

Uma comparação entre métodos baseados em imagens ou parâmetros escalares para a seleção de preconditionadores

Michael Souza,¹ UFC, Fortaleza, CE, Luiz Mariano Carvalho,² UERJ, Rio de Janeiro, RJ, Douglas Augusto,³ FOC, Rio de Janeiro, RJ, Jairo Panetta,⁴ ITA, São José dos Campos, SP, Paulo Goldfeld,⁵ UFRJ, Rio de Janeiro, RJ, José R. P. Rodrigues,⁶ CENPES/Petrobras, Rio de Janeiro, RJ

Resumo. Em computação de alto desempenho (HPC), a solução eficiente de grandes sistemas lineares esparsos é fundamental, sendo os métodos iterativos a escolha predominante. No entanto, o desempenho desses métodos está ligado ao preconditionador escolhido. A natureza multifacetada das matrizes esparsas torna difícil a prescrição universal de preconditionadores. Avançando em metodologias anteriores, esta pesquisa apresenta a representação da esparsidade de uma matrizes por meio de imagens RGB. Utilizando uma rede neural convolucional (CNN), a tarefa de seleção do preconditionador se transforma em um problema de classificação multi-classe. Testes extensivos em 126 matrizes da coleção SuiteSparse enfatizam a adequação do modelo CNN, observando um aumento de 32% na acurácia e uma redução de 25% no tempo computacional.

Palavras-chave. computação de alto desempenho (HPC), sistemas lineares esparsos, métodos iterativos, escolha de preconditionadores, rede neural convolucional, classificação multi-classe

1 Introdução

As simulações numéricas em, por exemplo, engenharia de reservatórios exigem a solução de grandes sistemas lineares com matrizes esparsas, ocupando em muitos casos mais de 50% do tempo de computação [16, 17]. Os métodos de Krylov são os solvers lineares preferidos nesse contexto. No entanto, seu sucesso depende da escolha de um preconditionador adequado, o que ainda é uma tarefa desafiadora, geralmente baseada em tentativa e erro ou na experiência do usuário [26, 29].

Um preconditionador é uma matriz ou operador que modifica o sistema original para acelerar a convergência de solvers lineares iterativos. Vários fatores influenciam a eficácia do preconditionador, incluindo, entre outros, as diversas propriedades - estruturais ou matemáticas - da matriz esparsa, a arquitetura computacional, as estruturas de dados empregadas [3, 4]. Neste trabalho, abordaremos apenas o primeiro desses fatores.

Propor uma representação adequada para matrizes esparsas quando da sugestão de preconditionadores é um desafio, principalmente devido aos requisitos de escalabilidade. Para sistemas esparsos grandes, o ideal é que a complexidade computacional seja linear em relação ao número de elementos não nulos. Essa restrição decorre da necessidade de manter a eficiência à medida que o tamanho do sistema aumenta, o que é comum em, por exemplo, simulações de reservatórios ou em dinâmica de fluidos computacional. A dificuldade em obter representações compactas está em

¹michael@ufc.br

²luizmc@ime.uerj.br

³daa@fiocruz.br

⁴jairo.panetta@gmail.com

⁵goldfeld@matematica.ufrj.br

⁶jrprodrigues@petrobras.com.br

capturar as propriedades da matriz que mais influenciam o desempenho dos preconditionadores dentro dessa restrição de complexidade linear.

Entre os atributos da matriz que influenciam a escolha dos preconditionadores, podemos destacar a ordem da matriz, seus autovalores e valores singulares, o número de condicionamento, o padrão de esparsidade, a densidade e a dominância diagonal, entre outros. Com a exceção do padrão de esparsidade, esses atributos são numéricos. A esparsidade encapsula atributos topológicos sobre a conexão entre os elementos não nulos da matriz. Embora alguns atributos escalares, como a largura de banda e o número de elementos diferentes de zero, forneçam informações sobre padrões complexos de esparsidade, sua sensibilidade é baixa. Essa característica dificulta a obtenção de uma representação numérica adequada para a esparsidade.

Embora não seja o único fator determinante, o padrão de esparsidade afeta o nível de paralelismo possível tanto na aplicação quanto na construção de preconditionadores como $ILU(k)$ [22, 26]. No método multigrid algébrico (AMG), os operadores grosseiros codificam a esparsidade e os valores numéricos para criar uma aproximação multinível do sistema [31]. Dependendo do padrão de esparsidade, até mesmo métodos diretos podem ser aplicáveis para resolver problemas esparsos [11].

