

Simulador computacional para problemas estacionários de transporte de partículas neutras

Matheus Ribeiro Alves¹; Odair Pinheiro da Silva^{2*}

¹ Aluno do Instituto Federal Fluminense; ² Professor do Instituto Federal Fluminense

*odair.silva@iff.edu.br

Resumo

O nêutron é uma partícula eletricamente neutra e essa característica o permite, mesmo com baixíssima energia cinética, penetrar num átomo e entrar em seu núcleo. A modelagem física do fenômeno de transporte de nêutrons é constituída da migração destas partículas no interior de um meio material com a probabilidade de interação com os núcleos dos átomos deste meio. Para fazermos a modelagem matemática, considerando o transporte de partículas neutras, utilizamos a equação linearizada de Boltzmann (DUDERSTADT; MARTIN, 1979). Esta equação de transporte representa um balanço entre produção e perda destas partículas. Por fim, a modelagem computacional se apresenta como ferramenta indispensável na investigação científica, visto que possibilita a realização de experimentos simulados, com segurança, de modo eficiente e com baixo custo. Durante o desenvolvimento da programação do software, foram testadas diferentes linguagens e bibliotecas. Primeiramente, levando em conta o conhecimento dos envolvidos, foram utilizadas/avaliadas as linguagens: JAVA e C#, e posteriormente, com o intuito de construir um software mais eficiente, adotou-se uma linguagem de programação chamada C, para a implementação dos métodos numéricos. Até este ponto, temos implementado os métodos Diamod Difference (DD) e Degrau (Dg). A versão final deste simulador tende a possuir uma interface visual em JAVA FX, biblioteca da linguagem JAVA, integrada ao código escrito em C, utilizando as bibliotecas OpenBLAS e OpenMP. No contexto de um fenômeno físico simples, este projeto envolve três aspectos interdisciplinares: Física Nuclear, Matemática Aplicada e Ciência da Computação.

Palavras-Chave: Modelagem Computacional. Matemática Aplicada. Transporte de Partículas. Ordenadas Discretas

Introdução

Desde a descoberta do nêutron em 1932, pelo físico inglês James Chadwick, devido a sua propriedade de não possuir carga elétrica, inúmeras aplicações para os nêutrons vêm sendo desenvolvidas. Dentre estas aplicações podemos citar: física médica (Boron Neutron Capture Therapy - BNCT), ensaios não destrutivos (NEUTRONGRAFIA), perfilagem de petróleo e gás natural (OIL WELL LOGGING) e produção de energia elétrica (USINAS NUCLEARES).

Neste contexto, uma descrição precisa da migração dos nêutrons no interior de um meio material é necessária. Para fazer a modelagem matemática deste problema, utilizamos a equação linearizada de Boltzmann, que na forma mais geral,

apresenta grande dificuldade ou até impossibilidade de obtenção de solução analítica. Este fato incentiva o estudo de métodos numéricos que forneçam uma solução, ainda que aproximada, para esta equação.

É comum classificar esses métodos em determinísticos e estocásticos, e.g., métodos de ordenadas discretas (SN) e métodos de Monte Carlo, respectivamente (LEWIS; MILLER, 1993) (BRIESMEISTER, 2000) (WAGNER et al., 2011). A implementação computacional desses métodos numéricos nos permite explorar diversos cenários de simulação. Isto, possibilita ações preventivas e de controle no contexto de blindagem de radiação.

Metodologia ou Materiais e Métodos

A modelagem matemática do problema de transporte de partículas neutras será feita utilizando a equação linearizada de Boltzmann. Esta equação de transporte representa um balanço entre produção e perda destas partículas que, em sua generalidade, é uma equação integro-diferencial parcial de primeira ordem dependente de sete variáveis independentes: três espaciais, duas angulares, uma da energia e uma variável temporal.

