BSc Thesis Applied Mathematics

Implementing boundary conditions with Fresnel laws for a finite element method for the Radiative Transfer Equation.

Pelle Steverink

Supervisor: M.Schlottbom, V.Bosboom
external examiner: M.Walter

July, 2023

Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.

Implementing boundary conditions with Fresnel laws for a finite element method for the Radiative Transfer Equation.

Pelle Steverink*

July, 2023

1 Introduction

When radiation travels through a medium, it gets scattered and moves in an erratic way. To model this movement of radiation, an equation called the Radiative Transfer Equation (RTE) is used. This equation is well defined. However, it is very computationally difficult to solve. To address this multiple methods of approximating the solution have been developed. One of those methods is the P_N method, which uses Legendre polynomials to approximate the solution to the RTE. When the radiation strikes a boundary between two mediums however, Snell's law and the law of reflection affect the radiation's path. We can include this phenomenon into the boundary conditions of the RTE. This paper will give a method of implementing such boundary conditions on the radiative transfer equation for a slab geometry.

2 The Radiative Transfer Equation

2.1 The equation

Radiation that passes through a medium can be modeled using the the following equation [3, 1]:

$$\mu \partial_z \phi(z,\mu) + \sigma_t \phi(z,\mu) = 2\pi\sigma_s \int_{-1}^{1} p(\mu,\mu')\phi(z,\mu')d\mu' + q(z,\mu). \tag{1}$$

In this equation, we have $0 \leq z \leq L$, the distance through a medium with length L, $-1 \leq \mu \leq 1$ the cosine of the angle that the radiation is flowing in with the normal vector to the medium, $\phi(z,\mu)$ indicates the intensity of radiation at a point z in the direction μ, σ_s is the scattering rate and $\sigma_t = \sigma_a + \sigma_s$ is the rate of absorption plus the scattering rate (which we will assume are all constant throughout z), and $q(z,\mu)$ is the external source term, which will be more explicitly stated in section 2.3. The integral in the equation represents radiation that scatters from other directions into the direction μ, and $p(\mu,\mu')$ is the probability that light at μ' scatters towards μ which we assume to behave like the Henyey-Greenstein phase function as it is in [1]:

$$p(\mu,\mu') = \sum_{l=0}^{\infty} \frac{1}{2\pi} g^l P_l(\mu) P_l(\mu'). \tag{2}$$

*Email: p.steverink@student.utwente.nl

Here we have the anisotropy g, which is a parameter we can manipulate, and $P_l(\mu)$ is the lth normalized Legendre polynomial. Legendre Polynomials are known functions with many useful properties that we will need to approximate the RTE quickly. An important property is that the specific polynomials we choose are orthonormal, meaning that $\int_{-1}^{1} P_l(\mu)P_k(\mu) = 1$ only if $l = k$ and 0 otherwise [7].

2.2 Boundary Conditions

Notice that the RTE does not yet have any boundary conditions. The boundary condition describe reflections at the boundary and will be taken from [2] to give the boundary value problem

$$
\begin{aligned}
&\mu\partial_z\phi(z,\mu) + \sigma_t\phi(z,\mu) = 2\pi\sigma_s\int_{-1}^{1} p(\mu,\mu')\phi(z,\mu')d\mu' + q(z,\mu) \\
&\phi(0,\mu) = R(\phi(0,\mu)), \qquad \mu > 0 \\
&\phi(Z,\mu) = R(\phi(Z,\mu)), \qquad \mu < 0.
\end{aligned} \tag{3}
$$

Here R is the reflection function. This is defined in such a way that the radiation that flows into a body is equal to the radiation that is inside the body already that is flowing to the left and reflected back into the body and the radiation that flows out of the body to the right is reflected back inside. This means that $R(\phi(z,\mu))$ looks as follows:

$$
\begin{aligned}
R(\phi(0,\mu)) &= r(-\mu)\phi(0,-\mu), \quad \mu > 0 \\
R(\phi(Z,\mu)) &= r(\mu)\phi(Z,\mu), \qquad \mu < 0
\end{aligned} \tag{4}
$$

where $\phi(z,-\mu)$ means that the light is redirected in the opposite direction, and $r(\mu)$ is a function based on the Fresnel laws [5], which takes the following form:

$$
r(\mu) = \begin{cases} \frac{1}{2}((\frac{n_j\eta(\mu)-n_i|\mu|}{n_j\eta(\mu)+n_i|\mu|})^2 + (\frac{n_i\eta(\mu)-n_j|\mu|}{n_i\eta(\mu)+n_j|\mu|})^2), & \mu \in [-1,\mu_{lim}] \\ 1, & \mu \in (\mu_{lim},0], \end{cases} \tag{5}
$$

$$
\eta(\mu) = \sqrt{1 - \frac{n_i^2}{n_j^2}(1-\mu^2)}, \tag{6}
$$

with $\mu_{lim} = \sqrt{1 - \frac{n_j^2}{n_i^2}}$.