Neste trabalho, exploramos técnicas de aprendizagem profunda (DL) para obter automaticamente representações compactas de matrizes esparsas para selecionar preconditionadores. Ampliando a abordagem de Yamada et al. [34], usamos imagens RGB para codificar espacialmente os padrões de esparsidade, facilitando a análise orientada por dados para aumentar a eficiência dos solvers lineares. Diferentemente do trabalho de Yamada, nossa metodologia incorpora a classificação de vários rótulos para identificar uma gama de preconditionadores adequados para uma determinada matriz esparsa. Para processar as representações baseadas em imagens, empregamos uma rede neural convolucional (CNN) [20].

Nossas contribuições são as seguintes:

Modelo multi-rótulo: Em tarefas de classificação convencionais, uma matriz esparsa é frequentemente mapeada para apenas um preconditionador. Entretanto, em nosso conjunto de dados, aproximadamente 20% das matrizes exibem vários preconditionadores ideais. Com base na pesquisa de Yamada et al. [34], nós não seguimos a abordagem de mapeamento de um para um. Ao adotar uma estrutura com vários rótulos, permitimos atribuir vários preconditionadores a uma única matriz, lidando melhor com casos em que vários preconditionadores têm desempenho semelhante.

Modelos escalares versus modelos baseados em imagens: Comparamos modelos escalares e com alguns baseados em imagens. Nossa pesquisa também introduziu um modelo misto que combina os dois atributos, transformando os atributos baseados em imagem em um formato vetorial (**achatamento**) e, posteriormente, integrando-os à tabela de atributos escalares.

Resultados promissores: Em nossa pesquisa inicial, os modelos baseados em imagens superaram os modelos baseados em escalas na seleção do preconditionador. Eles mostram uma probabilidade 32% maior de sugestão de preconditionador ideal e uma chance 26% maior de baixo impacto computacional quando não se escolhe o preconditionador ideal. Esses resultados destacam a eficiência e a eficácia superiores dos modelos baseados em imagens.

O restante deste artigo está assim estruturado: a Seção 2 analisa a literatura relevante, a Seção 3 descreve nossa metodologia, a Seção 4 discute os resultados empíricos e a Seção 5 conclui o artigo, destacando sua importância e sugerindo futuras direções de pesquisa.

2 Trabalhos relacionados

O potencial da aprendizagem profunda (DL) para discernir padrões complexos e facilitar a tomada de decisões orientada por dados foi reconhecido como uma solução eficaz para vários desafios na computação de alto desempenho aplicada à solução de sistemas lineares [14, 23, 33].

Uma área de aplicação da DL é a seleção automática da estrutura de dados para o armazenamento de matrizes esparsas. Sedaghati et al. [30] usaram algoritmos de árvore de decisão para automatizar a seleção do formato de armazenamento com base nas propriedades da matriz. Nisa et al. [24] aplicaram técnicas de aprendizado de máquina para prever os formatos de armazenamento mais adequados para GPUs. A importância de sincronizar a estrutura de dados de armazenamento com a eficiência computacional também é destacada na pesquisa de Barreda et al. [2] e Cui et al. [8] que exploram melhorias de desempenho em diferentes plataformas de computação.

Outra aplicação de DL notável envolve (**autoajuste**) de solvers lineares. Por exemplo, Peairs e Chen [25] utilizaram uma estratégia de aprendizagem por reforço para determinar os parâmetros de reinício ideais para o GMRES. Bhowmick et al. [5] aplicaram a DL para selecionar os melhores solvers para sistemas lineares esparsos em tempo de execução, adaptando-se aos dados e à arquitetura computacional disponível. Dufrechou et al. [13] empregaram técnicas de DL para prever o solver ideal para um sistema linear específico, concentrando-se em situações em que um número limitado de sistemas triangulares é resolvido para a mesma matriz. Em outra abordagem, Funk et al. [15] apresentaram uma técnica de aprendizagem profunda para identificar o solver iterativo ideal para um determinado sistema linear esparsos, obtendo uma acurácia de classificação de 60%.

