A High-Quality Workflow for Multi-Resolution Scientific Data Reduction and Visualization

Daoce Wang*, Pascal Grosset†, Jesus Pulido†, Tushar M. Athawale‡, Jiannan Tian *, Kai Zhao §,
Zarija Lukić¶, Axel Huebl¶, Zhe Wang‡, James Ahrens†, Dingwen Tao‖

* Indiana University, Bloomington, IN, USA; {daocwang, jti1}@iu.edu
† Los Alamos National Laboratory, Los Alamos, NM, USA; {pascalgrosset, pulido, ahrens}@lanl.gov
‡ Oak Ridge National Laboratory, Oak Ridge, TN, USA; {athawaletm, wangz}@ornl.gov
§ Florida State University, Tallahassee, FL, USA; kai.zhao@fsu.edu
¶ Lawrence Berkeley National Laboratory, Berkeley, CA, USA; {zarija, axelhuebl}@lbl.gov
‖ SKLP, Institute of Computing Technology, Chinese Academy of Sciences, China; taodingwen@ict.ac.cn

***Abstract*—Multi-resolution methods such as Adaptive Mesh Refinement (AMR) can enhance storage efficiency for HPC applications generating vast volumes of data. However, their applicability is limited and cannot be universally deployed across all applications. Furthermore, integrating lossy compression with multi-resolution techniques to further boost storage efficiency encounters significant barriers. To this end, we introduce an innovative workflow that facilitates high-quality multi-resolution data compression for both uniform and AMR simulations. Initially, to extend the usability of multi-resolution techniques, our workflow employs a compression-oriented Region of Interest (ROI) extraction method, transforming uniform data into a multi-resolution format. Subsequently, to bridge the gap between multi-resolution techniques and lossy compressors, we optimize three distinct compressors, ensuring their optimal performance on multi-resolution data. These optimizations can improve the compression ratio of SOTA approaches by up to 3.3× under the same data quality loss. Lastly, we incorporate an advanced uncertainty visualization method into our workflow to understand the potential impacts of lossy compression. Experimental evaluation demonstrates that our workflow achieves significant compression quality improvements.**

I. Introduction

In recent years, the complexity and costs associated with scientific simulations have significantly increased. To address these challenges, numerous HPC simulation tools have adopted multi-resolution methods, such as the Adaptive Mesh Refinement (AMR) technique [1–3]. AMR aims to reduce computational expenses while preserving the accuracy of simulation outcomes. Unlike traditional uniform mesh techniques that apply consistent resolution throughout the simulation space, AMR employs a dynamic approach. It selectively increases resolution in regions of interest, thereby optimizing computational resource usage and minimizing storage requirements.

While AMR offers significant benefits in terms of computational, storage, and memory efficiency, its implementation in some scientific simulations is hindered by several challenges. First, integrating AMR can be technically demanding, requiring substantial modifications to existing numerical algorithms and simulation codes, which may not be feasible for all projects. In some instances, simulation algorithms might not accommodate specific geometries or the dynamic adjustments of the grid throughout the simulation's evolution. Additionally, AMR algorithms introduce complexity in grid management and error control, posing optimization challenges for certain simulations, especially those involving highly complex phenomena like convex plasma shapes. For example, in WarpX electromagnetic simulation [4], mesh refinement is currently restricted to disjoint cuboids, which limits the full flexibility offered by AMR [5, 6].

In order to enable uniform-grid simulations to benefit from multi-resolution storage, thereby reducing disk usage, I/O (input/output) time, and memory footprint in visualization without complicating the simulation process, previous work [7–10] has adapted the multi-resolution storage approach for uniform grids. These methods can, for example, store regions of interest at full resolution while representing less critical areas at a lower resolution for visualization or analysis. *However, the space saved from using multi-resolution alone is often not enough.* For instance, a multi-resolution dataset with 0.5×1024^3 mesh points at the coarse level and 0.5×2048^3 at the fine level could yield about 1 TB of data per snapshot. Consequently, conducting five simulations with 200 snapshots would require a total disk storage of 1 PB. Simulations used in Exascale scenarios can be even larger than that, using many thousands of points per axis [4], making data size reduction a timely need.

To this end, data compression can be utilized alongside multi-resolution techniques to further reduce I/O and storage costs. However, traditional lossless compression methods provide limited data volume reduction for scientific simulations, typically achieving compression ratios of only up to 2×. As a solution, a new generation of error-bounded lossy compression techniques, such as SZ [11–13], ZFP [14], MGARD [15] and their GPU versions [16–18], have been widely used in the scientific community [13, 14, 19–29] due to their ability to offer high compression ratios while maintaining controllable accuracy impacts on various scientific applications.

