral element basis $\{h_j\}_{j=1}^{p+1}$ are associated with the LGL quadrature points and weights $\{\xi_j, w_j\}_{j=1}^{p+1}$. Thus,

$$ a_{ij} \approx \frac{(2i + 1)(2j + 1)}{4} \sum_{m=1}^{p_x+1} \sum_{n=1}^{p_y+1} u_{mn} w_m w_n L_i(\xi_m)L_j(\xi_n) \quad \text{for} \quad i = 0, \ldots, p_x, \quad j = 0, \ldots, p_y. $$

Since the LGL quadrature is exact for polynomials of degree up to $2p - 1$ [93], the above approximation sign can be replaced by an equal sign when $i < p_x$ and $j < p_y$. Also, by the definition of Legendre polynomials, we know $a_{ij} = 0$ when $i > p_x$ or $j > p_y$. Once $a_{ij}$ are obtained, one can estimate the regularity of $u$ by using the method introduced in [46, 47]. Namely, we compute

$$ (a^2)^2 := \sum_{j=0}^{\infty} |a_{ij}|^2 \frac{2}{2j + 1}, \quad (a^p)^2 := \sum_{i=0}^{\infty} |a_{i,j}|^2 \frac{2}{2i + 1}, \quad l_p = \frac{1}{2} \left( \frac{\log(\frac{2p_x+1}{2\log p_x})}{\log p_x} \right) \left( \frac{\log(\frac{2p_y+1}{2\log p_y})}{\log p_y} \right). $$

Then $u$ is predicted to belong to $H^{l_p - \frac{1}{2}}([-1,1]^2)$ where $0 < \epsilon \leq l_p - \frac{1}{2}$. If $k \geq 0$ is an integer index, then the weighted Sobolev space $H^k_w([-1,1]^2)$ is defined as follows:

$$ H^k_w([-1,1]^2) = \left\{ u \in L^2([-1,1]^2) : \sum_{i=0}^{k} \int_{[-1,1]^2} |D^{(i)}u|^2 w \ dx < \infty \right\}, $$

where the weight function is $w(x,y) := (x + 1)^2(1 - x)^2$. The space $H^k_w$ with non-integer $k$ is defined by the $K$-method of interpolation [46]. Now, we use the following criteria to determine whether an $h$ or a $p$ refinement should be applied: If $l_p = \frac{1}{2} \geq \frac{2p_x + 2p_y}{2} + 1$, we perform $h$-refinement, otherwise, we perform $p$-refinement.

For more on other $hp$-steering methods, we refer to [64].

4. Numerical Experiments

In this section we carry out numerical experiments to test the performance of the methods. In particular we investigate the memory reduction and cost, and we do so in different scenarios, to compare and contrast cases of smooth solutions versus steep gradients, and isotropic (Rayleigh) scattering versus anisotropic (Mie) scattering. First we consider Rayleigh scattering, and later Mie scattering.

4.1. Rayleigh scattering - smooth solution

For the scenario of Rayleigh (isotropic) scattering, the following scattering phase function is used:

$$ p(\theta, \theta') = \frac{1}{3\pi} \left( 1 + (\cos \theta \cos \theta' + \sin \theta \sin \theta')^2 \right). $$

This phase function is relatively smooth as a function of $\theta$ and $\theta'$ compared to the strongly-peaked phase function of anisotropic (Mie) scattering. Due to the phase function’s smoothness, Rayleigh scattering is a good first test to compare cases of spatially smooth solutions versus steep spatial gradients, without additional complicating factors.

