Query-driven Visualization over Compressed Scientific Data via Data-Index Co-Compression

Ning Yan[1], Sheng Di[2], Kai Zhao[3], Hanqi Guo[4], Franck Cappello[2], Lipeng Wan[1]

[1]Georgia State University, [2]Argonne National Laboratory, [3]Florida State University, [4]The Ohio State University

§1 Introduction

- Large-scale scientific simulations generate petabyte-scale datasets, making storage and I/O major bottlenecks.
- Error-bounded lossy compression has become indispensable for alleviating these bottlenecks.
- Many visualization algorithms are inherently query-driven and require only a small subset of the data.
- However, conventional visualization workflows over compressed scientific data still reconstruct the entire dataset before visualization.
- We apply data–index co-compression to enable efficient query-driven visualization over compressed scientific data, achieving up to 50.1× speedup with comparable visualization quality.

§2 Motivation

- Efficient query-driven visualization over compressed scientific data requires selective decompression, where only the subset of data relevant to a query is reconstructed
- Selective decompression, in turn, depends on accurately identifying the candidate data regions satisfying the query predicate.
- The dataset is partitioned into independent small blocks to enable partial decompression.
- A block-level index is constructed for efficient candidate block lookup.
- For large-scale datasets, block-level indexes introduce substantial storage overhead, significantly reducing the overall compression ratio.

§3 Block-Level Index

To enable efficient query-driven visualization, a block-level index is constructed over the compressed dataset. Each block records its value range $[min, max]$, and these block-level ranges are organized into an augmented AVL tree, as illustrated in Fig. 1. Each node maintains a *max_high* attribute—the maximum upper bound within its subtree—to enable efficient subtree pruning.

The AVL index supports $O(\log B)$ candidate lookup, enabling selective decompression instead of an $O(N)$ full scan.

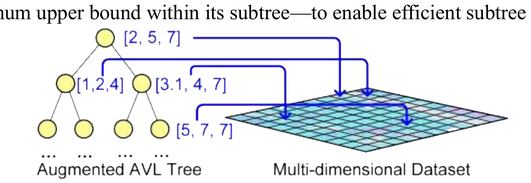


Fig. 1. Illustration of Index Based on an Augmented AVL Tree.

§4 Index Storage Overhead

- The block-level index enables efficient candidate block lookup by filtering irrelevant blocks.
- The storage cost of the block-level index scales with dataset size.
- When stored separately, the index introduces significant storage overhead, substantially reducing the overall compression ratio.

TABLE I
Average Compression Ratio with and without Block-Level Index

Dataset	Without Index	With Index	CR drops by (%)
NYX	9.32	5.97	35.9
WarpX	36.91	11.84	67.9
Gray-Scott	9.29	6.02	35.2

- A block-level index reduces the overall compression ratio by 35.2%–67.9% across representative scientific datasets, as shown in Table I.
- Reducing this index overhead while preserving efficient query processing is therefore the key challenge.

Acknowledgment

This research was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research (ASCR), under Contract No. DE-AC02-06CH11357. Additional support was provided by the National Science Foundation under Grant Nos. 2531903, 2311875, 2344717, and 2514036.

§5 Framework

To enable efficient query-driven visualization over compressed scientific data, we adopt a data–index co-compression framework to reduce index storage overhead while preserving efficient selective data access and processing, as illustrated in Fig. 2.

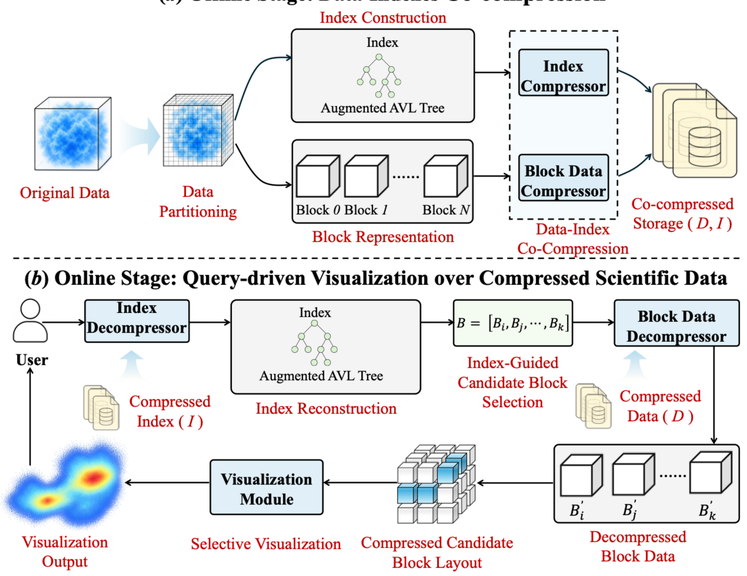


Fig. 2. Overview of the Proposed Data–Index Co-Compression Framework.

1. **Augmented AVL tree** — efficient candidate block filtering
2. **Data–Index Co-Compression** — jointly compresses index and block data to reduce storage overhead
3. **Query-driven visualization over compressed data** — decompresses and visualizes only selected candidate blocks

§6 Evaluation

TABLE II
Average Compression Ratio for Data, Index, and Overall

Dataset	Data	Index	Overall
NYX	9.32	4.33	8.61
WarpX	36.91	9.72	31.67
Gray-Scott	9.29	8.32	9.14

Compression Performance
The overall compression ratio remains comparable to data-only compression, demonstrating that data–index co-compression effectively reduces index storage overhead, as shown in Table II.

