Agentic AI Augmentation for Recursive Mixed-Precision Linear Solvers

Vicki Carrica[1], Rabab Alomairy[1,2], Pedro Valero-Lara[3], Alan Edelman[1]
[1]Massachusetts Institute of Technology, Cambridge, MA, USA
[2]Computer, Electrical and Mathematical Sciences and Engineering Division,
King Abdullah University of Science and Technology, KSA.
[3]Oak Ridge National Laboratory, Knoxville, TN, USA
vickicar@mit.edu,rabab.alomairy@mit.edu,valerolarap@ornl.gov,edelman@mit.edu

1 Introduction

Modern HPC systems increasingly rely on specialized low-precision hardware, such as Tensor Cores, to maximize computational throughput. To balance this performance with required numerical accuracy, we previously introduced a nested recursive mixed-precision framework for Cholesky factorization on GPUs [1–3, 5, 7**?**]. By organizing matrices into hierarchical data structures, this design safely assigns low precision to computationally intensive blocks, maximizing highly parallel GEMM operations while preserving numerical stability on critical diagonal blocks.

Extending this specialized framework to new algorithms, such as LU factorization, is an expert-level task requiring deep knowledge of the recursive software architecture, precision placement, and GPU execution. To accelerate this process, we investigate Agentic AI augmentation within the ChatHPC ecosystem [8] using two complementary approaches: in-context learning (ICL) and parameter-efficient fine-tuning.

Rather than prompting a large language model to invent algorithms from scratch, we use our published Cholesky implementation as structured knowledge. Through ICL, the AI agent learns the recursive algorithm organization, quantization rules, and hardware-specific abstractions, transferring these principles to generate a high-performance recursive mixed-precision LU solver. Complementarily, we fine-tune a model using a custom dataset derived directly from the expert framework's structures and implementation patterns. Together, these approaches demonstrate how expert HPC implementations can serve as reusable knowledge assets to accelerate future numerical library development.

2 Methods

This work investigates Agentic AI augmentation for transferring expert HPC knowledge to LU decomposition through two complementary mechanisms: in-context learning (ICL) and parameter-efficient fine-tuning (PEFT). Both approaches leverage our previously published recursive mixed-precision framework as structured reference knowledge.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, Washington, DC, USA
© 2026 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2.1 In-Context Learning

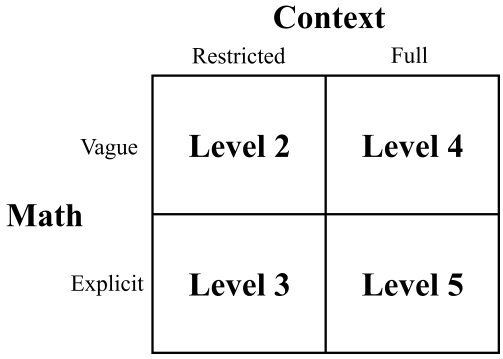


Figure 1: ICL Prompt Design Matrix illustrating the levels across mathematical explicitness and context availability.

To evaluate the ability of modern LLMs to generalize this framework, we tested four frontier models (ChatGPT 5.3 Codex, Gemini Pro 3.1, Claude Opus 4.8, and Claude Fable) across a hierarchy of five ICL prompting levels (Figure 1). These levels progressively scale the provided framework context (restricted versus full recursive data structures and algorithms) and mathematical explicitness (vague versus explicit formulation).

Implementations were assessed using a 15-item framework fidelity checklist to evaluate framework adherence. Generations that successfully executed underwent numerical validation with diagonally dominant matrices and runtime benchmarking against the vendor non-pivoting cuSOLVER FP32 baseline for matrix sizes up to $65{,}536 \times 65{,}536$.

2.2 Fine-Tuning and Data Generation

To reduce dependency on extensive prompt engineering, we explored PEFT within the ChatHPC ecosystem using StarCoder2-15B [4], selected over CodeLlama-7B for its strong pre-existing Julia capabilities. We constructed a dataset of 280 prompt-context-answer examples covering recursive data structures, GPU abstractions, quantization strategies, and mathematical formulations.