In the first experiment, we consider a smooth exact solution $I_1(x,y,\theta) = I_{s1}(x,y)I_a(\theta)$, where

$$ I_{s1}(x,y) = 1 + \cos(\frac{2\pi x}{L_x}) \sin(\frac{\pi y}{L_y}), $$

$$ I_a(\theta) = \begin{cases} \sin(\frac{\pi}{2}(1 - \frac{2\theta}{\pi})) & \theta \in [0, \frac{\pi}{2}], \\ \left(\frac{2}{\pi}(\theta - \frac{3\pi}{2})\right)^3 & \theta \in \left[\frac{3\pi}{2}, 2\pi\right], \\ 0 & \text{else.} \end{cases} $$

9
Here $I_1$ is defined on the domain $[0, L_x] \times [0, L_y]$ with $L_x = 3$ and $L_y = 2$; see Figure 3 for a visualization of $I_1$. For this numerical test, the angular component $I_a$ is adapted from the one used in Experiment 1 of [38]. Dirichlet boundary conditions are used for the top and the bottom boundaries of the domain, while periodic boundary conditions are applied for the left and right sides. We set $\beta_c = 2$ and $\tilde{\omega} = 0.5$, and use the method of manufactured solution. Namely, we define $B$ so that $I_1(x, y, \theta)$ satisfies Eq. (1a):

$$B(x, y, \theta) = \frac{1}{\beta_c (1 - \tilde{\omega})} \left( s \cdot \nabla I + \beta_c I - \beta_c \tilde{\omega} \frac{1}{|S|} \int_S p(s', s') I(s') ds' \right).$$

Then, the explicit functional formula for $B$ is obtained by using Matlab’s symbolic computation toolbox to evaluate the derivatives and integrals in this expression for $B$. We fix the angular discretization with four elements where each element uses a $P_7$ approximation. In this way we ensure that the dominating error comes from the spatial discretization. We use the preconditioned GMRES (GMRESprc) iteration method which is described above. We stop the iteration when the relative $l^2$ error of the solution array reaches $10^{-10}$.

For $I_1$, we show in Figure 3 the performance of four methods: $h$-refinement, $p$-refinement, $h$-AMR, and $hp$-AMR. The $hp$-AMR method is the proposed method in its full generality, including $hp$-adaptive mesh refinement in space. For comparison, the $h$-AMR method is the same as $hp$-AMR except the polynomial degree is fixed as $P_0$ for the spatial approximation. As two additional comparisons, the $h$-refinement case uses a spatially uniform mesh with fixed $P_0$ approximation, and the mesh size is decreased by half for each refinement iteration; and the $p$-refinement case starts with a $4 \times 4$ uniform mesh with $P_0$ approximation and increases the polynomial order by 1 for each refinement iteration.

Figure 3 shows that the $hp$-AMR algorithm is both fast and low-memory. It is fast in the sense that the solver time is approximately a linear function of the number of degrees of freedom. It is low-memory in the sense that the memory reduction ratio, $N/n$, is large.

As further detail regarding the memory reduction ratio, recall from the heuristic estimate in Eq. (6) that $N/n$ is the ratio of, on the one hand, the number of spatial degrees of freedom $N$ for a finite difference or finite volume type of method, and, on the other hand, the number of spatial degrees of freedom $n$ for a low-memory method such as the proposed $hp$-AMR. To evaluate $N/n$, we consider an error $E$, and we define $n$ as the number of DOFs needed for a given method to achieve error $E$; so a different value of $n$ is found for each of the four methods ($h$-refinement, $p$-refinement, $h$-AMR, and $hp$-AMR). As the value of $N$, we use the estimated number of DOFs for a first-order method to reach the same error: $E = CN^{-1/d} = CN^{-1/2}$. Here $C$ is estimated by averaging the samples of $E/N^{-1/2}$ from the $h$-refinement experiments, since the $h$-refinement method uses a $P_0$ approximation and can therefore act as a typical finite difference or finite volume type of method; hence $N/n \approx 1$ in Figure 3 for the $h$-refinement method, since it is not a low-memory method. The $h$-AMR method also displays $N/n \approx 1$ in Figure 3 and is not a low-memory method. On the other hand, both the $hp$-AMR and $p$-refinement methods provide a large memory-reduction ratio $N/n$ and hence are low-memory methods, for this example solution $I_1$.

In essence, the low-memory property is a different interpretation of the well-known spectral accuracy property. A typical view of spectral accuracy is that one can achieve errors that decrease very rapidly (e.g., approximately exponentially), as shown in Figure 3, top-right. As a different viewpoint, one can say that spectral accuracy allows a certain error to be achieved with a large memory savings (Figure 3, bottom-right), relative to a typical finite difference or finite volume method. The memory reduction ratio can be written, based on the heuristic estimates from Eq. (6), as

$$\frac{N}{n} \sim \frac{1}{n} \exp \left( \frac{c d}{q} n^{1/d} \right),$$

which shows very rapid (roughly exponential) growth with respect to $n$. Hence an exponentially decaying error could be interpreted instead as an exponentially increasing savings in memory.