Here n_i and n_j are the refractive indices of the medium and the outside respectively.

2.3 Normalization

We want to check whether the model from (3) reproduces the results in [1]. This paper writes the RTE in terms of the albedo, anisotropy, refractive index difference, and optical thickness. In order to reproduce those results, we first have to rewrite the RTE to find these easily manipulated parameters. We represent the albedo with a, where $\sigma_s = \frac{a*b}{L}$, the anisotropy with g (as seen in equation (2)), and the optical thickness with $b = \sigma_t * L$. Now we take the RTE in the same form as and with the source function as [1],

$$
\mu\partial_z\phi(z,\mu) + \sigma_t\phi(z,\mu) = 2\pi\sigma_s\int_{-1}^{1} p(\mu,\mu')\phi(z,\mu')d\mu' + q(z,\mu), \tag{7}
$$

with $0 \leq z \leq L$, and we normalize the coordinate z to $z = \frac{\tilde{z}}{L}$, which changes the equation to

$$\frac{1}{L}\mu\partial_{\tilde{z}}\phi(\tilde{z},\mu) + \sigma_t\phi(\tilde{z},\mu) = \frac{\sigma_s}{2}\int_{-1}^{1} p(\mu,\mu')\phi(\tilde{z},\mu')d\mu' + q(\tilde{z},\mu).$$

Now it's finally time to take a look at the source term taken from [1], which is defined to be plane waves expanded into normalized Legendre polynomials to get

$$q(\tilde{z},\mu) = \sum_{k=0}^{\infty}\sqrt{\frac{2k+1}{2}}P_k(\mu)F_0 abg^k e^{-b\tilde{z}}. \tag{8}$$

Here F_0 is the incident flux, which we will assume to be 1. Now we multiply the RTE with L, together with $b = \sigma_t * L$, $\sigma_s = \frac{a*b}{L}$, and the source term we get the normalized equation

$$\mu\partial_{\tilde{z}}\phi(\tilde{z},\mu) + b\phi(\tilde{z},\mu) = ab2\pi\int_{-1}^{1} p(\mu,\mu')\phi(\tilde{z},\mu')d\mu' + \sum_{k=0}^{\infty}\sqrt{\frac{2k+1}{2}}P_k(\mu)F_0 abg^k e^{-b\tilde{z}}. \tag{9}$$

3 Weak formulation

3.1 Even-odd splitting

Now that we have established our model, we need to find a way to solve it. We do this by splitting the RTE (equation (9)) and the boundary conditions (equation (3)) into the even and odd parts of the solution $\phi(\tilde{z},\mu)$ in μ. As a reminder, the definition of an even function is a function $f(x)$ that satisfies $f(x) = f(-x)$, and an odd function is a function $g(x)$ where $g(x) = -g(-x)$ for all x in the domain of those functions. Also, any function $h(x)$ can be decomposed into an even and odd part

$$h(x) = h^+(x) + h^-(x), \tag{10}$$

where $h^+(x)$ is an even function and $h^-(x)$ is an odd function which are defined as

$$\begin{aligned} h^+(x) &= \frac{h(x)+h(-x)}{2} \\ h^-(x) &= \frac{h(x)-h(-x)}{2}. \end{aligned} \tag{11}$$

We perform this splitting so that we can use the symmetry properties of even and odd function to simplify the weak formulation. For example, we have for any arbitrary odd function $\psi^-(x)$ that

$$\int_{-1}^{1}\psi^-(x)\mathrm{d}x = 0. \tag{12}$$

3.2 Weak formulation

To obtain the weak formulation we start by multiplying the RTE with an arbitrary even test function ψ^+ and integrating the resulting product both with respect to z and μ. This gives us the equation

$$(\mu\partial_{\tilde{z}}\phi^-, \psi^+) + ((b - ab2\pi\mathcal{P}^+)\phi^+, \psi^+) = (q^+, \psi^+), \tag{13}$$

where

$$
\begin{aligned}
(\phi, \psi) &= \int_{-1}^{1}\int_{0}^{1} \phi(\tilde{z},\mu)\psi(\tilde{z},\mu)\mathrm{d}z\mathrm{d}\mu, \\
\mathcal{P}^{+}(\phi(\tilde{z},\mu)) &= \int_{-1}^{1} p(\mu,\mu')^{+}\phi(\tilde{z},\mu')d\mu',
\end{aligned} \tag{14}
$$

and

$$
p^{+}(\mu,\mu') = \sum_{l=0,2,4,6\ldots}^{\infty} \frac{1}{2\pi} g^{l} P_l(\mu)P_l(\mu'). \tag{15}
$$

