Análise e Otimização de Operações de E/S em Aplicações Científicas de Aprendizado de Máquina Orientada pelo Drishti*

Arthur A. da Silva[1], **Thiago Araújo**[1], **Cristiano A. Künas**[1], **Philippe O. A. Navaux**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)

{aasilva, tsaraujo, cakunas, navaux}@inf.ufrgs.br

Resumo. *O avanço do Aprendizado de Máquina Científico (SciML) evidenciou as operações de Entrada e Saída (E/S) como gargalos na Computação de Alto Desempenho (HPC). Usando a ferramenta Drishti para guiar otimizações de dados no benchmark PDEBench, avaliamos três redes (FNO, U-Net, PINN) e revelamos que a eficácia das técnicas depende estritamente da arquitetura. Em modelos limitados por E/S (FNO), o Alinhamento reduziu o tempo em* $\sim 4\%$ *em discos NVMe locais. Em contraste, modelos densos (U-Net) mascaram a latência, e técnicas distribuídas como MPI-IO geram overhead. Demonstra-se que essas otimizações impõem um trade-off marginal à acurácia devido à estocasticidade da formação de lotes.*

1. Introdução

O avanço do SciML em simulações físicas, como dinâmica de fluidos, modelagem climática e física de plasmas, evidenciou gargalos críticos de E/S em ambientes HPC, onde a latência de acesso aos dados limita a eficiência geral [Gunda et al. 2025]. Trabalhos seminais mostram ganhos com MPI-IO e *buffering* em sistemas de arquivos paralelos distribuídos [Cappello et al. 2025, Lewis et al. 2025], porém ainda há pouca evidência da eficácia em arquiteturas de nó único com Non-Volatile Memory Express (NVMe) local.

Embora o perfilamento com o `Drishti` [Bez et al. 2022] permita identificar gargalos de leitura, a literatura carece de evidências empíricas de ganhos práticos nesses cenários. Para preencher essa lacuna, este trabalho utiliza essa ferramenta para guiar otimizações no *benchmark* PDEBench [Takamoto et al. 2022], investigando o impacto dessas otimizações em NVMe local. As contribuições incluem: aplicação prática de otimizações de E/S no *data loader* de modelos SciML, validação empírica das diretrizes recomendadas pela ferramenta, e demonstração de que sua eficácia não é universal, sendo ditada pela arquitetura matemática da rede (espectral, convolucional e PINN).

2. Metodologia

A metodologia segue um ciclo iterativo de modificação guiada por evidências: (1) **Perfilamento:** execução dos modelos com *Darshan DXT* [Xu et al. 2023] para capturar traços detalhados de E/S por processo durante o treinamento, incluindo padrões de acesso e latência; (2) **Análise:** processamento dos logs pelo `Drishti` para hierarquizar gargalos, correlacionar métricas e gerar recomendações; e (3) **Refatoração:** implementação direta das otimizações no *data loader*, fundamentada na literatura e nos padrões observados.

*Este estudo foi apoiado pela CAPES Brasil - Código de Financiamento 001 e pelo edital CNPq/MCTI/FNDCT - Universal 18/2021 sob número 408755/2025-3. Os experimentos deste trabalho utilizaram os recursos da infraestrutura PCAD, `http://gppd-hpc.inf.ufrgs.br`, no INF/UFRGS.

2.1. Ferramentas e Benchmark

O `Drishti` foi selecionado por recomendar otimizações para aplicações, baseadas em seu padrão de operações de E/S, utilizando traços temporais do módulo DXT e heurísticas específicas para sistemas de arquivos paralelos. O estudo de caso adota o PDEBench, *benchmark* de referência para SciML [Sharma et al. 2023], com foco nos problemas de dinâmica de fluidos: **1D Advection** (transporte linear com padrões de acesso constantes) e **1D Burgers** (equação não linear com formação de ondas de choque). A seleção desses *datasets* leves se dá pelo baixo custo computacional, que acelera o processamento na GPU, aumentando a frequência de requisições de leitura e evidenciando gargalos de E/S.