O tratamento analítico desta equação é muito complexo; portanto, métodos numéricos são desenvolvidos no intuito de se obterem soluções aproximadas para o problema. Neste projeto, visando à modelagem computacional do transporte de partículas neutras, utilizaremos o método S_N , que consiste em discretizar as variáveis angulares em N direções (ordenadas discretas) e em utilizar um conjunto de quadraturas angulares para a aproximação dos termos integrais de fonte (LEWIS; MILLER, 1993). Ademais, vamos considerar a equação linearizada de Boltzmann simplificada, seguindo um modelo estacionário, monoenergético, em geometria unidimensional cartesiana e com espalhamento isotrópico, que se apresenta como

$$\mu_m \psi_m(x) + \Sigma_t(x) \psi_m(x) = \frac{1}{2} \Sigma_s(x) \sum_{n=1}^N \psi_n(x) \omega_n + Q(x). \quad (1)$$

Para implementação computacional, utilizamos um método numérico generalizado de malha fina, que apresenta parâmetros nas N equações auxiliares de balanço espacial que indicarão o método numérico pelo qual o usuário poderá optar: método DD, método Degrau ou método degrau característico. Também será implementado um método de malha grossa, o método MR (SILVA, 2018), que é completamente livre de erros de truncamento espacial e pode ser caracterizado em três passos essenciais:

- Primeiro passo é construir, em cada nodo de discretização, um conjunto completo de soluções elementares para as equações SN. Caso as seções de choque macroscópicas de dois ou mais nodos sejam as mesmas, usamos a mesma expressão da solução geral local.

- Segundo passo é usar a fonte interior e os fluxos incidentes nas fronteiras de cada nodo na solução geral local obtida no passo anterior e determinar matrizes de avanço para cada nodo da grade de discretização espacial.

- Terceiro e último passo é implementar o esquema iterativo NBI (one-node block inversion) com as matrizes de avanço obtidas no segundo passo, que, com as mais recentes estimativas dos fluxos incidentes nas fronteiras de cada nodo e as

fontes interiores, calcula novas estimativas dos fluxos emergentes, que são usadas para se obterem novas estimativas dos fluxos incidentes nos nodos adjacentes.

Os primeiros códigos para o método DD foram implementados em Java e C#, para as versões paralelas desses códigos usamos no JAVA as Classes Runnable e Threads, e no C# o Namespace System.Threading.Tasks. Entretanto, recorreremos a Linguagem C pelo seu acesso irrestrito a memória e baixos requerimentos de hardware. Assim, o desenvolvimento da versão final do código, que irá tratar dos cálculos numéricos, tem apresentado um melhor desempenho.

Ademais, estão sendo utilizadas as Interfaces de Programações de Aplicações (API's), OpenMP (Open Multi-Processing) e OpenBLAS (Basic Linear Algebra Subprograms), que nos permitem paralelizar nossos códigos e implementar Subprogramas otimizados de Álgebra Linear, respectivamente.

A interface visual do nosso sistema, conta com uma plataforma de software multimídia chamada JavaFX, que é integrada à linguagem JAVA. Além disso, o software SceneBuilder tem nos possibilitado personalizar essa interface por meio de folhas de estilos, Cascading Style Sheets (CSS), agregando uma melhor aparência ao software.

Neste ponto, destacamos que para a integração das linguagens JAVA e C foi adicionada à estrutura de arquivos do projeto um ShellScript, que ao ser acionado pela interface visual executa o código compilado escrito em C. Observamos ainda, que o código escrito em C para cálculos numéricos é totalmente independente da interface visual, que visa apenas facilitar a criação e edição dos arquivos de dados de entrada e manipular visualização dos resultados, por meio de tabelas e gráficos.

Resultados e discussão

Apresentamos na Tabela 1 resultados de um problema homogêneo proposto por Domingues (2018). O problema consiste de um domínio unidimensional de comprimento $L = 10$ cm, discretizado em 20000 células, com $\Sigma_t = 1$ cm^{-1} , $\Sigma_s = 0,8$ cm^{-1} e condições de contorno iguais a 1 nêutron/cm²s, em ambas as extremidades do domínio.