We take the first term of this equation and apply integration by parts,

$$
(\mu\partial_{\tilde{z}}\phi^{-},\psi^{+}) = -(\phi^{-},\mu\partial_{\tilde{z}}\psi^{+}) + \langle\phi^{-},\psi^{+}\rangle_{L_2^{+}} - \langle\phi^{-},\psi^{+}\rangle_{L_2^{-}},
$$

where

$$
\begin{aligned}
\langle\phi,\psi\rangle_{L_2^{-}} &= \int_{0}^{1} |\phi(0,\mu)\psi(0,\mu)||\mu|d\mu + \int_{-1}^{0} |\phi(1,\mu)\psi(1,\mu)||\mu|d\mu, \\
\langle\phi,\psi\rangle_{L_2^{+}} &= \int_{0}^{1} |\phi(1,\mu)\psi(1,\mu)||\mu|d\mu + \int_{-1}^{0} |\phi(0,\mu)\psi(0,\mu)||\mu|d\mu.
\end{aligned} \tag{16}
$$

By the symmetry of the boundaries $\langle\phi^{-},\psi^{+}\rangle_{L_2^{+}} = -\langle\phi^{-},\psi^{+}\rangle_{L_2^{-}}$, we get

$$
(\mu\partial_{\tilde{z}}\phi^{-},\psi^{+}) = -(\phi^{-},\mu\partial_{\tilde{z}}\psi^{+}) - 2\langle\phi^{-},\psi^{+}\rangle_{L_2^{-}}.
$$

We do this in order to include the boundaries conditions into equation (13) and to obtain an equation that does not depend on ϕ^{-}.

Now we need to find a way to express $\langle\phi^{-},\psi^{+}\rangle_{L_2^{-}}$ in a way that only depends on ϕ^{+} When we split the boundary conditions up into two parts, we want to be able to find an equation that only depends on ϕ^{+}, that we can use in the method of [3]. First we take the boundary condition from formula (4) and split it up into two parts, using the definition of even-odd decomposition: (For simplicity, we perform this calculation at the boundary $(0,\mu)$, but at $(1,\mu)$ the procedure is the same)

$$
\phi^{+}(0,\mu) = \frac{1}{2}(\phi(0,\mu) + \phi(0,-\mu)).
$$

Rewriting this to express $\phi(\tilde{z},\mu)$ gives:

$$
\phi(0,\mu) = 2\phi^{+}(0,\mu) - \phi(0,-\mu).
$$

Now we can replace $\phi(\tilde{z},\mu)$ with the equation for (4), which gives:

$$
2\phi^{+}(0,\mu) - \phi(0,-\mu) = r(-\mu)\phi(0,-\mu).
$$

Rewriting this gives:

$$
\phi^{+}(0,\mu) = \frac{1}{2}(1 + r(-\mu))\phi(0,-\mu).
$$

Now we do the same process but with an odd function, resulting in

$$\phi^-(0,\mu) = \frac{1}{2}(r(-\mu)-1)\phi(0,-\mu).$$

Lastly, we can combine these two expressions of even and odd functions to get an expression to replace ϕ^- with ϕ^+,

$$\frac{r(-\mu)-1}{1+r(-\mu)}\phi^+(0,\mu) = \phi^-(0,\mu). \tag{17}$$

Now, using the even-odd decomposition of the boundary equations we can convert the integral of the boundary part into

$$(\mu\partial_{\tilde{z}}\phi^-,\psi^+) = -(\phi^-,\mu\partial_{\tilde{z}}\psi^+) - 2\langle A\phi^+,\psi^+\rangle_{L_2^-},$$

with

$$A(\mu) = \frac{r(-|\mu|)-1}{1+r(-|\mu|)}. \tag{18}$$

(The absolute value appears because of the boundary at $(1,\mu)$)Entering this into (13) gives

$$-2\langle A\phi^+,\psi^+\rangle_{L_2^-} - (\phi^-,\mu\partial_{\tilde{z}}\psi^+) + ((b-ab2\pi\mathcal{P}^+)\phi^+,\psi^+) = (q^+,\psi^-). \tag{19}$$