A avaliação contempla três arquiteturas de redes neurais distintas: o **FNO** (*Fourier Neural Operator*), que opera no domínio espectral por meio de transformadas de Fourier; a **U-Net**, arquitetura convolucional clássica de leitura intensiva de malhas espaciais; e a **PINN** (*Physics-Informed Neural Networks*), que incorpora restrições das equações diferenciais diretamente na função de custo para garantir consistência física das soluções.

2.2. Estratégias de Otimização

A Figura 1 exibe os gargalos de E/S detectados na execução padrão de um treinamento do PDEBench. Com base neles, três otimizações foram implementadas isoladamente:

A. **MPI-IO:** Substituição de chamadas POSIX independentes pela interface coletiva de E/S MPI-IO (*Message Passing Interface Input/Output*) no *data loader*, permitindo ao *middleware* agregar pequenas requisições de múltiplos processos em transações maiores e contíguas, reduzindo a contenção de metadados no sistema de arquivos e melhorando a eficiência de acesso paralelo em larga escala em ambientes HPC.

B. **Alinhamento de Requisições:** Ajuste dos *offsets* de leitura e fronteiras de página (4 KB) do SSD local, garantindo mapeamento em blocos físicos inteiros e evitando leitura de páginas parciais (*read amplification* no sistema operacional (SO)).

C. ***Buffering* (Agregação Temporal):** Leitura antecipada de grandes blocos em memória (*staging*) para mitigar a latência de múltiplas chamadas pequenas, com dados carregados em *chunks* na RAM e posteriormente fatiados para alimentar os tensores na GPU, reduzindo a ociosidade do pipeline computacional durante treinamento.

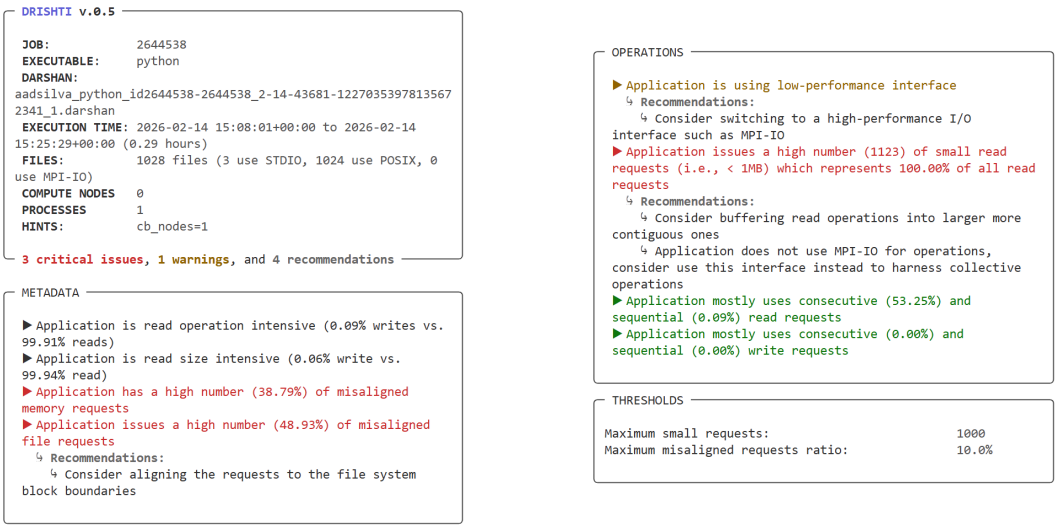

Figura 1. Relatório do `Drishti` gerado pelo treinamento do PDEBench.

2.3. Configuração Experimental e Métricas

Os experimentos utilizaram processadores Intel i9-14900KF, GPUs RTX 4090 e armazenamento NVMe local. O código modificado os parâmetros completos da configuração experimental e métricas de avaliação estão disponíveis no repositório público[1].