Using synthetic instruction augmentation [6, 9], we generated five stylistic variations per prompt to encourage generalized learning over exact memorization. The fine-tuned model was subsequently evaluated on both in-distribution and out-of-distribution tasks to assess its generalization capacity against the ICL approach

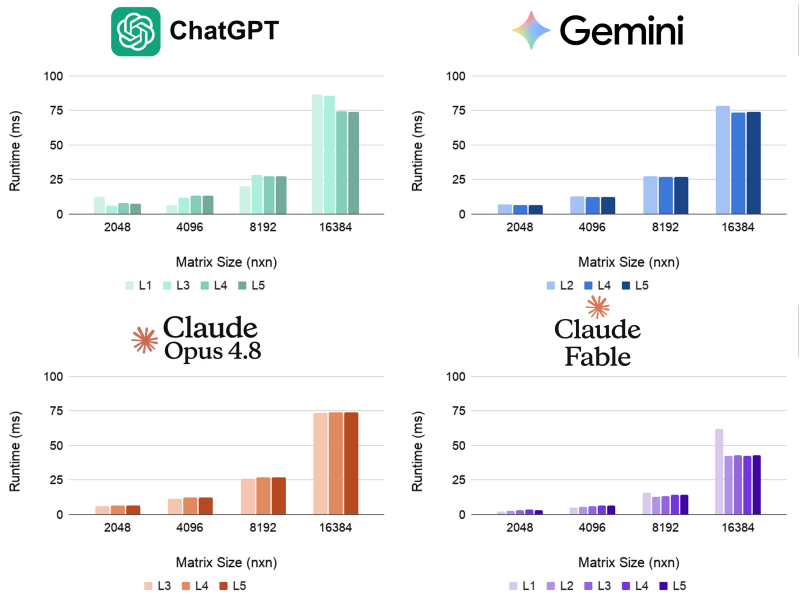


Figure 2: Evolution of factorization runtime across increasing prompt context levels (L1–L5). Only configurations that successfully passed the accuracy threshold are plotted. The y-axis is runtime (lower = better).

and to track failure modes such as mathematical hallucinations and structural errors.

3 Results and Conclusions

3.1 In-Context Learning

Successful generation of recursive mixed-precision linear solvers depends heavily on the availability of context. With restricted context (Levels 1–3), models frequently produced execution failures, API hallucinations, and incomplete recursive architectures, often defaulting to monolithic vendor BLAS routines (Figure 2).

Providing the complete framework context (Levels 4 and 5) substantially improved implementation quality and performance across all evaluated models (Table 1). At Level 5, all four models achieved successful execution and passed numerical validation. Claude Fable demonstrated the strongest framework fidelity, generating a fully nested recursive implementation that satisfied all 15 checklist requirements. By successfully exposing low-precision GEMM operations to H200 Tensor Cores, Fable's Level 5 implementation achieved a 4.9× speedup over the vendor non-pivoting cuSOLVER FP32 baseline for a 65,536 × 65,536 matrix while preserving numerical stability (Figure 3).

Table 1: Evaluation of frontier models across increasing ICL levels.

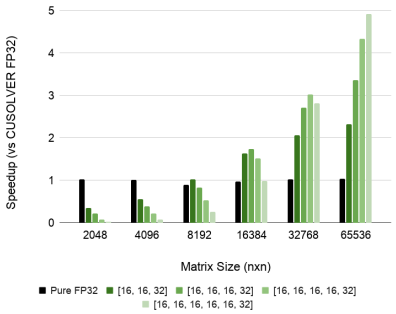
Model	L1	L2	L3	L4	L5
Execution Success					
ChatGPT 5.3 Codex	✓	×	✓	✓	✓
Gemini Pro 3.1	×	✓	×	✓	✓
Claude Opus 4.8	×	×	✓	✓	✓
Claude Fable	✓	✓	✓	✓	✓
Framework Fidelity (Checklist Score / 15)					
ChatGPT 5.3 Codex	8	14	14	14	14
Gemini Pro 3.1	8	10	10	10	10
Claude Opus 4.8	8	9	9	10	10
Claude Fable	8	11	11	11	**15**
Nested Recursive LU Implementation					
ChatGPT 5.3 Codex	No	Full	Full	Full	Full
Gemini Pro 3.1	No	Partial	Partial	Partial	Partial
Claude Opus 4.8	No	Partial	Partial	Partial	Partial
Claude Fable	No	Partial	Partial	Partial	**Full**

3.2 Fine-Tuning

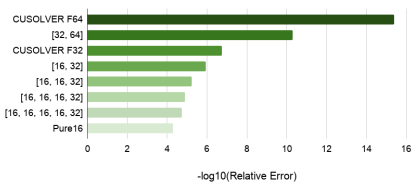
Parameter-efficient fine-tuning of StarCoder2-15B improved overall task accuracy from a baseline of **0% (0/40)** to **55% (22/40)**, achieving 100% accuracy on all in-distribution prompts (Table 2). However, the model failed to generalize to out-of-distribution (OOD) tasks, scoring 0% on code synthesis. As illustrated in Figure 4, the primary OOD failure modes were mathematical hallucinations, structural errors, and exact memorization of training templates.

These results highlight an important trade-off: fine-tuning provides fast, token-efficient inference but struggles to generalize beyond memorized structures. Conversely, in-context learning requires large context windows and capable frontier models, but successfully transfers complex HPC implementation principles to previously unseen numerical algorithms.