We can also test equation (9) with an odd function, which gives the following expression for ϕ^-

$$(\mu\partial_{\tilde{z}}\phi^+,\psi^-) + ((b-ab2\pi\mathcal{P}^-)\phi^-,\psi^-) = (q^-,\psi^-) \tag{20}$$

where

$$\mathcal{P}^-(\phi(\tilde{z},\mu)) = \int_{-1}^{1} p(\mu,\mu')^-\phi(\tilde{z},\mu')d\mu' \tag{21}$$

and

$$p^-(\mu,\mu') = \sum_{l=1,3,5...}^{\infty} \frac{1}{2\pi} g^l P_l(\mu)P_l(\mu'). \tag{22}$$

Equation (20) can be solved to

$$\phi^- = (b-ab2\pi P^-)^{-1}(q^- - \mu\partial_{\tilde{z}}\phi^+), \tag{23}$$

which we can enter into the equation to get an equation that is only dependent on ϕ^+:

$$\begin{aligned} &-2\langle A\phi^+,\psi^+\rangle_{L_2^-} - (-(b-ab2\pi P^-)^{-1}\mu\partial_{\tilde{z}}\phi^+,\mu\partial_{\tilde{z}}\psi^+) + ((b-ab2\pi\mathcal{P}^+)\phi^+,\psi^+) = \\ &(q^+,\psi^+) + ((b-ab2\pi P^-)^{-1}q^-,\mu\partial_{\tilde{z}}\psi^+). \end{aligned} \tag{24}$$

4 Discretization

4.1 Discretizing the weak formulation

In order to be able to solve the weak formulation that we have established in the previous section in an efficient manner, we perform the following steps. We approximate function in a basis of N Legendre polynomials and n_z hat functions, and then apply a Gaussian quadrature rule to solve the resulting integrals. For the sake of brevity, we take for each of these two methods one example each, and note here that the other parts of the weak formulation can be done in a similar way. Let us start with the expansion into Legendre polynomials and hat functions, and take as an example the total scattering term from equation (24),

$$(b\phi^+, \psi^+) = \int_0^1 \int_{-1}^1 b\phi^+(z,\mu)\psi^+(z,\mu)\mathrm{d}\mu\mathrm{d}z. \tag{25}$$

Notice how we were working with an arbitrary function $\psi^+(z,\mu)$ until now. We now take this function to be more specifically

$$\psi^+(z,\mu) = \gamma_j(z)P_l(\mu), \tag{26}$$

where $P_l(\mu)$ is the lth Legendre polynomial, and $\gamma_j(z)$ is a hat function with the maximum at $\frac{j-1}{n}$. A hat function is defined to be a piecewise linear function of the form

$$\gamma_j(z) = \begin{cases} \frac{z-(j-2)n}{n}, & \frac{j-1}{n} \leq z \leq \frac{j}{n} \\ 1 - \frac{z-(j-1)n}{n}, & \frac{j}{n} \leq z \leq \frac{j+1}{n} \\ 0, & \text{otherwise}, \end{cases} \tag{27}$$

with precision n (notice that this definition is not unique, but has two indices j and l) [4]. We also expand the function $\phi^+(z,\mu)$ to

$$\phi^+(z,\mu) \approx \sum_{i=1}^{n_z} \sum_{l=0}^{N-1} u_{il}\gamma_i(z)P_l(\mu). \tag{28}$$

Here $\gamma_i(z)$ is again a hat function with the maximum at $\frac{i}{n_z}$, $P_l(\mu)$ is the lth Legendre polynomial, and u_{il} are the weights of the function. We will be looking for a solution for u_{il} This expansion is a known property of Legendre polynomials. We can now rewrite equation (25) to be

$$(b\phi^+, \psi^+) \approx \int_0^1 \int_{-1}^1 b \sum_{i=1}^{n_z} \sum_{l=0}^{N-1} u_{il}\gamma_i(z)P_l(\mu)\gamma_j(z)P_l(\mu)\mathrm{d}\mu\mathrm{d}z.$$

We can rewrite this to group the terms dependent to z and terms dependent on μ, giving

$$\sum_{i=1}^{n_z} \sum_{l=0}^{N-1} u_{il} \int_0^1 \gamma_i(z)\gamma_j(z)b\mathrm{d}z \int_{-1}^1 P_l(\mu)P_k(\mu)\mathrm{d}\mu. \tag{29}$$

These two integrals are easily solved, since the Legendre polynomials are orthonormal and the hat functions are piecewise linear. We can also see that the equation looks like the definition of matrix multiplication, which means we can rewrite the equation using a vector u of length il, contain the coefficients u_il, we take multiple test functions ψ^+_{jl} and the discretization of ϕ^+ in a Kronecker product of the form

$$C_{il,jk} = A_{ij} \otimes B_{lk} \tag{30}$$

where

$$\begin{aligned} A_{ij} &= \int_0^1 \gamma_i(z)\gamma_j(z) b \mathrm{d}z \\ B_{lk} &= \int_{-1}^1 P_l(\mu)P_k(\mu)\mathrm{d}\mu. \end{aligned} \tag{31}$$