### 4.2. Rayleigh scattering - steep gradients

In the second experiment, as a test of the challenges associated with steep spatial gradients, we consider a different solution $I_2(x, y, \theta) = I_{2a}(x, y)I_a(\theta)$, where

$$I_{2a} = x(L_x - x) \left( L_y + y \tanh \left( 20(x - \frac{L_x}{2}) \right) \right).$$

10
Figure 3: Fast and low-memory numerical solution for the smooth exact solution $I_1$ with Rayleigh scattering. Top-left: Illustration of the mean intensity for the smooth exact solution $I_1$. Top-right: Relative $L^2$ error versus number of degrees of freedom in each spatial dimension, $n^{1/d}$, with $d = 2$ here. Bottom-left: Cost of the iterative solver versus total number of degrees of freedom, $nM$. The inset shows the scaling as fit to a power law. Bottom-right: Memory reduction factor, $N/nM$, versus number of degrees of freedom in each spatial dimension, $n^{1/d}$, with $d = 2$ here. Comparisons are shown for four different refinement strategies: $h$-refinement, $p$-refinement, $h$-AMR, and $hp$-AMR. The $hp$-AMR and $p$-refinement methods are fast and low-memory methods for this example, as shown by the approximately linear scaling in cost (bottom-left) and large amount of memory reduction (bottom-right).
The above function has a steep gradient near \( x = \frac{L}{2} \). Note that we avoid using \( \frac{L}{2} \) since this leads to a special case in which the interface aligns with element boundaries.

From Figure 4 for \( I_2 \), we observe again that \( hp\)-AMR is significantly more efficient than \( h \)-refinement and \( h \)-AMR, as was also the case for the smooth solution \( I_1 \). The main change due to steep gradients is that the cost of \( p \)-refinement is much higher (Figure 4, bottom-left) and it scales with \( n \) not as the fast, linear scaling of \( O(n) \) but as \( O(n^{1.42}) \). Figure 5 takes into account the different costs of different methods, and it shows a plot of error versus cost. From this perspective as well, \( p \)-refinement is less efficient than \( hp\)-AMR. Hence, with steep gradients, \( hp\)-AMR stands out as the only method here that is both fast and low-memory.

As some additional details of the behavior of the \( hp\)-AMR algorithm, for the \( I_2 \) case, we also plot the error landscape, the polynomial degree \( p \), and the regularity estimation index \( l_p \) (see Eq. (11)) in the last step of the \( hp\)-AMR refinement; see Figure 6. From this figure, we observe that the mesh-adaptive algorithm correctly captures the interface of the irradiance. For the error landscape, we plot both the real error and the estimated error, and we see that they share similar spatial structures and have similar magnitudes.

Furthermore, in designing a fast solver, we chose the preconditioned GMRES solver (GMRESprc) as the fastest solver among several options, as described above. Figure 7 shows a comparison of its performance against several other options: the original GMRES, source iteration (srcIter), and the Matlab default sparse matrix solver UMFpack. From Figure 7, we observe that the GMRES by itself is expensive and does not scale linearly with respect to \( n \). Both source iteration and GMRESprc are fast in
Figure 5: Error versus cost, as measured in solver compute time, for numerical solution $I_2$ from the case with Rayleigh scattering (see also Figure 4). The $hp$-AMR method achieves the lowest error for a given cost for errors smaller than $10^{-4}$.

Figure 6: Error landscape (real and estimated errors), polynomial degree, and regularity index, for $I_2$ numerical solution. Data collected at last step of the $hp$-AMR algorithm. Note that the numerical solution of $I_2$ and the numerical mesh are shown in Figure 4.
the sense that the solver time scales approximately linearly with respect to $n$. However, GMRESprc is overall faster than source iteration and is therefore used as the method of choice here. The Matlab solver UMFpack is the fastest solver when $n$ is small; however, as $n$ increases, UMFpack becomes slower than GMRESprc. We have observed that UMFpack uses a much larger amount of memory than GMRESprc, which could be the reason why UMFpack becomes slow when $n$ increases.