Overall, we presented an Agentic AI augmentation framework leveraging ICL and fine-tuning to generate recursive mixed-precision linear solvers. While fine-tuning failed to generalize out-of-distribution,



(a) Speedup over non-pivoting cuSOLVER FP32.



(b) Relative numerical error.

Figure 3: Performance and numerical accuracy of the Claude Fable Level 5 generated nested-recursive mixed-precision LU implementation: (a) speedup relative to the non-pivoting cuSOLVER FP32 baseline and (b) relative error across mixed-precision hierarchies.

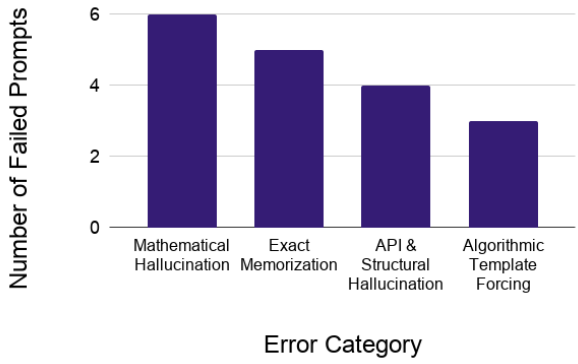


Figure 4: Distribution of failure modes across the 18 Out-of-Distribution (OOD) prompts for the fine-tuned StarCoder2-15B model.

Table 2: Fine-Tuned StarCoder2-15B Accuracy Across Prompt Categories

Evaluation Category	Total Prompts	Successful	Accuracy (%)
In-Distribution (Exact Match)	10	10	100%
In-Distribution (Paraphrased)	10	10	100%
OOD (Code Synthesis)	10	0	0%
OOD (Conceptual Reasoning)	10	2	20%
Overall	**40**	**22**	**55%**

full-context ICL enabled frontier models to successfully generate a nested recursive LU factorization. The Claude Fable implementation achieved a 4.9× speedup over cuSOLVER FP32 baseline while maintaining accuracy.

References

[1] Harun Bayraktar, Cole Brower, John Gunnels, Greg Henry, Cherin Joseph, Jack Kosaian, Dmitry Lyakh, Lukas Mosimann, Victor Podlozhnyuk, Addison Richards, et al. 2026. Exceeding the Numerical and Performance Characteristics of IEEE-754 SGEMM with BFloat16 Tensor Cores on GPUs for Scientific Computing. *arXiv preprint arXiv:2605.16617* (2026).

[2] Vicki Carrica, Rabab Alomairy, Evelyne Ringoot, and Alan Edelman. 2025. Accelerating Linear Solve with Mixed Precision Nested Recursive Subdivision on AI Hardware. (2025).

[3] Vicki Carrica, Rabab Alomairy, Evelyne Ringoot, and Alan Edelman. 2026. Hierarchical Recursive Precision for Accelerating Symmetric Linear Solves on MXUs. *arXiv preprint arXiv:2601.08082* (2026).

[4] Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, Tianyang Liu, Max Tian, Denis Kocetkov, Arthur Zucker, Younes Belkada, Zeyu Wang, Qian Liu, Dmytro Abulkhanov, Indnish Paul, Zhuang Li, Wen-Ding Li, Megan Risdal, Jia Li Li, Jian Zhu, and Terry Yue Zhuo. 2024. StarCoder 2 and The Stack v2: The Next Generation. *arXiv preprint arXiv:2402.19173* (2024).

[5] Hatem Ltaief, Rabab Alomairy, Qinglei Cao, Jie Ren, Lotfi Slim, Thorsten Kurth, Benedikt Dorschner, Salim Bougouffa, Rached Abdelkhalak, and David E Keyes. 2024. Toward capturing genetic epistasis from multivariate genome-wide association studies using mixed-precision kernel ridge regression. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–12.

[6] Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2024. WizardCoder: Empowering Code Large Language Models with Evol-Instruct. In *The Twelfth International Conference on Learning Representations (ICLR)*.

[7] Satoshi Matsuoka. 2026. FP8 is All You Need (Part 1): Debunking Hardware FP64 as the HPC Holy Grail (June 13th version). *arXiv preprint arXiv:2606.06510* (2026).

[8] Pedro Valero-Lara, Aaron R. Young, Jeffrey S. Vetter, Zheming Jin, Swaroop Pophale, Mohammad Alaul Haque Monil, Keita Teranishi, and William F. Godoy. 2025. ChatHPC: Building the Foundations for a Productive and Trustworthy AI-Assisted HPC Ecosystem. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2025, St. Louis, MO, USA, November 16-21, 2025*. ACM, 458–474. doi:10.1145/3712285.3759787

[9] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-Instruct: Aligning Language Models with Self-Generated Instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 13484–13508.