While lossy compression has the potential to significantly reduce I/O and storage costs for multi-resolution data, its effective application in this context remains under-explored. Three recent studies have targeted the development of efficient lossy compression methods for multi-resolution data including AMR data. zMesh [30] was proposed to reorder AMR data using z-order across different refinement levels into a 1D array,

Dingwen Tao is the corresponding author.
Kai Zhao and Jiannan Tian's participation in this work ended on 03/26/2024.

SC24, November 17-22, 2024, Atlanta, Georgia, USA
979-8-3503-5291-7/24/$31.00 ©2024 IEEE

leveraging data redundancy. However, zMesh cannot leverage higher-dimension compression by compressing data in a 1D, leading to a loss of spatial information in higher-dimension data. On the other hand, TAC [31, 32] improved zMesh's compression quality through adaptive 3D compression. While zMesh and TAC offer offline compression solutions for AMR data, they did not delve into in-situ compression, which could notably reduce the I/O cost. AMRIC [33] addressed this by introducing an in-situ AMR compression framework designed to lower I/O costs while improving compression quality for AMR applications.

These efforts have primarily focused on optimizing multi-resolution data compression for block-wise compressors like SZ2 and ZFP. The block-wise nature of these compressors enables higher speed but also renders them susceptible to compression artifacts due to the loss of spatial information between blocks. In contrast, non-block-wise (global) compressors like SZ3, despite their lower throughput, often achieve better compression quality in most cases by leveraging prediction across the entire input data. However, SZ3's compression approach presents significant challenges when applied to multi-resolution data, a topic that will be further discussed in §III-A.

To this end, this paper proposes a comprehensive workflow for compressing multi-resolution data, suitable for both adaptive data derived from uniform-resolution simulations and AMR data. Our strategy not only addresses SZ3's performance issues with multi-resolution data but also introduces a novel post-processing technique to enhance data quality from block-wise compressors like SZ2 and ZFP. Moreover, compression may result in compression artifacts, and there has been no study on identifying potential compression artifacts. Thus, we explore the ramifications of compression-induced uncertainty, aiding users in understanding how compression affects their data.

Our primary contributions are as follows:

- We employ a compression-oriented, adaptive region-of-interest (ROI) method to convert uniform data into multi-resolution data, thereby enhancing storage efficiency while maintaining the quality of visualization and post-analysis.
- We propose SZ3MR, an optimization of the state-of-the-art global lossy compressor SZ3 for multi-resolution data, incorporating dynamic padding and adaptive error bounds within the SZ3 compressor to improve prediction accuracy and compression quality.
- We develop an efficient and effective error-bounded post-processing solution that leverages spatial information across each compressed block to significantly enhance the quality of block-wise compressors (e.g., SZ2/ZFP). This solution is also adaptable to improving multi-resolution data compression with global compressors like SZ3.
- We investigate the uncertainty introduced by lossy compression, an under-explored topic, by integrating a cutting-edge uncertainty visualization technique. This enables a clearer understanding of how compression affects the data through visual representation (will be detailed in §III-C).
- Our experiments show significant compression performance improvements with low overhead for five scientific applications. Our workflow is also integrated into real-world scientific applications, WarpX and Nyx, for in-situ processing.

II. Background

A. Lossy Compression for Scientific Data

Recent research has introduced high-precision lossy compression algorithms for scientific data, notably SZ [13, 34, 35], ZFP [14], MGARD [36], and TTHRESH [37], which differ from traditional compressors like JPEG by targeting floating-point data with strict error control based on user requirements. This work focuses on three compression algorithms: SZ2, ZFP, and SZ3. The key difference between them is that SZ2 and ZFP are block-wise, while SZ3 is global (non-block-wise). SZ2 and ZFP partition the input data into smaller blocks (e.g., $4\times4\times4$ for ZFP) and process them separately to leverage the spatial information. Specifically, SZ2 uses the Lorenzo predictor or linear regression for each block, and ZFP applies a DCT-like transform. In contrast, SZ3 employs global interpolation prediction across the entire input data without partitioning it. SZ2 and ZFP offer fast compression speeds, but the global interpolation of SZ3 enables it to capture more spatial information across the dataset, thus producing a higher compression quality/ratio than the block-wise SZ2/ZFP. We refer readers to [13, 14, 35] for more details.

B. AMR Method and Multi-resolution Data

By using a non-uniform grid, AMR can significantly enhance computational efficiency and lower storage requirements while still achieving the desired accuracy level. In AMR applications, the mesh or spatial resolution is dynamically adjusted according to the simulation's demands, implementing a finer mesh in areas of greater significance or interest and a coarser mesh in less critical regions as depicted in Fig. 1. In AMR application, the mesh is refined based on specific criteria, such as when the average value of a block exceeds predefined thresholds.