Para garantir a confiabilidade estatística dos resultados, cada configuração de rede (FNO, U-Net, PINN) foi executada 10 vezes de forma independente nas quatro variações de coeficientes físicos dos *datasets* **Advection** e **Burgers**. Avaliou-se o tempo total de treinamento (s) e os erros matemáticos de predição da física (MSE, nRMSE e MaxError), calculados diretamente contra os simuladores analíticos de referência do PDEBench, comparando as otimizações implementadas com a linha de base (código original).

3. Resultados e Discussão

A integridade numérica e o desempenho variaram significativamente (Figura 2). O modelo **PINN** falhou ao convergir no Advection (MSE $\approx 0,82$), anulando quaisquer ganhos de E/S. Tal falha decorre de patologias de gradiente em equações convectivas [Toscano et al. 2025], onde o otimizador não equilibra as perdas de contorno e o resíduo da física, aprisionando a rede em mínimos locais espúrios.

A **U-Net** revelou-se estritamente *compute-bound* ($\sim$2550s), pois o alto custo das convoluções densas na GPU mascara a latência de leitura. Esse perfil explica as perdas ocasionais frente à execução padrão: ao aplicar técnicas como *Buffering* e MPI-IO — projetadas para mitigar latências de rede em sistemas distribuídos — em um NVMe local, a CPU perde ciclos gerenciando *locks* e sincronização de E/S em vez de pré-processar tensores [Dantas 2022]. Esse *overhead* gera contenção não determinística e aumenta o tempo médio (ex: de 2566s para 2639s com Alinhamento no Burgers), evidenciando que otimizações de E/S podem degradar modelos computacionalmente intensivos.

[1] `https://github.com/ArthurAndradee/drishti-pdebench-io`

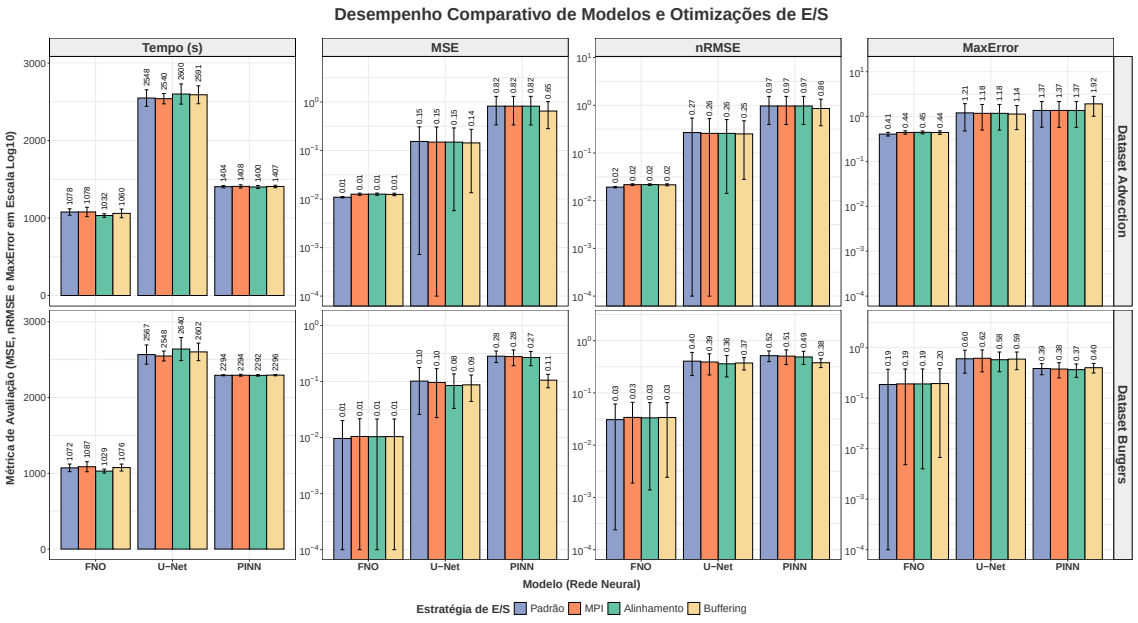

Figura 2. Desempenho comparativo das estratégias de otimização de E/S.