We can do a similar discretization for all parts of the weak formulation (24), resulting in

$$(C + D + B + A)u = Q + F, \tag{32}$$

where C is the discretization we just calculated, D, B and A are the discretizations of the other terms on the left side of the weak formulation (formula (24)), and Q and F are two vectors which result from discretizing the right side of the weak formulation, which we will discuss more in section 4.2. Now that we have the weak formulation in a form of square matrices, we can solve this as

$$u = (C + D + B + A)^{-1}(Q + F), \tag{33}$$

which is a matrix vector problem quickly solved by a computer.

4.2 Gauss-Legendre quadrature

To construct the matrices in equation (32) we need to evaluate integrals. However, not all of these integrals can easily be evaluated by hand. We take as an example the source term from the weak formulation:

$$(q, \psi^+) = \int_{-1}^1 \int_0^1 q(\tilde{z}, \mu)\psi^+(\tilde{z}, \mu) d\tilde{z} d\mu = \int_{-1}^1 \int_0^1 \sum_{k=0}^{\infty} \sqrt{\frac{2k+1}{2}} P_k(\mu) F_0 a b g^k e^{-b\tilde{z}} \psi^+(\tilde{z}, \mu) d\tilde{z} d\mu. \tag{34}$$

We choose, just like before, multiple test functions and put them in matrix form. These test functions look as follows:

$$\psi^+(z, \mu)_{il} = \gamma_i(z)P_l(\mu), \tag{35}$$

where $\gamma_i(z)$ are hat functions with their maximum at $\frac{i}{n}$, with n is the desired accuracy for the weak formulation, and $P_l(\mu)$ is a Legendre polynomial of degree l. Now we rewrite the integral to

$$Q_{il} = \sum_{k=0}^{\infty} \sqrt{\frac{2k+1}{2}} \int_0^1 \gamma_i(z) F_0 a b g^k e^{-bz} dz \int_{-1}^1 P_k(\mu)P_l(\mu) d\mu \tag{36}$$

We can now split the equation into a kronecker product from two matrices

$$Q_{il} = \sqrt{\frac{2k+1}{2}} Q_z \otimes Q_\mu \tag{37}$$

Here we now notice that the Legendre polynomials $P_k(\mu)$ and $P_l(\mu)$ are orthonormal, similar to the above part, which simplifies the system from a matrix to a vector of the form

$$Q_{il} = \sqrt{\frac{2l+1}{2}} F_0 abg^k \int_0^1 e^{-bz} \gamma_i(z) dz. \tag{38}$$

The integral here can not be so easily calculated. Instead we approximate it using a Gauss-Legendre quadrature rule, which states

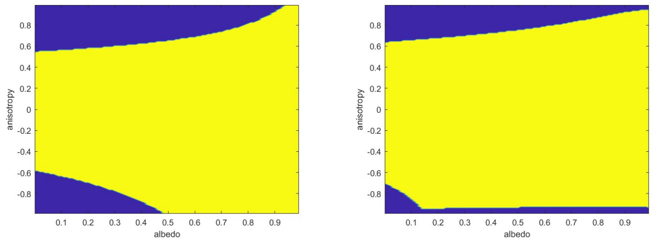
$$\int_0^1 e^{-bz} \gamma_i(z) dz \approx \sum_{j=1}^{n} e^{-bz_j} \gamma_i(z_j) w_j. \tag{39}$$

Here z_j is the root of the nth Legendre polynomial, and w_j are the weights of the quadrature, which are known and given by the formula [6]

$$w_j = \frac{2}{(1 - z_j^2)(P_n'(z_j))^2}. \tag{40}$$

5 Results

We implemented this model in Matlab, editing code that was provided by MACS. Our goal here is to reproduce and improve upon the results presented in [1]. We first take the albedo and anisotropy as variables, taking the albedo $a \in (0, 1)$ and anisotropy $g \in (-1, 1)$. Notice that these are open sets, which is needed since the perfect absorption and scattering cases are not physical [1]. Varying the albedo and anisotropy The model gives an unphysical result when either the intensity of the radiation is negative at any point in the medium, the radiation is flowing to the left on the left side of the medium, or the light is flowing to the right on the right side of the medium. We can obtain the results shown in figure 1, and we can compare them to figure 2 from [1] to see whether or not the method works.



(A) P_1 (B) P_3

Figure 1: Results for the model when varying the albedo and anisotropy. Here $b = 3$, $n_i = 1.4$ and $n_j = 1$.