![Figure 7](image_url)

Figure 7: Cost in terms of solver computing time, as a function of the total number of degrees of freedom, $nM$, for the $hp$-AMR method for the solution $I_2$ with Rayleigh scattering. Comparisons are shown for four different iterative solvers: the original GMRES, source iteration (srcIter), the Matlab default sparse matrix solver (UMFpack), and a preconditioned GMRES solver (GMRESprc). Two solvers are fast in the sense of approximately linear $O(n)$ scaling of the cost—srcIter and GMRESprc—and the overall cost of GMRESprc is lower than srcIter.

### 4.3. Test 3 - Mie Scattering

As another test case, we now consider the challenges that arise from Mie (anisotropic) scattering, for which the scattering phase function is highly peaked as a function of the angular coordinate. For such a case, we use the Henyey-Greenstein scattering phase function in Eq. (1b) with $g = 0.8$. We consider the domain $[0, L_x] \times [0, L_y]$ with $L_x = 3$ and $L_y = 2$, and incident radiation coming from the top boundary:

$$I_3(x, L_y, \theta) = \begin{cases} \frac{16}{\pi} & \text{if } \theta \in \left[\frac{3}{2} \pi, \frac{3}{2} \pi + \frac{\pi}{16}\right], \\ 0 & \text{else,} \end{cases}$$

where the solution is denoted $I_3$ for this test case. No incident radiation is applied on the bottom boundary, and periodic boundary conditions will be used for the left and right sides of the domain. The single scattering albedo will be fixed as $\tilde{\omega} = 10^{11}$. We chose the extinction parameter $\beta_e$ to be

$$\beta_e = \frac{1.1}{1 + \exp \left(-2k_0 r(x, y)\right)}, \text{ where}$$

$$r(x, y) := \frac{L_y}{5} - \sqrt{(x - \frac{L_x}{2})^2 + (y - \frac{L_y}{2})^2}.$$  \hspace{1cm} (13)

See Figure 8 for an illustration of the shape of the scatterer, which is localized in space, similar to an idealized cloud in the atmosphere, and has steep spatial gradients. Since no exact solution is provided in this case, we use the error indicator $\eta_K$ to estimate the global $L^2$ error by calculating $\sqrt{\sum_{K \in T_h} \eta_K^2 |K|}$, where $\eta_K$ is given in Eq. (10). For the angular discretization, we use 32 elements with $P_0$ approximation, which is equivalent to a discrete ordinate discretization for the angular space, and which we use in order to focus here on the spatial $hp$-adaptivity. For the spatial discretization, we again consider the four different refinement methods, namely, $h$-refinement, $p$-refinement, $h$-AMR, $hp$-AMR, and we use the preconditioned GMRES and stop the iteration at the accuracy level $10^{-10}$.

For this case of Mie scattering, results of the numerical experiments are shown in Figure 9. In this figure, we observe essentially the same conclusions as in the test case of Rayleigh scattering in Figure 4. In particular, the $hp$-AMR method is fast and low-memory, in the sense that the solver cost is $O(n^{1.16})$ and hence nearly $O(n)$ (Figure 9, bottom-left), and the memory-reduction ratio is large (Figure 9, bottom-right).
Figure 8: Illustration of the extinction parameter, $\beta_e$, defined in Eq. (13), for the case of Mie (anisotropic) scattering. The scatterer is localized in space, similar to an idealized cloud in the atmosphere, and has steep spatial gradients. Corresponding numerical solution $I_3$ is shown in Figure 9.