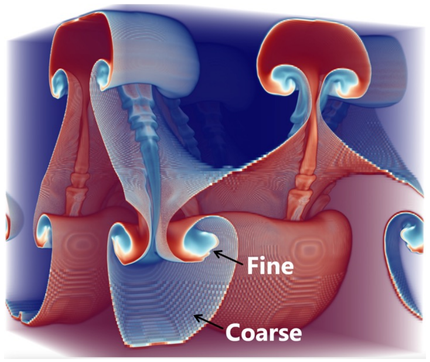
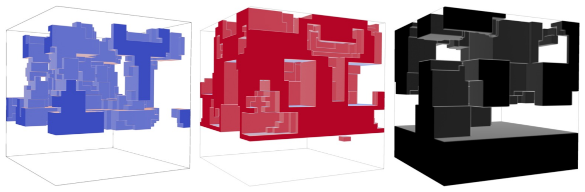
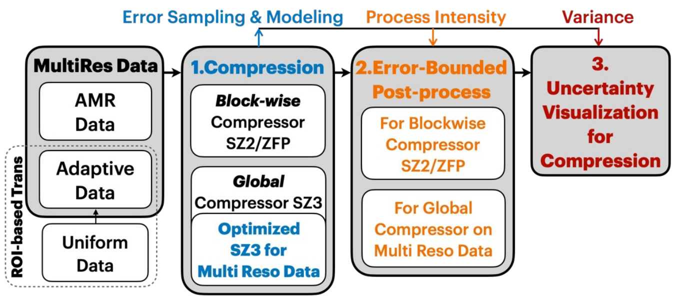


Fig. 1: Example of an AMR dataset of Rayleigh–Taylor instability.



(a) Fine level (b) Mid level (c) Coarse level

Fig. 2: Vis of data distributions for different level for Fig. 1.



Error Sampling & Modeling
Process Intensity
Variance
MultiRes Data
AMR Data
Adaptive Data
Uniform Data
ROI-based Trans
1.Compression
Block-wise Compressor SZ2/ZFP
Global Compressor SZ3
Optimized SZ3 for Multi Reso Data
2.Error-Bounded Post-process
For Blockwise Compressor SZ2/ZFP
For Global Compressor on Multi Reso Data
3. Uncertainty Visualization for Compression

Fig. 3: Overview of our proposed workflow for multi-resolution scientific data compression.

For non-AMR (uniform) simulation, one can also achieve storage efficiency by storing important regions at full resolution and nonessential regions at lower resolution. For example, Previous work [7] proposes using range thresholding to identify ROIs and reduce non-ROI resolution and then using HZ ordering to traverse all the resolution levels to benefit the I/O. However, HZ-ordering prevents us from achieving optimal compression performance because it flattens high-dimensional data into 1D, resulting in the loss of spatial information. At the same time, many previous studies [31, 34, 38] proved that leveraging more spatial information can significantly improve compression performance. To compress the data in 3D, we propose compression-oriented importance-driven storage of uniform data by processing different resolution levels separately (see §III).

The multi-resolution data, including AMR data and the adaptive data generated from uniform data, are hierarchical with different resolutions, with each resolution level holding a different part of the domain, as illustrated in Fig. 2.

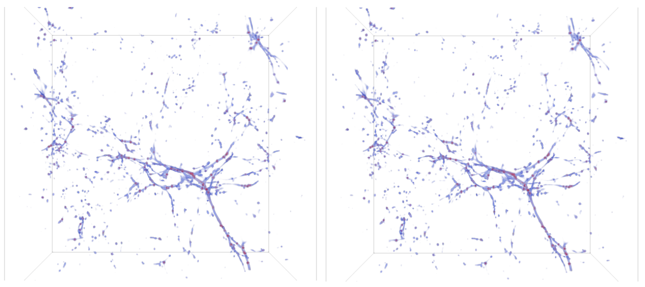
C. Uncertain Data and Visualization

Significant research has been conducted on effective methods for visualizing uncertain scientific data [39–44], as not knowing uncertainty in data can lead to incorrect scientific conclusions. Uncertainty in data often arises from inaccuracies in data acquisition or due to the limitations and incompleteness of measurements available for computational simulations [45]. Similarly, uncertainties in model parameters during scientific simulations introduce variability into the computed solutions [46]. Uncertain data is typically represented by probability distributions at each data point [47–49], in contrast to deterministic data, which assigns a specific value to each point.

The compression techniques, when applied to the original data, can result in a loss of information and introduce error/uncertainty in decompressed data. However, there has been a gap in research regarding treating decompressed data as a form of uncertain data and using uncertainty visualization techniques to explore the effects of compression on scientific datasets. In our work, we apply cutting-edge uncertainty visualization techniques to decompressed data, aiming to provide a clearer understanding of the potential impacts of the compression (see §III-C).

