In-Situ Data Reduction for AMR-Based Cosmology Simulations

Daoce Wang[1], Jesus Pulido[2], Pascal Grosset[2], Sian Jin[1], Jiannan Tian[1], James Ahrens[2], Dingwen Tao[1]

[1]Washington State University, [2]Los Alamos National Laboratory

ABSTRACT

This work investigates an effective in-situ error-bounded lossy compression for Nyx, an adaptive mesh refinement (AMR) based cosmology application. Our contribution is threefold: (1) We explore the best-fit in-situ error-bounded lossy compressor (including SZ and TTHRESH) for Nyx considering both compression ratio and post-analysis quality. (2) We propose an approach to adaptively optimize the compressor for different AMR levels based on our developed metric and data characteristics. (3) Our evaluation shows that our approach can improve the compression ratio by 1.7× over the baseline (i.e., from 66 to 116) with the same post-analysis quality.

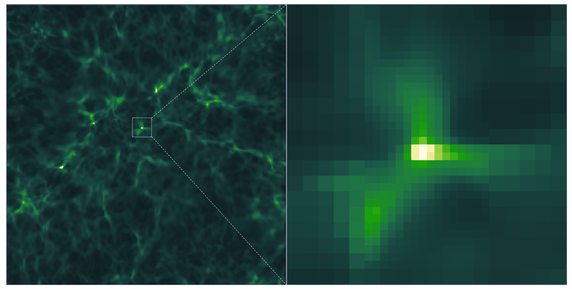


Fig 1: Vis. of Nyx's baryon field (brighter for higher resolution).

INTRODUCTION AND MOTIVATION

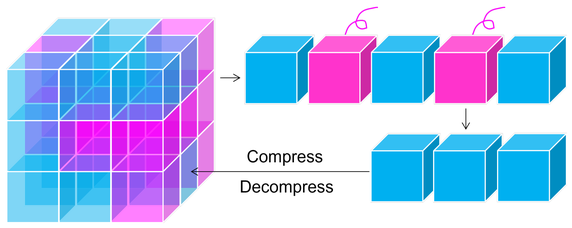
Adaptive mesh refinement and Nyx application.

AMR is a method of adapting the accuracy of a solution by using a non-uniform grid for increasing computational and storage savings. Nyx is an AMR-based code that investigates the evolution of the universe. The size of data generate by Nyx could be prodigious, which makes it urgent to introduce lossy compression to further save time and space for users. As shown in Fig 1, Nyx uses different resolutions in different data regions.

Lossy compression for AMR data.

Compared to lossless compression, error-bounded lossy compression can offer tens to hundreds of higher compression ratios while controlling the loss of post-analysis quality. Some previous works [2] [4] have studied lossy compression for AMR data, but they still suffer from several significant drawbacks: (1) They do not target in-situ compression or consider application-specific post-analysis. (2) They do not optimize compression by utilizing data features (e.g., topology and sparsity) of different AMR levels, which can provide useful spatial information for lossy compression. (3) They do not perform an evaluation on real-world HPC application datasets.

To this end, we explore the best-fit in-situ lossy compressor for AMR-based cosmology simulations and optimize it for each AMR level based on the data features with Nyx datasets.



Sparse Tensor

METHODOLOGY AND EVALUATION

Best-fit Compressor

We explore the best-fit compressor in terms of power spectrum, an FFT-based post-analysis for the Universe's matter distribution.

The upper part of Fig 2 shows that the compression ratio of SZ [3] is better than TTHRESH under the same PSNR. However, the lower part of Fig 2 shows that under the same 1% relative error bound (i.e., the red dashed line) of power spectrum for the first ten outputs, TTHRESH [1] has an 8× higher compression ratio than SZ. This is because of two factors: (1) Power spectrum can tolerate more errors produced by TTHRESH. For example, compared to SZ, TTHRESH has a lower PSNR but a similar power spectrum quality (i.e., the blue box vs. orange box), which is consistent on baryon density and V-z. (2) The principle of TTHRESH and power spectrum are similar. Both of them try to preserve the global information of data, whereas SZ focuses more on preserving the local information. To demonstrate this, we partition the data, compress and decompress each subblock by TTHRESH, and reassemble the reconstructed data for post-analysis. As shown in Fig 3, the more subblocks we use (hence smaller globality), the larger the error.