Figure 9: Fast and low-memory numerical solution for $I_3$, a case with Mie (anisotropic) scattering. Top-left: Illustration of the mean intensity of solution $I_3$, and an example mesh from $hp$-AMR. Top-right: Relative $L^2$ error versus number of degrees of freedom in each spatial dimension, $n^{1/d}$, with $d = 2$ here. Bottom-left: Cost of the iterative solver versus total number of degrees of freedom, $n_M$. The inset shows the scaling as fit to a power law. Bottom-right: Memory reduction factor, $N/n$, versus number of degrees of freedom in each spatial dimension, $n^{1/d}$, with $d = 2$ here. Comparisons are shown for four different refinement strategies: $h$-refinement, $p$-refinement, $h$-AMR, and $hp$-AMR. The $hp$-AMR method is a fast and low-memory method for this example, as shown by the approximately linear scaling in cost (bottom-left) and large amount of memory reduction (bottom-right). The $p$-refinement method has a higher cost in this example, due to the presence of steep gradients in the solution.
Note that in Figure 9 the \( p \)-refinement method appears to have a lower error and larger memory reduction than the \( hp \)-AMR method, for a given \( n \). However, the \( p \)-refinement method has a higher cost (Figure 9, bottom-left). Consequently, in a comparison of error versus cost, in Figure 10, we observe that \( hp \)-AMR is most efficient.

Finally, we test the memory occupied by the assembled sparse matrix \( \mathbb{A} \) (see Eq. (9)) for the three test scenarios we considered previously, namely, for (1) the test using the smooth solution \( I_1 \), for (2) the test using the solution with steep gradients \( I_2 \), and for (3) the Mie scattering test with the solution \( I_3 \).

The top row of the Figure 11 shows how the memory occupation of the matrix \( \mathbb{A} \) increases as we increase the DOFs of the numerical solution. We observe that the high-order methods (\( hp \)-AMR and \( p \)-ref) use more memory to assemble the matrix \( \mathbb{A} \) compared to the low-order methods (\( h \)-AMR, \( h \)-ref), for a given number of solution DOFs. This result is expected since a higher polynomial degree \( p \) will lead to a larger non-sparsity pattern due to the assembly of the advection term \( \int_{K^*} \int_{K} (I_s \cdot \nabla v) \); we refer to the Appendix B for more details on this term. Between the high-\( p \) methods, the \( hp \)-AMR method uses less memory than \( p \)-ref in the two cases with steep gradients (\( I_2 \) and \( I_3 \)). Also, while this top row of panels is informative for comparisons at a fixed number of solution DOFs, it is also informative to compare the matrix \( \mathbb{A} \) memory that is used to achieve a certain accuracy or error.

In the bottom row of Figure 11, we show how fast the numerical errors decrease based on the memory occupation of the matrix \( \mathbb{A} \). We observe that our proposed \( hp \)-AMR is the most efficient method in reducing the corresponding errors for a given amount of matrix \( \mathbb{A} \) memory occupation, for test 2 and test 3. For test 1, it behaves closely to the \( p \)-refinement method and is much more efficient than the other methods (\( h \)-AMR and \( h \)-ref). These results further strengthen our conclusion that \( hp \)-AMR is the most memory-efficient and versatile method in reducing the numerical error.

5. Concluding Discussion

Given the large memory needed to represent the radiant intensity \( I(x, y, z, \theta, \phi) \), and the steep gradients that arise in applications such as medical imaging and atmospheric clouds, an \( hp \)-AMR strategy was investigated here as a candidate for a fast, low-memory method.

For all cases of numerical tests here—smooth or steep gradients, Rayleigh or Mie scattering—high-order methods are significantly more efficient than low-order methods. In other words, methods with high \( p \) substantially outperform methods with fixed smaller \( p \) such as \( h \)-AMR. Of the two high-order methods here, \( hp \)-AMR is more efficient at representing the solution than \( p \)-refinement at places where the solution has steep gradients. The use of \( hp \)-AMR for radiative transfer has received little attention, as described in the Introduction section. An important result here is the potential benefit of \( hp \)-AMR for helping to overcome the prohibitive memory expense that commonly occurs for radiative transfer.