III. Our Proposed Design

This section outlines our proposed workflow for multi-resolution data compression, as shown in Fig. 3. In §III-A,



(a) Original data (b) ROI

Fig. 4: Visualization of the original Nyx cosmology dataset (left) and the ROI (right, 15% of the dataset) extracted using our approach, the SSIM of the two pictures is 0.99995.

We detail our optimization of the SZ3 compressor for multi-resolution data compression (SZ3MR). By employing dynamic padding and an adaptive error-bound approach considering the features of multi-resolution data, we significantly improve SZ3's compression performance on multi-resolution data.

In §III-B, we improve the decompressed data quality from block-wise compressors (e.g., SZ2 and ZFP). We introduce a dynamic, error-bounded post-processing technique that optimally utilizes the spatial information within the dataset. This versatile post-processing method can also improve multi-resolution data compression when using global compressors like SZ3.

In §III-C, we explore the uncertainties introduced by the compression. By integrating a cutting-edge uncertainty visualization solution, we provide users with insights into how compression may impact the data. This exploration aids in understanding and mitigating the effects of compression error.

ROI selection and preprocessing of multi-resolution data. We will first introduce how we convert uniform data into multi-resolution data (referred to as adaptive data) and then detail the preparation of the multi-resolution data (including adaptive data and AMR data) for 3D compression.

We begin by partitioning the original dataset into blocks of size $b \times b \times b$, where b is $2^n, (n > 2)$. Then, following the method of [7], we utilize range thresholding to identify ROIs due to its lightweight and effective characteristics. Specifically, we calculate each block's value range and select the top x percent of the blocks as the ROIs ($x = 50\%$ by default, adjustable for specific applications). Non-ROI blocks are stored at a lower resolution to enhance storage efficiency. For example, as shown in Fig. 4, our range-based ROI selection method effectively extracts the over-density halos from the Nyx cosmology dataset. By selecting just 15% of the dataset, we can capture almost all the halos for the Halo-finder analysis of Nyx [50].

After processing, the adaptive data acquires a data structure similar to AMR data. To compress them in 3D, we diverge from the HZ-ordering method used in [7], which flattens the data to 1D. Instead, we propose compressing each resolution level separately in 3D. However, as illustrated in Fig. 2, each level exhibits many empty regions and an irregular data distribution. To address this, we employ a uniform partitions method, which divides the data into a collection of 3D "unit blocks", as shown in the left part of Fig. 6, for later process and compression.

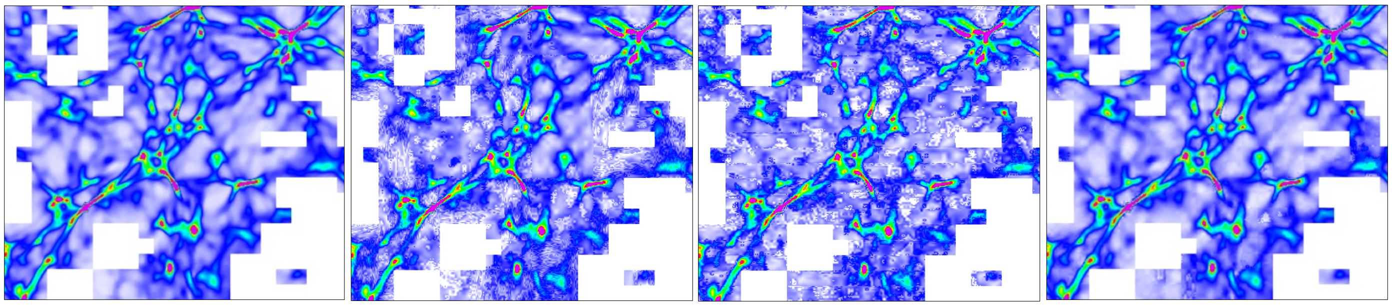

(a) Original data (b) TAC, SSIM=.64, PSNR=117.6 (c) AMRIC, SSIM=.57, PSNR=115.0 (d) Ours, SSIM=.91, PSNR=123.4

Fig. 5: Vis comparison (one $1.5\times$ zoom in 2D slice) of original data and decompressed data produced by TAC's SZ3, AMRIC's SZ3 and our SZ3MR on Nyx's "baryon density" field (fine level). Warmer colors indicate higher values. The CR of TAC, AMRIC, and ours is the same, 163.