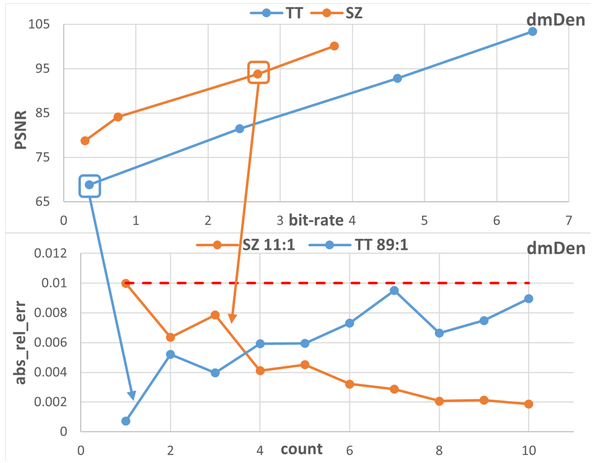


Fig 2: Rate-distortion (upper) and relative power spectrum error (bottom) with TTHRESH and SZ.

Because of the globality, TTHRESH seems more suitable than SZ for Nyx, but as our goal is in-situ compression, the speed is another major consideration. We test the CPU time of TTHRESH and SZ and find that TTHRESH is 100× slower than SZ (568 secs vs 4.3 secs). This is because the time complexity of HOSVD (the decoration step in TTHRESH) is very expensive (i.e., $O(N^4)$ for N^3 input data), taking about 98% of the total compression time, which is infeasible for in-situ solution. Thus, we focus on SZ in this work.

Compressor Optimization

In this section, we propose an approach to optimize the compressor for AMR data based on the data characteristics of each AMR level.

The workflow of Nyx is shown in Fig 4, that is, Nyx will generate plot-file, which can be converted to 1d binary data. We then reshape the 1d data into 3d for post-analysis. A straightforward idea is to compress the 1d data for each AMR level directly. However, in that way, we will lose almost all the topology information, which is of vital importance for data compression as aforementioned.

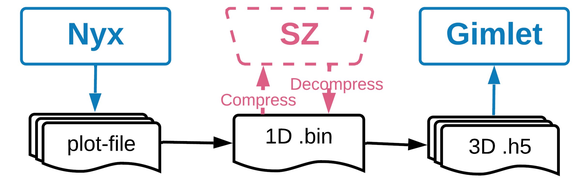
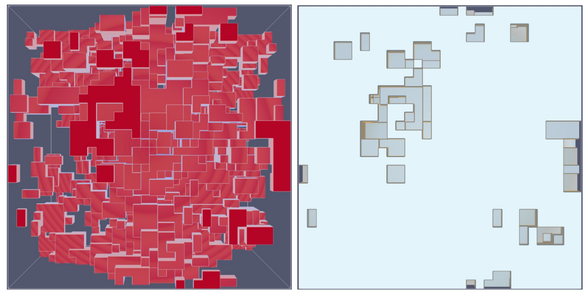


Fig 4: Work flow of Nyx and baseline compression

Another approach is to compress the 3d data, but the distribution of the finest level is very complex as shown in Fig 5a. Therefore, we propose to use a sparse tensor representation (removing the empty regions) to optimize the topology: (1) Partition the data into subblocks. (2) Remove the empty blocks and place the remaining blocks into a 4D array. (3) Pass the 4D array to SZ for compression, and (4) Emplace the subblocks in the decompressed 4D array back to the original data. Overall, we can both keep the topology information and reduce the data size.



(a) Fine, best subblock size: 16 **(b) Coarse, best subblock size: 256**

Fig 5: Distribution and best subblock size for each AMR level

An interesting problem with sparse tensor representation is what the sub-block size should be. On the one hand, if it is too large, we have to fill the empty regions with many zeros, which would greatly reduce the compression ratio of SZ (called "zero-data boundaries"). On the other hand, a small subblock size results in too many boundaries between data (called "data-data boundaries"), which would also affect SZ. We find out that the subblock size depends on the sparsity of the data. For example, as shown in Fig 5, the finest level of z10 is fairly spares, 76.8% of the data is empty, leading to a small block size. By contrast, the coarse level is much denser, only 23.2% of the data are empty, thus a large subblock size is needed.