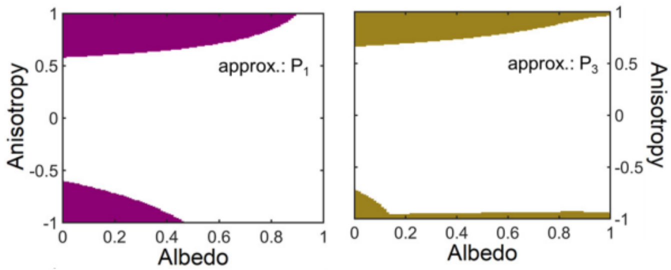
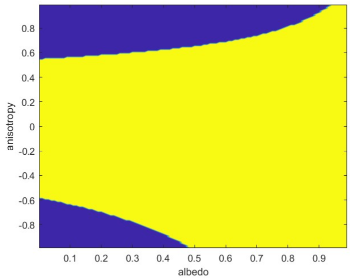


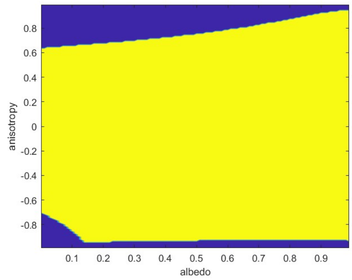
Figure 2: Values of the albedo and anisotropy varied in the model of [1], this uses the same values for the optical thickness and refractive indices as 1.

In figure 1 the regions that give an unphysical result are marked blue. We can see that the figures 1 and 2 are very similar to one another, which means that the matlab code works appropriately.

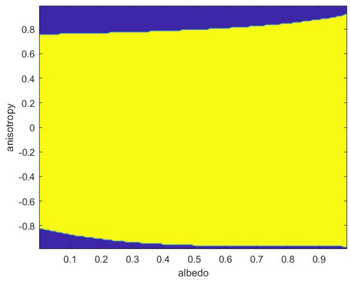
Our method can very easily be extended to moments higher than 3. These higher moments take very little time to compute, whereas the authors of [1] limit their equation to 3 moments. We can also confirm with these higher moments that the approximation from [3] becomes more accurate with more moments:



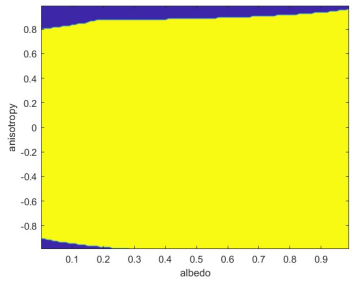
(A) $P_1, C \approx 0.788$



(B) $P_3, C \approx 0.845$



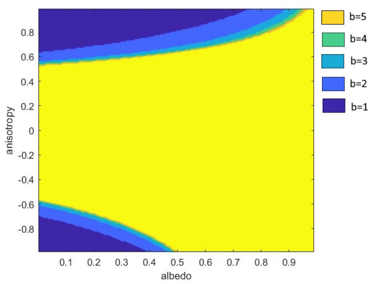
(C) $P_5, C \approx 0.878$



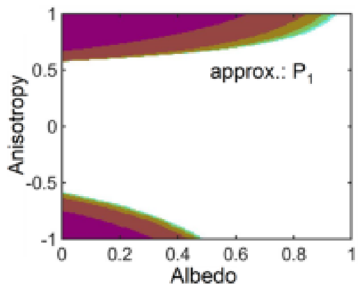
(D) $P_{13}, C \approx 0.936$

FIGURE 3: The unphysical ranges of the depicted P_N approximation. Here all parameters are the same as figure 1. The value C gives a value for the ratio of areas that are physically feasible.

So far we only varied the albedo and anisotropy. We can also investigate the influence of the optical thickness and the refractive indices. We first check to see if we can get a similar result as in [1]. We do this by making a P_1 approximation and running the matlab program for the optical thickness $b \in \{1, 2, 3, 4, 5\}$.



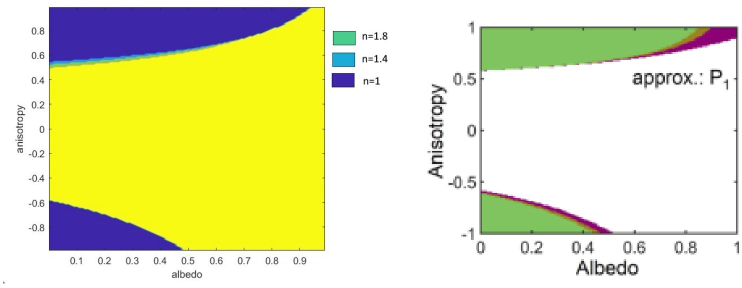
(A) Unphysical ranges of the P_1 approximation with refractive index $n_j = 1.4$



(B) Results from [1] for similar refractive index $n_j = 1.4$

FIGURE 4: comparing the variation of the models when varying the optical thickness

We can indeed see that the figures 4a and 4b line up. We can also do a similar process for the refractive indices, where we take $n_i \in \{1, 1.4, 1.8\}$.