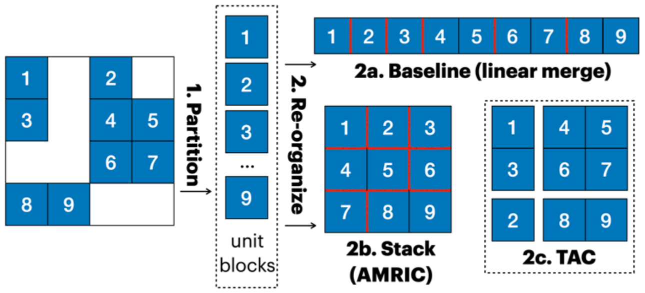


Fig. 6: 2D Example of uniform partition (left, part 1) and different arrangements (the linear merge baseline, stack merge, and TAC) of the unit block (right, part 2). The bold red line indicates unsmooth boundaries because of the merge of non-neighboring blocks.

A. *Improved SZ3 for multi-resolution data (SZ3MR)*

The processed multi-resolution data, however, faces a significant challenge that prevents it from achieving optimized compression quality with the original SZ3 compressor. To overcome these challenges and enhance compression performance, we propose optimizing SZ3. As shown in Fig. 5, our approach achieves much better data quality than SOTA AMRIC [33] and TAC [31] using SZ3. We will now detail the challenge, the limitations of the current approach, and describe our solution.

Challenge: limit compression performance for SZ3 on multi-resolution data. Previous studies have successfully adapted block-wise compressor SZ2/ZFP to handle AMR data. However, optimizing SZ3 for multi-resolution data introduces considerable challenges. A primary concern with SZ3 is the data needs to be partitioned into small unit blocks to leverage 3D compression as mentioned. This disrupts the spatial information's integrity and diminishes data smoothness. When SZ3 confronts partitioned unit blocks from multi-resolution data, the disrupted spatial information can significantly undermine the effectiveness of the interpolation prediction without suitable preprocessing steps.

Limitations of the current solutions. Segmented unit blocks of multi-resolution data can be intuitively linearized (e.g., along z-axis) into a large 3D array before the compression as shown in Fig. 6-**2a**. However, this method can significantly affect the effectiveness of SZ3's interpolation, since the other two dimensions of the merged array (e.g., x and y) are small, compromising prediction accuracy (will be detailed later).

A previous study AMRIC [33] presented an alternative approach by arranging unit blocks into a cubic, instead of linear merging. This method aims to enable more balanced prediction for each dimension. However, stacking unit blocks into cubic forms aggregates blocks that are not adjacent in the original dataset. This leads to rapid changes in data values between these non-neighboring blocks, resulting in misprediction and adversely affecting the precision of SZ3's prediction. As depicted in the bottom mid part of Fig. 6-**2b**, the stacking process introduces more unsmoothness to the data than linear merging does (indicated by the bold red line).

Another work TAC [31] adopts a dynamic strategy, such as using a kD tree, to merge more adjacent unit blocks from the original dataset, aiming to enhance data smoothness and locality. This approach is depicted in the bottom right part of Fig. 6-**2c**. However, TAC does not have an in-situ solution because TAC's preprocessing requires reconstructing the entire physical domain's hierarchy, a complex task that incurs high overhead for in-situ data compression. Also, the challenge of small blocks persists (e.g., block 2 remains small) due to the inherent sparsity of multi-resolution data. Moreover, because the merged blocks vary in shape, TAC must compress the merged blocks with different shapes separately, which brings encoding overhead.

Improvement 1: Better prediction via Padding. We propose to still linearize unit blocks like baseline to avoid the issue of AMRIC and TAC. However, compared to the baseline, we introduce a padding strategy aimed at enhancing SZ3's prediction accuracy and compression performance for small unit blocks. This strategy is specifically designed to improve prediction performance for the two smaller dimensions of the large linearized array. To demonstrate the process and limitations of SZ3 interpolation for small blocks, we present an example using 1D linear interpolation. Although simplified, this example embodies the core principles applicable to more complex scenarios like cubic and 3D interpolation.

Consider a dataset in one dimension containing N elements. SZ3's interpolation approach happened level by level and begins by predicting the first data point (d_1) using an initial value of 0 for level 0. Then, for level 1, d_1 is used to predict the final data point (d_N). The interpolation process then proceeds in steps size S of 2^n, satisfying the condition:

$$2^n < N - 1, \quad n \in \mathbb{N}$$

with n decreasing each level. Each S^{th} point, not yet predicted, is interpolated from adjacent steps (e.g., predict d_{s+1} using d_1

and d_{2s+1}). Points outside the interpolation range are handled through extrapolation.