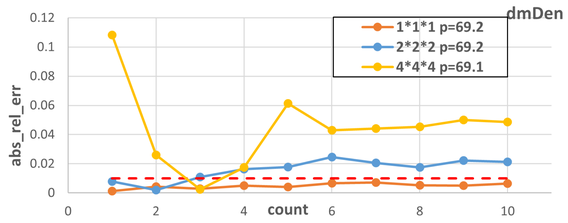
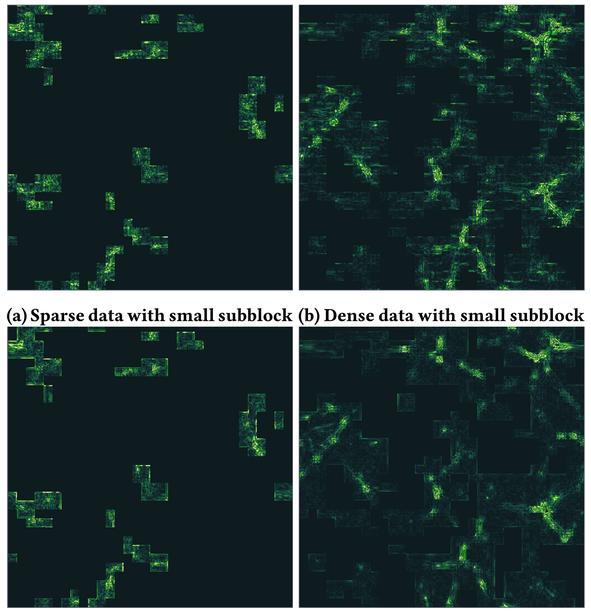


Fig 3: Relative power spectrum error of block-wise-TTHRESH. Legend shows the number of subblocks and PSNR.

We visualize the compression error with different sparsities and subblock sizes. For the sparse data, if we use a larger subblock size, the zeros will significantly affect the quality of SZ, resulting in obvious error at the edge of each subblock, as shown in Fig 6c. For the dense data, if we use a small subblock size, the error will increase almost everywhere, as shown in Fig 6b, since we lose the topology information; if we use a large subblock size, the error is more acceptable, as shown in Fig 6d. After our optimization, SZ can significantly improve the compression over the baseline (i.e., from 66 to 116).



(a) Sparse data with small subblock **(b) Dense data with small subblock**

(c) Sparse data with large subblock **(d) Dense data with large subblock**

Fig 6: Error slice for different sparsity and subblock sizes

CONCLUSION AND FUTUREWORK

In this work, we find the best-fit in-situ lossy compressor for Nyx in terms of both compression ratio and post-analysis quality. We also optimize the compressor for each AMR level based on the data features. Our solution can deliver up to 116× of compression ratio with only 1% post-analysis quality loss. We plan to further optimize each AMR level by using different compression configurations (e.g., error bound) and apply our approach to more AMR applications.

REFERENCES

[1] Rafael Ballester-Ripoll, Peter Lindstrom, and Renato Pajarola. 2020. TTHRESH: Tensor Compression for Multidimensional Visual Data.IEEE Transactions on visualization and Computer Graphics26, 9 (2020), 2891–2903. https://doi.org/10.1109/TVCG.2019.2904063

[2] Huizhang Luo, Junqi Wang, Qing Liu, Jieyang Chen, Scott Klasky, and Norbert Podhorszki. 2021. zMesh: Exploring Application Characteristics to Improve Lossy Compression Ratio for Adaptive Mesh Refinement. In2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS). 402–411. https://doi.org/10.1109/IPDPS49936.2021.00048

[3] Dingwen Tao, Sheng Di, Zizhong Chen, and Franck Cappello. 2017. Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization. In2017 IEEE International Parallel and Distributed Processing Symposium. IEEE, 1129–1139.

[4] Mengxiao Wang, Huizhang Luo, Qing Liu, and Hong Jiang. 2019. Load-aware Elastic Data Reduction and Re-computation for Adaptive Mesh Refinement. In2019 IEEE International Conference on Networking, Architecture and Storage (NAS).

LA-UR-21-28494