Extensions of the present methods could provide even further speedup and memory reduction. For instance, note that the memory reduction was in the spatial dimensions here, as a first investigation as it already produces substantial savings. One could consider similar methods in the angular dimensions and possibly in the electromagnetic frequency dimension, which we leave as future work. Furthermore, the two-dimensional case is presented here since it allows a high-resolution solution to be computed as a
basis of comparison for assessing accuracy. The three dimensional case is expected to be amenable to a similar framework with appropriate modifications, and DG and spectral element methods in general are efficiently scalable in parallel computations [22, 69]. Also, note that the two-dimensional case is already potentially relevant for multi-scale modeling frameworks in climate modeling (sometimes called “super-parameterization”), for which the fine-scale models are typically two-dimensional atmospheric models [37, 80, 71, 40]. Furthermore, in two or three dimensions, the use of AMR for radiative transfer could also be useful for some recent efforts on atmospheric fluid dynamics simulations with mesh refinement in the vertical direction [68, 63].

Additional benefit could also come from theoretical advances for AMR methods, such as finding or proving reliable and efficient error estimators [94, 6], and refinement strategies which guarantee sufficient error reduction [18, 17, 16]. The theory for AMR is relatively well-developed for $h$-AMR and elliptic type problems. For $hp$-AMR and transport equations, theoretical works are less developed, and heuristic approaches are widely used and are based on theoretical ideas and show successful results but can benefit from advances in provable guarantees for designing error estimators and refinement strategies.

With the memory reduction achieved with $hp$-AMR, it is possible that the computational cost of 3D radiative transfer could be significantly more affordable. Many applications are possible for future work, including medical imaging, satellite remote sensing, and weather and climate prediction.

Declaration of competing interest

The authors declare no conflicts of interest.

Acknowledgements

Support for this research was provided by the Office of the Vice Chancellor for Research and Graduate Education at the University of Wisconsin-Madison with funding from the Wisconsin Alumni Research Foundation.

Appendix A. Low-memory representation by DG spectral element: Further details

Here we explain how the basis functions, the mass and the stiffness matrices are constructed. We begin by considering the one-dimensional basis functions. Let \( \{ h_m(x) \}_{m=1}^{p+1} \) \( (p \text{ as the polynomial degree}) \) be the spectral element basis on the reference element \([-1, 1]\). When \( x \notin [-1, 1] \), we let \( h_m(x) = 0 \) for convenience. Here we consider the nodal basis. Namely, we choose \( \{ h_m \}_{m=1}^{p+1} \) to be the Lagrange
polynomials associated with the Legendre–Gauss–Lobatto (LGL) points \( \{ \xi_i \}_{i=1}^{p+1} \) such that \( h_j(\xi_i) = \delta_{ij} \). We also let \( \{ w_i \}_{i=1}^{p+1} \) be the associated quadrature weights. Let \( F_{[a,b]} \) be the affine map from \([-1,1]\) to \([a,b]\). Then the basis on \([a,b]\) are constructed as follows:

\[
\phi^{[a,b]}_m(x) := h_m \circ F_{[a,b]}^{-1}(x).
\]

For any given function \( f(x) \) with sufficient regularity, we have

\[
\int_a^b f(x)\phi^{[a,b]}_m(x)dx = \frac{|b-a|}{2} \int_{-1}^1 f(F_{[a,b]}(\hat{x}))h_m(\hat{x})d\hat{x} \approx \frac{|b-a|}{2} w_m f(F_{[a,b]}(\xi_m)).
\]

We remark that the quadrature rule by \( \{ \xi_i, w_i \}_{i=1}^{p+1} \) is exact for function in \( P_{2p-1} \) [93]. If we replace \( f \) by \( g(\omega)^{[c,d]} \), then

\[
\int_a^b g(x)\phi^{[a,b]}_m(x)\phi^{[c,d]}_j(x)dx = \int_a^b g(x)h_m \circ F_{[a,b]}^{-1}(x)h_j \circ F_{[c,d]}^{-1}(x)dx = \frac{|b-a|}{2} \int_{-1}^1 h_m(\hat{x})(g h_j \circ F_{[c,d]}^{-1}) \circ F_{[a,b]}(\hat{x})d\hat{x} \\
\approx g \circ F_{[a,b]}(\xi_m) \frac{|b-a|}{2} w_m h_j \circ F_{[c,d]}^{-1} \circ F_{[a,b]}(\xi_m) := g \circ F_{[a,b]}(\xi_m) M_{m_j}^{[a,b][c,d]},
\]

where the matrix \( M_{m_j}^{[a,b][c,d]} \) represents the transfer of the degrees of freedoms (DOFs) from the element \([c,d]\) to \([a,b]\). Specifically, if \([a,b] = [c,d]\), then we write

\[
M_{m_j}^{[b-a]} := M_{m_j}^{[a,b][a,b]} = \frac{|b-a|}{2} w_m \delta_{jm}.
\]

Note that the above mass matrix is diagonal, allowing an efficient inversion.