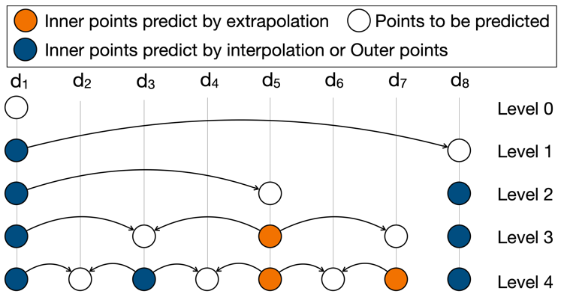

Inner points predict by extrapolation
Points to be predicted
Inner points predict by interpolation or Outer points
d1 d2 d3 d4 d5 d6 d7 d8
Level 0
Level 1
Level 2
Level 3
Level 4

Fig. 7: Interpolation example of 8 data points.

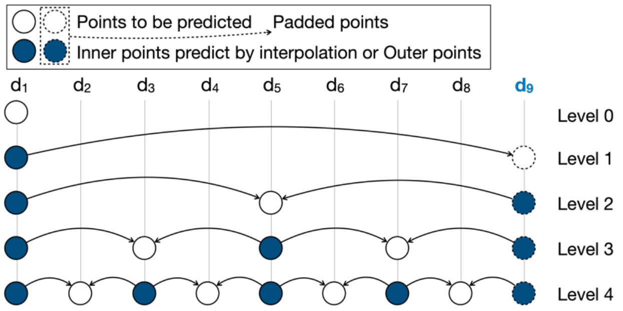

Points to be predicted
Padded points
Inner points predict by interpolation or Outer points
d1 d2 d3 d4 d5 d6 d7 d8 d9
Level 0
Level 1
Level 2
Level 3
Level 4

Fig. 8: Interpolation example of 9 data points with one padded point.

For small unit blocks partitioned from multi-resolution data, typically of size 2^n, we examine a scenario with a block size of 8, as shown in Fig. 7. Initially, 0 is used to predict d_1, and d_1 is used to predict d_8 for levels 0 and 1. At level 2, with an interpolation step size of 4, we aim to use d_1 and d_9 to interpolate d_5. However, d_9 does not exist, forcing us to depend solely on d_1 to extrapolate d_5, resulting in limited accuracy. Similarly, at level 3, only d_5 is available for extrapolating d_7. After completing the interpolation, it is clear that except for the outer values d_1 and d_8, 2 out of 6 inner points undergo undesired extrapolation (highlighted in orange). If the block size is 16, this sub-optimal prediction affects 3 out of 14 inner points. Since points predicted at earlier levels are used to predict other points at subsequent levels, the inaccuracies significantly compromise overall compression performance.

To address this issue, we propose the application of padding to the two smaller dimensions of the merged array to enhance prediction performance and eliminate sub-optimal predictions. Given that the multi-resolution data typically adhere to a block size of 2^n, padding merely requires a single layer of data points to each of the two smaller dimensions (i.e., padding one point for the 1D array), thus introducing acceptable data size overhead. As demonstrated in Fig. 8, for a block size of 8, padding an additional point d_9 effectively eliminates all sub-optimal predictions for inner data points. It is also important to determine the pad value, we test using constant, linear, and quadratic extrapolation. After many experiments, we find that the linear extrapolation overall produce the best prediction performance, especially for the relatively smooth dataset.

On the other hand, while padding can enhance prediction accuracy, it also incurs a size overhead to the data. This overhead is quantified by $(u+1)^2/u^2$, where u denotes the unit block size. With $u = 4$, the overhead amounts to 56%. In this scenario, padding improves the prediction for 2 out of 3 inner points, but the overall performance gain remains constrained. Moreover, the increased dataset size introduces additional time overhead for compression. Consequently, we opt to implement the padding approach only when $u > 4$.

As illustrated in Fig. 18 in §IV-C, our padding strategy, denoted by the curve labeled "***Ours (pad)***", significantly enhances the rate-distortion trade-off (PSNR vs. compression ratio) relative to both the AMRIC and baseline. Moreover, our method outperforms the offline-only solution TAC, especially at higher compression ratios.

Improvement 2: Use of adaptive error-bound. We further improve SZ3's performance on multi-resolution data by employing an adaptive error-bound for each interpolation level. This method accounts for the fact that data points predicted at earlier levels influence subsequent-level predictions. For example, as illustrated in Fig. 8, point d_9 is used for predicting d_5, d_7, and d_8, highlighting the need for smaller error bounds at early interpolation levels to boost compression efficiency.