O **FNO** mostrou-se *I/O-bound* ($\sim$1077s), pois suas operações espectrais de Fourier possuem baixa complexidade computacional, transferindo o gargalo de desempenho para o subsistema de armazenamento. O Alinhamento de Requisições mitigou a *read amplification* do SO, reduzindo o tempo em $\approx 4\%$ (1032s no Advection e 1028s no Burgers) com excelente precisão (MSE $\sim 10^{-2}$). Apesar da alta variância no Burgers, no qual ondas de choque induzem o fenômeno de Gibbs e instabilidade espectral [Azizzadenesheli et al. 2024], o *trade-off* preditivo ($0,0109 \rightarrow 0,0125$ no Advection; $0,0096 \rightarrow 0,0104$ no Burgers) permanece dentro do desvio padrão do problema ($\approx 0,011$), validando sua adoção.

4. Conclusão

A aplicação do `Drishti` no *data loader* do PDEBench demonstrou redução de $\sim 4\%$ no tempo do FNO com o Alinhamento de Requisições, com *trade-off* marginal na acurácia. O tempo médio caiu de $1077,66s$ $(\pm 41,57)$ para $1032,04s$ $(\pm 23,38)$, enquanto o aumento do erro $(+0,0008)$ permanece dentro do desvio padrão do modelo $(\pm 0,011)$. Os resultados mostram que a eficácia das técnicas de E/S depende da carga matemática: falhas de convergência (PINN) anulam ganhos, modelos densos (U-Net) mascaram I/O e soluções distribuídas (MPI-IO) geram apenas *overhead* em NVMe local. Conclui-se que o perfilamento conjunto de E/S, processamento e convergência são indispensáveis. Trabalhos futuros avaliarão essas estratégias em ambientes HPC com sistemas de arquivos paralelos e *datasets* maiores, visando validação em cenários de exaescala.

Referências

Azizzadenesheli, K., Kovachki, et al. (2024). Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, 6(5):320–328.

Bez, J. L., Ather, H., and Byna, S. (2022). Drishti: Guiding end-users in the i/o optimization journey. In *IEEE/ACM International Parallel Data Storage Workshop (PDSW)*, pages 1–6, Dallas, TX, USA.

Cappello, F. et al. (2025). Multifacets of lossy compression for scientific data in the joint-laboratory of extreme scale computing. *Future Gener. Comput. Syst.*, 163:107323.

Dantas, M. F. L. (2022). Accelerating deep learning training on high-performance computing with storage tiering. Master's thesis, Universidade do Minho (Portugal).

Gunda, S. K. et al. (2025). Accelerating scientific discovery with machine learning and hpc-based simulations. In *Integr. Mach. Learn. HPC Simul. Anal.*, pages 229–52. IGI.

Lewis, N., Bez, J. L., and Byna, S. (2025). I/o in machine learning applications on hpc systems: A 360-degree survey. *ACM Computing Surveys*, 57(10):1–41.

Sharma, P., Chung, W. T., Akoush, B., and Ihme, M. (2023). A review of physics-informed machine learning in fluid mechanics. *Energies*.

Takamoto, M., Praditia, and othrs (2022). Pdebench: An extensive benchmark for scientific machine learning. *NeurIPS*, 35:1596–1611.

Toscano, J. D., Oommen, V., et al. (2025). From pinns to pikans: Recent advances in physics-informed machine learning. *Mach. Learn. Comput. Sci. Eng.*, 1(1):15.

Xu, C., Snyder, S., Kulkarni, O., et al. (2023). Dxt: Darshan extended tracing. In *Cray User Group (CUG) Meeting*, number OSTI ID: 1490709. U.S. Department of Energy.