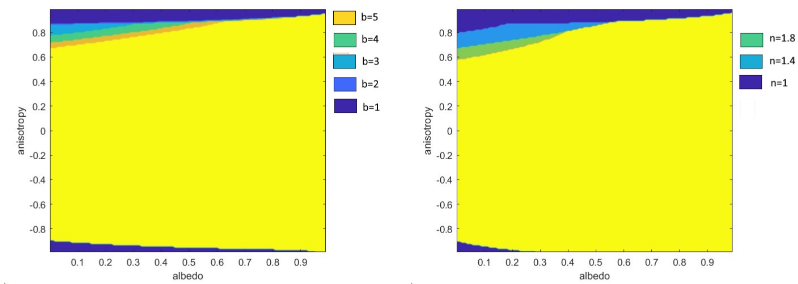


(A) The results from our model

(B) The result from [1]

FIGURE 5: Comparing the unphysical results when varying the refractive index for opticall thickness $b = 3$

In this situation we see a notable difference between the resulting graph from our Matlab code and the result from [1]. An explanation could be that the boundary conditions in [1] are not explicitly given, so there is a strong chance we have set up a different boundary conditions from that paper.
Instead of just recreating the results from [1], we can now use the quicker computation time to do this variation for P_N approximations for arbitrarily high n. Let us look at P_{13}, and see what influence varying the optical thickness and the refractive indices has.



(A) Varying the optical thickness for the P_{13} method with refractive index $n_i = 1.4$

(B) Varying the refractive index for the P_{13} method with optical thickness $b = 3$

We see here that with higher moments, similarly to when we did the P_1 method, that higher values for both the optical thickness and the difference between the two refractive indices result in more unphysical results. We should also note that these unphysical results vary more when we have a high anisotropy and low albedo, whereas the unphysical range for low anisotropy almost does not change.

6 Conclusion

In this report, we have derived and solved a model for the transfer of radiation through a dense medium with reflection at the boundaries. This model gives results similar to those in [1], and can be extended to arbitrarily high moments. In this model the results for varying the refractive index differs from that in [1], an explanation of this could be that the boundary conditions from [1] are not explicitly given, and that we have implemented different ones. Further research could include using this model for research regarding light transport, using the access to higher moments to get more accurate results for the RTE.

References

[1] O. Akdemir, A. Lagendijk, and W. L. Vos. Breakdown of light transport models in photonic scattering slabs with strong absorption and anisotropy. *Optica Advanced Photonics Congress 2022*, Mar 2022.

[2] A. A. Amosov. Radiative transfer equation with fresnel reflection and refraction conditions in a system of bodies with piecewise smooth boundaries. *Journal of Mathematical Sciences*, 219(6):821–849, 2016.

[3] Olena Palii and Matthias Schlottbom. On a convergent dsa preconditioned source iteration for a dgfem method for radiative transfer. *Computers & Mathematics with Applications*, 79(12):3366–3377, 2020.

[4] Wikipedia contributors. Triangular function — Wikipedia, the free encyclopedia, 2022. [Online; accessed 26-June-2023].

[5] Wikipedia contributors. Fresnel equations — Wikipedia, the free encyclopedia, 2023. [Online; accessed 13-June-2023].

[6] Wikipedia contributors. Gauss–legendre quadrature — Wikipedia, the free encyclopedia, 2023. [Online; accessed 24-June-2023].

[7] Wikipedia contributors. Legendre polynomials — Wikipedia, the free encyclopedia, 2023. [Online; accessed 31-May-2023].