Although the original SZ3 offers an adaptive error-bound strategy, its coarse granularity limits optimization. Inspired by the QoZ approach [51], we implemented a more refined adaptive error-bound strategy for each interpolation level:

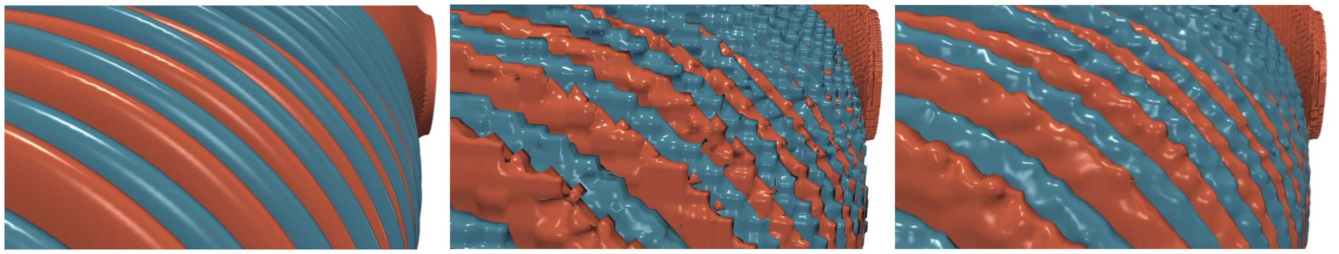
$$eb_l = eb \cdot \left(\min(\alpha^{maxlevel-l}, \beta)\right)^{-1}.$$

Unlike QoZ, which uses sampling and trial-and-error to select α and β—a process that introduces overhead—we leverage the characteristics of multiresolution data for a more assertive strategy, setting α to 2.25 and β to 8. These parameters are larger than those used by QoZ. This method accelerates the reduction of error bounds for early interpolation levels, particularly for data shapes resulting from linear merges and padding, typically featuring two smaller and one larger dimension (e.g., $17\times17\times8192$). The total interpolation level is low for the two small dimensions, necessitating higher α and β to attain small enough error bounds for the initial interpolation levels. Extensive offline experiment shows that $\alpha = 2.25$ and $\beta = 8$ deliver the best compression performance in most scenarios.

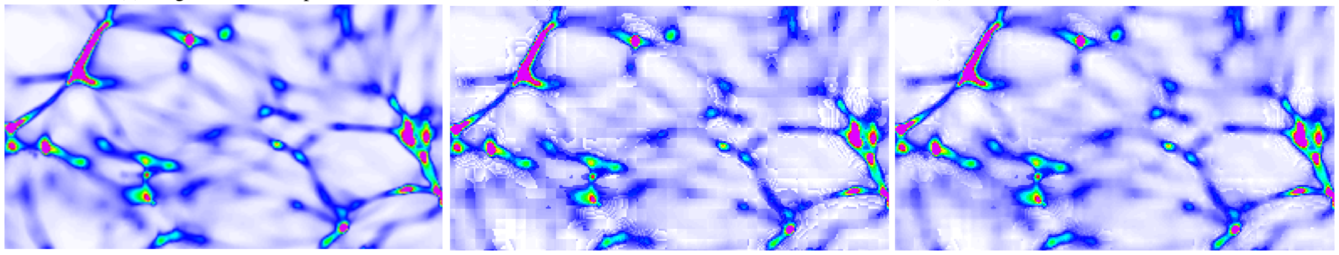
As illustrated in Fig. 18 in §IV-C, our approach with padding and adaptive error bound (denoted by the curve "***Ours (pad+eb)***") can further improve the compression performance. And, as shown in Fig. 5, after the two-step optimization, our approach notably improves the overall compression and visualization quality in comparison to the AMRIC and TAC.

B. Error bounded Adaptive post processing

For block-wise scientific compressors like SZ2 and ZFP, previous studies have made significant strides in optimization for multi-resolution data. However, there remains scope for enhancement. Block-wise compressors often produce limited compression quality and are prone to compression artifacts [52], as shown in Fig. 9b. To address these issues, we introduce a fast and effective post-processing solution that enhances the data quality of block-wise compressors. As shown in Fig. 9c, our post-



(a) Original data, WarpX (b) ZFP, SSIM=.72, PSNR=75.5 (c) Processed ZFP, SSIM=.79, PSNR=78.1



(d) Original data, Nyx (e) SZ2, SSIM=.76, PSNR=116.0 (f) Processed SZ2, SSIM=.85, PSNR=118.1

Fig. 9: Visual comparison (iso-surface and 2D slice) of original data, decompressed data produced by ZFP and SZ2, and after our post-process on WarpX's "Ez" field and Nyx's "density" field. The CR is 139 and 143, respectively.

TABLE I: Comparison of data quality (in PSNR) of original decompressed data from ZFP, decompressed data processed by image smooth/denoise filters, and our solution.

	Decomp. data	Median Filter	Gaussian Blur	Anisotropic Diffusion	**Ours**
PSNR	80.5	67.2	71.6	74.4	**82.9**

processing solution significantly reduces compression artifacts and errors. We will now discuss the challenges posed by block-wise compressors and detail our post-processing approach.