Uma tendência crescente na DL é o uso de redes neurais para acelerar as operações de álgebra linear. Cui et al. [8] empregaram um sistema de DL para prever a melhor implementação da multiplicação matriz-vetor (SpMV) para uma determinada matriz. Götz e Anzt [19] introduziram uma rede neural convolucional (CNN) para detectar estruturas de blocos em padrões de esparsidade de matriz. Em uma abordagem diferente, Ackmann et al. [1] propuseram o uso de uma rede neural feed-forward como preconditionador. Taghibakhshi et al. [32] introduziram um método inovador usando um agente de aprendizagem por reforço baseado em redes neurais de grafos (GNNs) para construir espaços grosseiros em uma abordagem multigrid.

Embora a DL tenha otimizado substancialmente vários aspectos da álgebra linear computacional, seu potencial na seleção de preconditionadores merece ser mais explorado. Este trabalho é uma contribuição nessa direção.

3 Metodologia

4 Resultados

5 Conclusões

Equações inseridas no trabalho completo devem ser enumeradas sequencialmente e à direita no texto, por exemplo

$$\frac{\partial u}{\partial t} - \Delta u = f, \quad \text{em } \Omega. \quad (1)$$

Consulte o arquivo `.tex` para mais detalhes sobre o código-fonte gerador da equação (1).

6 Tabelas e Figuras

As(os) autoras(es) podem inserir figuras e tabelas no artigo. Elas devem estar dispostas próximas de suas referências no texto.

6.1 Inserção de Tabelas

A inserção de tabela deve ser feita com o ambiente `table`, sendo enumerada, disposta horizontalmente centralizada, próxima de sua referência no texto, e a legenda imediatamente acima dela. Por exemplo, consulte a Tabela 1.

Tabela 1: Categorias dos trabalhos.

Categoria do trabalho	Número de páginas	Tipo do trabalho
1	2	A , B e C
2	entre 5 e 7	apenas C

6.2 Inserção de Figuras

A inserção de figura deve ser feita com o ambiente `figure`, ela deve estar enumerada, disposta horizontalmente centralizada, próxima de sua referência no texto, e legenda imediatamente abaixo dela. **Quando não própria, deve-se indicar/referências a fonte.** Por exemplo, consulte a Figura 1.



Figura 1: Exemplo de imagem. Fonte: indicar.

7 Sobre as Referências Bibliográficas

As referências bibliográficas devem ser inseridas conforme especificado neste padrão, sendo que serão automaticamente geradas em ordem alfabética pelo sobrenome do primeiro autor. Este *template* fornece suporte para a inserção de referências bibliográficas com o Bib_{La}T_EX. Os dados de cada referência do trabalho devem ser adicionados no arquivo `refs.bib` e a indicação da referência no texto deve ser inserida com o comando `\cite`. Seguem alguns exemplos de referências: livro [6], artigos publicados em periódicos [7, 9], capítulo de livro [10], dissertação de mestrado [12], tese de doutorado [21], livro publicado dentro de uma série [18], trabalho publicado em anais de eventos [27], *website* e outros [28]. Por padrão, os nomes de todos os autores da obra citada aparecem na bibliografia. Para obras com mais de três autores, é também possível indicar apenas o nome do primeiro autor, seguido da expressão *et al.* Para implementar essa alternativa, basta remover

“,maxnames=50” do comando correspondente do código-fonte. Sempre que disponível forneça o DOI, ISBN ou ISSN, conforme o caso.

8 Considerações Finais

Esta seção é reservada às principais conclusões e considerações finais do trabalho.

Agradecimentos (opcional)

Seção reservada aos agradecimentos dos autores, caso for pertinente. Por exemplo, agradecimento a fomentos. Se os autores optarem pela inclusão de Agradecimentos, a palavra “(opcional)” deve ser removida do título da seção. Esta seção não é numerada e deve ser disposta entre a última seção do corpo do texto e as Referências.