Finally, we consider the integration of the following form as the matrix responsible for advection:

\[
\int_a^b \phi^{[a,b]}_m(x)\partial_x \phi^{[a,b]}_m(x)dx = \int_a^b \phi^{[a,b]}_j(x)\partial_x h_m \circ F_{[a,b]}^{-1}(x) \frac{2}{|b-a|} dx = \int_{-1}^1 h_j(\hat{x})\partial_x h_m(\hat{x})d\hat{x} \approx w_j \partial_x h_m(\xi_j) := S_{jm}.
\]

This form will appear in the discretization of the advection operator \( s \cdot \nabla I \) in the radiative transfer equation.

Appendix B. Discontinuous Galerkin: Further details

In this section, we rewrite the integral terms of Eq. (8) into matrix forms, which can be directly used for implementation. We will put an underline to these matrices for the ease of identification. For each spatial-angular element \( K \times K^a \in T_h \times T_h^{K^a} \), we have \( K \times K^a = [x^K, y^K] \times [y^K, y^K] \times [\theta^K, \theta^K] \). Thus, we can express the numerical solution \( I_h \) as follows:

\[
I_h = \sum_{K \in T_h} \sum_{K^a \in T_h^{K^a}} \sum_{m_{1,2,3}} I_{m_{1,2,3}}^{K \times K^a} \phi^{[x^K, x^K]}(x)\phi^{[y^K, y^K]}(y)\phi^{[\theta^K, \theta^K]}(\theta),
\]

where \( I_{m_{1,2,3}}^{K \times K^a} \) represents the DOFs of \( I_h \) on the element \( K \times K^a \). Let \( v = \phi^{[x^K, x^K]}(x)\phi^{[y^K, y^K]}(y)\phi^{[\theta^K, \theta^K]}(\theta) \) be the test function. Then, for the advection term, we have

\[
\int_{K \times K^a} \int K s \cdot \nabla v = \sum_{m_{1,2,3}} I_{m_{1,2,3}}^{K \times K^a} \int_{K^a} \int_{K} \phi^{[x^K, x^K]}(x)\phi^{[y^K, y^K]}(y)\phi^{[\theta^K, \theta^K]}(\theta) \\
\times \partial_x \phi^{[x^K, x^K]}(x)\phi^{[y^K, y^K]}(y)\phi^{[\theta^K, \theta^K]}(\theta) dxdy d\theta \\
\approx \sum_{m_{1,2,3}} I_{m_{1,2,3}}^{K \times K^a} \partial_x \phi^{[x^K, x^K]}(x)\phi^{[y^K, y^K]}(y)\phi^{[\theta^K, \theta^K]}(\theta) dxdy d\theta \\
\times \phi^{[x^K, x^K]}(x)\phi^{[y^K, y^K]}(y)\phi^{[\theta^K, \theta^K]}(\theta) \frac{1}{|K \times K^a|}(\xi_m) \\
\times S_{m_{1,2,3}} M_{m_{1,2,3}}^{[a,b][a,b]} \cos(F_{[a,b]}(\xi_m)) \\
+ M_{m_{1,2,3}}^{[a,b][a,b]} S_{m_{1,2,3}} \sin(F_{[a,b]}(\xi_m)) M_{m_{1,2,3}}^{[a,b][a,b]} \cos(F_{[a,b]}(\xi_m)),
\]