7 Appendix

In this section we present the matlab code used to solve our model. This code is mostly based on code given by MACS and the sections that have been edited most heavily are highlighted with comments.

```
%intialization
ni=1.4; %refraction indeces
nj=1;
Z = 1; %End of domain

tol=1e-10; %error tolerance
dx = 1/256; %number of spatial grid points
N = 1; %number of moments
%information on position
x=0:dx:1; %interval [0,1]
nx=length(x);
xm=(x(1:end-1)+x(2:end))/2; %midpoints
PML=false; %perfectly matched layer

%Varying albedo and anisotropy
alb= 0.0000001:0.02:1;
anis=-0.99:0.02:0.99;
result= ones(length(alb),length(anis));
for i=1:length(alb)
    for j=1:length(anis)
        g=anis(j); % only, ie. isotropic scattering
        b=3; %optical thickness
        a=alb(i); %albedo

        [phie,phio,iter]=solveRTE_PN_mixed(N,x,a,ni,nj,g,b,PML
           ); %solve RTE with PN method
        avgu= 0.5.*sqrt(2).*phie(1:length(x)); %average
           intensity
        check = sum(avgu<0) + (phio(1)>0) + (phio(length(xm))
           <0);%check for unphysical results

        if check >0
             result(i,j)=false;
        end
    end
end
%plot results
[X,Y]=meshgrid(anis,alb);
contourf(Y,X,result,'LineColor', 'none')
```

```
function [phie,phio,iter,timing]=solveRTE_PN_mixed(N,x,a,ni,nj
    ,g,b,PML)

n=length(x);
eM=0:2:N-1; oM=1:2:N; %even and odd moments
NeM=length(eM); NoM=length(oM);
xm=(x(1:end-1)+x(2:end))/2;
dx=diff(x);

% assemble transport
A=kron(assemble_A_PN(N),assemble_dz(n));

% assemble mass matrix for pcw. linear functions
bf= @(x)b+0.*x;
Mte=assemble_M(x,bf);

% assemble mass matrices for pcw constants
Mto=diag(sparse(dx.*bf(xm)));

% assemble absorption and scattering parts
Ce=kron(speye(NeM),Mte)-a.*kron(diag(sparse(g.^eM)),Mte);
Co=kron(speye(NoM),Mto)-a.*kron(diag(sparse(g.^oM)),Mto);

% assemble boundary functional THIS SECTION HAS BEEN EDITED
bound=zeros(n,1); bound(1)=1; bound(end)=1;
etaij= @(mu, refindi, refindj) sqrt(1-(((refindi.^2)./(refindj
    .^2)).*(1-mu.^2)));
mulim=@(refindi,refindj) -sqrt(1-(refindj.^2)./(refindi.^2));
refr= @(mu, refindi, refindj) .5*(((refindj.*etaij(mu,refindi,
   refindj)-refindi.*abs(mu))...
     ./(refindj.*etaij(mu,refindi,refindj)+refindi.*abs(mu)))
        .^2+((refindi.*etaij(mu,refindi,refindj)-refindj.*abs(
        mu))...
     ./(refindi.*etaij(mu,refindi,refindj)+refindj.*abs(mu)))
        .^2).*(mu<=mulim(refindi,refindj))+1.*(mu>mulim(refindi
        ,refindj));
if PML==true
     Bmu=speye(NeM);
else

     [muip,wmuip]=lgwt(8.*max(N,2),0,1); % integrates
        Polynomials of order 2N-1 correctly, the max is only
        for convenience
     refrvr = (refr(-muip, ni,nj)-1)./(1+refr(-muip,ni,nj));
     Hlip=leg(N,muip)*spdiags(sqrt((2.*(0:N)'+1)/2),0,N+1,N+1);
         %evaluates normalized legendre polynomials at
        integration points
     Hlip_even=Hlip(:,1:2:N); %get only the even evaluated
        moments
```

```
    Bmu=-2*Hlip_even'*(diag(muip.*refrvr.*wmuip))*Hlip_even; %
       calculate boundary integral using Gaussian quadrature
end
B=kron(Bmu,diag(sparse(bound)));

%Source term ......
Qs = @(z) exp(-b.*z);

qex = zeros(length(x),1);
qox = zeros(length(xm),1);
qeN = (sqrt((2*eM+1)/2).*g.^(eM))';
qoN = (sqrt((2*oM+1)/2).*g.^(oM))';

[xip, wip] = lgwt(16,0,dx(1));

topology = 2;
elmat = zeros(length(x)-1,topology);
for i=1:length(x)-1
    for j=1:topology
        elmat(i,j) = i+j-1;
    end
end

for i=1:length(x)-1
    qex(elmat(i,1)) = qex(elmat(i,1))+(Qs((i-1)*dx(1)+xip).*
       wip)'*((dx(1)-xip)./dx(1));
    qex(elmat(i,2)) = qex(elmat(i,2))+(Qs((i-1)*dx(1)+xip).*
       wip)'*((xip)./dx(1));
   qox(i) = Qs((i-1)*dx(1)+xip)'*wip;
end

qe = kron(a.*b.*qeN,qex);
qo = kron(a.*b.*qoN,qox);
% MOST SIGNIFICANT EDITS END HERE

% solve system via even-parity
tic
phie=(Ce+B+A'*(Co\A))\(qe+A'*(Co\qo)); iter=1;
phio=Co\(qo-A*phie);
timing=toc;
end
```