Referências

- [1] Jan Ackmann et al. **Machine-Learned Preconditioners for Linear Solvers in Geophysical Fluid Flows**. 2020. arXiv: 2010.02866 [physics.aos-ph].
- [2] Maria Barreda et al. “Performance modeling of the sparse matrix–vector product via convolutional neural networks”. Em: **The Journal of Supercomputing** 76.11 (2020), pp. 8883–8900. URL: doi.org/10.1007/s11227-020-03186-1.
- [3] Nathan Bell e Michael Garland. **Efficient sparse matrix-vector multiplication on CUDA**. Rel. técn. Nvidia Technical Report NVR-2008-004, Nvidia Corporation, 2008.
- [4] Michele Benzi. “Preconditioning Techniques for Large Linear Systems: A Survey”. Em: **Journal of Computational Physics** 182.2 (2002), pp. 418–477. DOI: 10.1006/jcph.2002.7176.
- [5] Sanjukta Bhowmick et al. **Application of Machine Learning in Selecting Sparse Linear Solvers**. 2006.
- [6] J. L. Boldrini et al. **Álgebra Linear**. 3a. ed. São Paulo: Harbra, 1986. ISBN: 9788529402024.
- [7] L. O. Contiero et al. “Rainbow Erdős–Rothschild Problem for the Fano Plane”. Em: **SIAM Journal on Discrete Mathematics** (2021). Aceito. DOI: 10.1137/20M136325X.
- [8] Hang Cui et al. “A code selection mechanism using deep learning”. Em: **2016 IEEE 10th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSOC)**. IEEE. 2016, pp. 385–392. DOI: 10.1109/MCSoc.2016.46.
- [9] J. A. Cuminato e V. Ruas. “Unification of distance inequalities for linear variational problems”. Em: **Computational and Applied Mathematics** 34 (2014), pp. 1009–1033. DOI: 10.1007/s40314-014-0163-6.
- [10] P. L. Da Silva e I. L. Freire. “On the group analysis of a modified Novikov equation”. Em: **Interdisciplinary Topics in Applied Mathematics, Modeling and Computational Science, Springer Proceedings in Mathematics and Statistics**. Ed. por M. Cojocaru et al. Vol. 117. Springer, 2015. Cap. 23, pp. 161–166. DOI: 10.1007/978-3-319-12307-3_23.
- [11] Timothy A. Davis, Sivasankaran Rajamanickam e Wissam M. Sid-Lakhdar. “A survey of direct methods for sparse linear systems”. Em: **Acta Numerica** (2016), pp. 383–566. DOI: 10.1017/S0962492916000076.
- [12] G. L. Diniz. “A mudança no habitat de populações de peixes: de rio a represa - o modelo matemático”. Dissertação de mestrado. Unicamp, 1994.

- [13] Ernesto Dufrechou, Pablo Ezzatti e Enrique S. Quintana-Ortí. “Automatic selection of sparse triangular linear system solvers on GPUs through machine learning techniques”. Em: **2019 31st International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)**. IEEE. 2019, pp. 41–47. DOI: 10.1109/SBAC-PAD.2019.00020.
- [14] Thomas L. Falch e Anne C. Elster. “Machine learning-based auto-tuning for enhanced performance portability of OpenCL applications”. Em: **Concurrency and Computation: Practice and Experience** 29.8 (2017). e4029 cpe.4029, e4029. DOI: 10.1002/cpe.4029. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4029>.
- [15] Yannick Funk, Markus Götz e Hartwig Anzt. “Prediction of optimal solvers for sparse linear systems using deep learning”. Em: **Proceedings of the 2022 SIAM Conference on Parallel Processing for Scientific Computing**. SIAM. 2022, pp. 14–24. DOI: 10.1137/1.9781611977141.2.
- [16] Vassilis Gaganis e Nikos Varotsis. “Machine Learning Methods to Speed up Compositional Reservoir Simulation”. Em: **SPE Europec featured at EAGE Conference and Exhibition**. Jun. de 2012, SPE-154505–MS. DOI: 10.2118/154505-MS. eprint: <https://onepetro.org/SPEURO/proceedings-pdf/12EURO/A11-12EURO/SPE-154505-MS/1612365/spe-154505-ms.pdf>.
- [17] Leonardo Gasparini et al. “Hybrid parallel iterative sparse linear solver framework for reservoir geomechanical and flow simulation”. Em: **Journal of Computational Science** 51 (2021), p. 101330. ISSN: 1877-7503. DOI: 10.1016/j.jocs.2021.101330.
- [18] L. T. Gomes, L. C. Barros e B. Bede. **Fuzzy differential equation in various approaches**. Springer Briefs in Mathematics. SBMAC - Springer, 2015. ISBN: 978-3-319-22575-3.
- [19] Markus Götz e Hartwig Anzt. “Machine learning-aided numerical linear algebra: Convolutional neural networks for the efficient preconditioner generation”. Em: **2018 IEEE/ACM 9th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (scalA)**. IEEE. 2018, pp. 49–56. DOI: 10.1109/ScalA.2018.00010. URL: <https://doi.ieeecomputersociety.org/10.1109/ScalA.2018.00010>.
- [20] Zewen Li et al. “A survey of convolutional neural networks: analysis, applications, and prospects”. Em: **IEEE Transactions on neural networks and learning systems** 33.12 (2022), pp. 6999–7019. DOI: 10.1109/TNNLS.2021.3084827.
- [21] S. M. Mallet. “Análise Numérica de Elementos Finitos”. Tese de doutorado. LNCC/MCTI, 1990.
- [22] Jan A. Meijerink e Henk A. van der Vorst. “An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix”. Em: **Mathematics of Computation** 31 (1977), pp. 148–162.
- [23] Suejb Memeti et al. “Using meta-heuristics and machine learning for software optimization of parallel computing systems: a systematic literature review”. Em: **Computing** 101.8 (2019), pp. 893–936. URL: 10.1007/s00607-018-0614-9.
- [24] Israt Nisa et al. “Effective machine learning based format selection and performance modeling for SpMV on GPUs”. Em: **2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)**. IEEE. 2018, pp. 1056–1065. DOI: 10.1109/IPDPSW.2018.00164.
- [25] Lisa Peairs e Tzu-Yi Chen. “Using reinforcement learning to vary the m in GMRES (m)”. Em: **Procedia Computer Science** 4 (2011), pp. 2257–2266. DOI: 10.1016/j.procs.2011.04.246.
- [26] Yousef Saad. **Iterative methods for sparse linear systems**. SIAM, 2003.