18
For the extinction term, we have
\[ \int \beta_e I_h v = \sum_{m_1, m_2, m_3} I_{m_1, m_2, m_3} \int_{K} \int_{K} \beta_e(x, y) \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y} (\theta) dx dy \]
\[ \approx \sum_{m_1, m_2, m_3} I_{m_1, m_2, m_3} \beta_e \cdot F_K(\xi_{m_1}, \xi_{m_2}) M^{x_1 \cdot x_2, \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y}} \cdot \frac{K^2}{4} p(F_K(\xi_{m_1}), F_K(\xi_{m_2})). \]

For the scattering term, we have
\[ \int \frac{\beta_e}{|S|} \int_{s} p(s, s') I_h(s') v ds' ds = \int \frac{\beta_e}{|S|} \int_{s} p(\theta, \theta') \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y} (\theta) \]
\[ \sum_{K' \in T_{h} K} \sum_{m_1, m_2, m_3} I_{m_1, m_2, m_3} \int_{s} \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y} (\theta) \int_{s} \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y} (\theta') d\theta d\theta' d\theta d\phi d\theta = 0, \]
where \( F_{br} \) is the restriction of \( I_h \) on \( \partial K \) from the neighbour elements of \( K \), and \( F_K \) represents the collection of the faces of \( K \). Let us next consider a specific case that \( F \) is the right face of the element \( K \), namely, \( F = \{ x_1^t \} \times [y_0^t, y_1^t] \). Then we can write the first term as follows:
\[ \int_{K \cap \{ s \in F \}} I_h v(s \cdot n) = \sum_{F \in F_K} \int_{F} \int_{F} \frac{\beta_e}{|S|} \int_{s} p(\theta, \theta') \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y} (\theta) \]
\[ \sum_{K' \in T_{h} K} \sum_{m_1, m_2, m_3} I_{m_1, m_2, m_3} \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y} (\theta) \Phi_{m_1, m_2, m_3} (\frac{\omega}{\theta}) \cos(\theta, \sin(\theta)) \cdot (1, 0) d\theta d\phi d\theta = 0. \]

where \( \mathbb{I} \) is the indicator function. For the second boundary integral term, we can proceed as follows:
\[ \int_{K \cap \{ s \in F \}} I_h v(s \cdot n) = \sum_{F \in F_K} \int_{F} I_{br} v(s \cdot n) \]
\[ = \sum_{K' \in T_{h} K} \sum_{m_1, m_2, m_3} I_{m_1, m_2, m_3} \frac{\beta_e}{|S|} \int_{s} p(\theta, \theta') \phi_{m_1}^{x, y} \phi_{m_2}^{x, y} \phi_{m_3}^{x, y} (\theta) \Phi_{m_1, m_2, m_3} (\frac{\omega}{\theta}) \cos(\theta, \sin(\theta)) \cdot (1, 0) d\theta d\phi d\theta = 0. \]
where $T_{nbr}^{K,F}$ represents the collection of neighbour elements of $K$ through the face $F$. Similar calculation can be performed for the other faces of $K$ (the top, left, and bottom face) and we will not repeat here. Finally for the forcing term, we have

$$
\int_{K} \int_{K} \tilde{B} v = \int_{K} \int_{K} \tilde{B}(x,y) \phi_{m_1}^{Kx} x_{i}^{Kx} (x) \phi_{m_2}^{Ky} y_{i}^{Ky} (y) \phi_{m_3}^{K\theta} \theta_{i}^{K\theta} (\theta) \, dx \, dy \, d\theta \\
\approx \frac{|K||K|}{8} w_{m_1} w_{m_2} w_{m_3} \tilde{B}(\tilde{F}_{i}^{x_{K}^{0}}, \tilde{F}_{i}^{y_{K}^{0}}, \tilde{F}_{i}^{\theta_{K}^{0}}) \phi_{m_1}^{x_{K}^{0}} x_{i}^{x_{K}^{0}} (\xi_{m_1}^{0}), \phi_{m_2}^{y_{K}^{0}} y_{i}^{y_{K}^{0}} (\xi_{m_2}^{0}), \phi_{m_3}^{\theta_{K}^{0}} \theta_{i}^{\theta_{K}^{0}} (\xi_{m_3}^{0})).
$$

The above discretized forcing term can be written into an array which has the same size of the free DOFs of the numerical solution $I_{h}$.

References