Tabela 1. Resultados numéricos do método DD para o problema homogêneo S_{256} , nas versões de código sequencial e paralelo

	Tempo (s) ^a	Tempo (s) ^b
Sequencial	19,308	11,258
Paralelo	13,388	4,261

a – resultados obtidos por Domingues (2018).

b – resultados gerados pelo autor.

Utilizamos, para gerar nossos resultados, uma máquina equipada com 16 GB de RAM e um processador Intel® Core™ i7-8750H, as configurações da máquina utilizada por Domingues (2018) não foram informadas. Contudo, podemos fazer uma análise entre os resultados sequencial e paralelo, verificamos que Domingues (2018) obteve um ganho de cerca de 44% na execução de seu código paralelo, nosso código paralelo quando comparado ao sequencial apresentou um ganho de cerca de 62%.

Creditamos o bom desempenho do nosso código a nossa estratégia de paralelização do método DD. Implementamos duas seções usando a diretiva **#pragma omp sections**, uma para o cálculo dos fluxos angulares nas direções positivas e outra seção para as direções negativa. Dentro de cada uma dessas seções aplicamos a diretiva **#pragma omp parallel for** para paralelizar o cálculo dessas direções em cada célula de discretização espacial, também utilizamos a mesma diretiva para paralelizar o cálculo da fonte de espalhamento e para a atualização das condições de contorno. Neste ponto, observamos que nossa estratégia difere de Domingues (2018).

Conclusão

Nesta fase do desenvolvimento do projeto temos definido a linguagem de programação C para a implementação dos métodos numéricos. Tendo em vista seu bom desempenho e a praticidade que a biblioteca OpenMP oferece para a paralelização dos códigos. Para a implementação da interface gráfica do software seguimos com o Java, linguagem que apresenta muitas vantagens para esta tarefa.

Como prosseguimento do trabalho estamos desenvolvendo os códigos que implementam o método Matriz Resposta, que terá sua versão sequencial e paralela. Temos, ainda, a intenção de investigar mais cuidadosamente a eficiência das versões paralelas dos métodos, simulando outros problemas da literatura.

Agradecimentos

À Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro, (FAPERJ).

Referências

- BRIESMEISTER, J. F. (Ed.). *MCNPTM-A General Monte Carlo N{Particle Transport Code*. Version 4c. Los Alamos National Laboratory, 2000.
- DOMINGUES, L. J. *Aplicação da técnica de paralelização de programas usando OpenMP na solução numérica de transporte de nêutrons*. Dissertação (Mestrado) – Universidade Federal do Rio Grande do Sul. Porto Alegre, 2018.
- DUDERSTADT, J. J.; MARTIN, W. R. *Transport Theory*. New York, USA: Wiley - Interscience, 1979.
- LEWIS, E. E.; MILLER, W. F. *Computational methods of neutron transport*. Illinois, USA: American Nuclear Society, 1993.
- SILVA, O. P. *Um método de matriz resposta para cálculos de transporte multigrupos de energia na formulação de ordenadas discretas em meios não-multiplicativos*. Tese (Doutorado) – UERJ Instituto Politécnico. Nova Friburgo, 2018.
- WAGNER, J. C. et al. Review of hybrid (deterministic/monte carlo) radiation transport methods, codes, and applications at oak ridge national laboratory. *Progress in Nuclear Science and Technology*, v. 2, n. 104, p. 808{814, 2011.
- UTFPR.C Pthreads. Paraná 15/07/2019. Disponível em: <http://cocic.cm.utfpr.edu.br/progconcorrente/doku.php?id=c_pthreads>.
- OpenBLAS Wiki. Rio de Janeiro, 20/09/2019. Disponível em: <<https://github.com/xianyi/OpenBLAS/wiki>>.
- The OpenMP Architecture Review Board. Rio de Janeiro, 20/09/2019. Disponível em: <<https://www.openmp.org/about/about-us/>>.