- [27] I. L. D. Santos e G. N. Silva. “Uma classe de problemas de controle ótimo em escalas temporais”. Em: **Proceeding Series of the Brazilian Society of Computational and Applied Mathematics**. 2013, pp. 010177-1–6. DOI: 10.5540/03.2013.001.01.0177.
- [28] SBMAC. **Site oficial do Congresso Nacional de Matemática Aplicada**. Online. Acessado em 08/12/2021, <http://www.cnmac.org.br>.
- [29] Jennifer Scott e Miroslav Tůma. “An Introduction to Sparse Matrices”. Em: **Algorithms for Sparse Linear Systems**. Cham: Springer International Publishing, 2023, pp. 1–18. ISBN: 978-3-031-25820-6. DOI: 10.1007/978-3-031-25820-6_1.
- [30] Naser Sedaghati et al. “Automatic selection of sparse matrix representation on GPUs”. Em: **Proceedings of the 29th ACM on International Conference on Supercomputing**. ICS ’15. Association for Computing Machinery, 2015, pp. 99–108. ISBN: 9781450335591. DOI: 10.1145/2751205.2751244.
- [31] Klaus Stüben. “A review of algebraic multigrid”. Em: **Numerical Analysis: Historical Developments in the 20th Century**. Ed. por C. Brezinski e L. Wuytack. Amsterdam: Elsevier, 2001, pp. 331–359. ISBN: 978-0-444-50617-7. DOI: 10.1016/B978-0-444-50617-7.50015-X. URL: <https://www.sciencedirect.com/science/article/pii/B978044450617750015X>.
- [32] Ali Taghibakhshi et al. “Optimization-Based Algebraic Multigrid Coarsening Using Reinforcement Learning”. Em: **Advances in Neural Information Processing Systems**. Ed. por M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 12129–12140. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/6531b32f8d02fece98ff36a64a7c8260-Paper.pdf.
- [33] Ozan Tuncer et al. “Diagnosing Performance Variations in HPC Applications Using Machine Learning”. Em: **High Performance Computing: 32nd International Conference, ISC High Performance 2017, Frankfurt, Germany, June 18–22, 2017, Proceedings 32**. Ed. por Julian M. Kunkel et al. Springer International Publishing, 2017, pp. 355–373. ISBN: 978-3-319-58667-0.
- [34] Kenya Yamada et al. “Preconditioner auto-tuning using deep learning for sparse iterative algorithms”. Em: **2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)**. IEEE. 2018, pp. 257–262. DOI: 10.1109/CANDARW.2018.00055.