End-to-End Speedup
The proposed framework achieves average end-to-end speedups of 50.1× on NYX, 15.7× on WarpX, and 11.7× on Gray-Scott, as shown in Figs. 3–5.

Visualization Quality
The extracted isosurfaces closely match the originals, demonstrating high geometric fidelity, as shown in Figs. 6.

Reference

[1] Yan, N., Di, S., Zhao, K., and Wan, L. "Eureka: Enabling Fine-Grained Access and Range Queries on Compressed Scientific Data via Data-Index Co-Compression." *Proceedings of the VLDB Endowment (PVLDB)*, vol. 19, no. 4, pp. 631–643, 2025.
[2] Stockinger, K., Shalf, J., Wu, K., and Bethel, E. W. "Query-Driven Visualization of Large Data Sets." *Proceedings of the IEEE Visualization Conference (VIS)*, pp. 167–174, 2005.

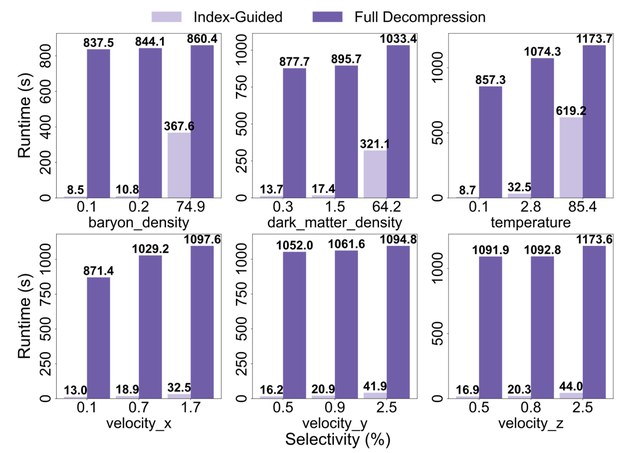


Fig. 3. Isosurface Extraction Runtime on NYX.

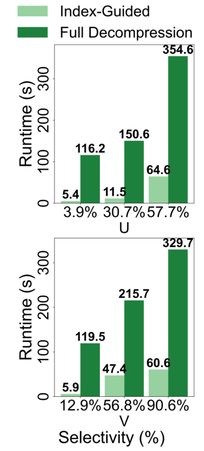


Fig. 5. On Gray-Scott.

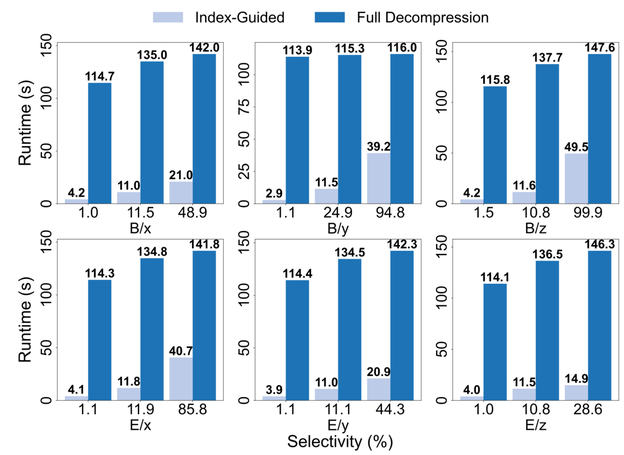


Fig. 4. Isosurface Extraction Runtime on WarpX.

[50.1]×
NYX

[15.7]×
WarpX

[11.7]×
Gray-Scott

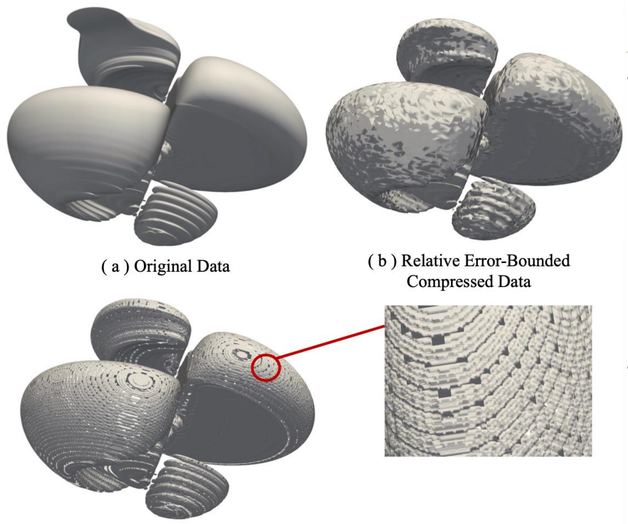


Fig. 6. Isosurface Comparison for the B/y Variable in WarpX.

§7 Conclusion & Future Work

- This work can achieve substantial end-to-end speedups while minimizing index storage overhead and preserving the overall geometric structure of extracted isosurfaces, albeit with boundary artifacts.
- Future work will investigate boundary-aware compression and reconstruction techniques to improve cross-block isosurface continuity while preserving query-driven efficiency.