Challenge: low quality of block-wise compression. The low-quality issue of block-wise compressors is mainly attributed to their block-wise nature of dividing the dataset into small blocks (e.g., $4 \times 4 \times 4$) before the compression. Specifically, the partition can cause each block to lose spatial information of its neighboring blocks, losing the opportunity for better compression quality. Furthermore, the separate processing of blocks disrupts the coherence of features that span across the block boundaries, leading to a degradation in data visualization and quality. It is important to note that, for SZ2, the issue with blocking artifacts will be more severe for multi-resolution data than for uniform-resolution data. This is because, for multi-resolution data, SZ2 needs to reduce its compression block size from $6 \times 6 \times 6$ to $4 \times 4 \times 4$ to achieve optimal performance [33], thus leading to more artifacts due to the smaller block size.

Limitation of the image processing filters. Numerous image smoothing and denoising techniques, such as Anisotropic Diffusion, Gaussian Blur, and Median Filter, are widely used for post-processing. However, their effectiveness often diminishes when applied to decompressed data from error-bounded scientific compressors. This shortfall arises because these filters are designed for lossy image compressors like JPEG. When used on scientific data, they can over-smooth the data, leading to significant detail loss and a marked reduction in PSNR, as illustrated in TABLE I. This issue stems from the filters' lack of

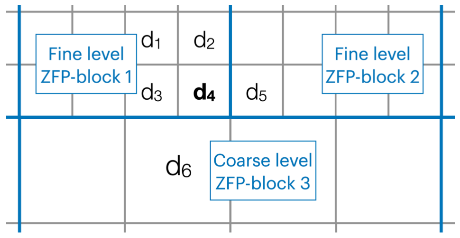


Fig. 10: Example of the multi-resolution data gird when using ZFP compressor, the gray grid indicates data points and the bold blue box indicates 4×4 blocks partitioned by ZFP.

consideration for the error-bounded nature of the decompressed data, resulting in a notable deviation from the original dataset.

Improvement: Use post-process to improve the compression quality. To tackle this challenge, we introduce an adaptive post-processing technique specifically designed for error-bounded scientific lossy compressors. This method starts by applying Bézier curves to exchange spatial information overlooked during compression among data blocks. It then utilizes the error-bound properties of the decompressed data and dynamically adjusts the processing intensity. This strategy markedly improves both visualization quality (e.g., Structural Similarity Index Measure (SSIM)) and data quality (e.g., PSNR) of the decompressed output. Particularly, it excels at mitigating blocking artifacts, a common drawback of block-wise compression.

We opt for Bézier curves due to their ability to smooth transitions between points, effectively mitigating discontinuities or artifacts introduced during compression. Additionally, Bézier curves are computationally efficient and highly parallelizable, making them suitable for post-processing needs where computational speed is important.

A 2D example is illustrated in Fig. 10 using ZFP. Data points are partitioned into 4×4 blocks for compression, isolating them from points in other blocks. We aim to utilize Bézier curves to exchange spatial information between adjacent data blocks,

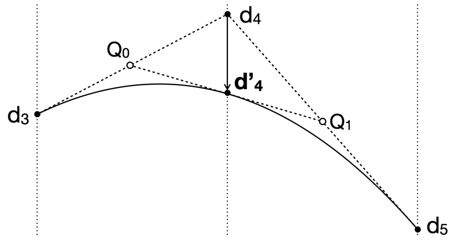


Fig. 11: Example of the Bézier curves for $t = 0.5$, Q_0 and Q_1 are the midpoints of d_3d_4 and d_4d_5. d'_4 is obtained by B(0.5), mid of Q_0Q_1.

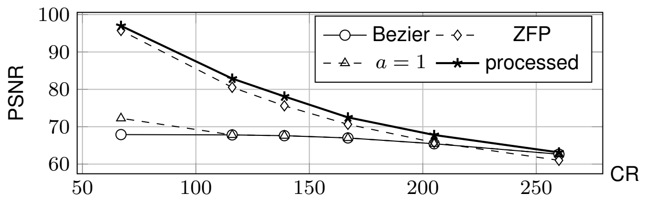


Fig. 12: Rate-distortion comparison of different post-process approaches on WarpX using ZFP.

thereby enhancing data quality. Specifically, for decompressed data at the boundary (e.g., d_4), we can leverage its neighboring point d_5 along the x-direction from another block to improve its quality. This is achieved by constructing a quadratic Bézier curve with d_3, d_4, and d_5, where d_3 and d_5 are the start and end points, respectively, and d_4 serves as the control point. The curve is defined as:

$$B(t) = (1-t)^2 d_3 + 2(1-t)td_4 + t^2 d_5 \quad \text{for} \quad 0 \le t \le 1$$

with t being the parameter ranging from 0 to 1. As t progresses from 0 to 1, the Bézier curve formula generates points tracing the curve's path from d_3 to d_5. The adjusted d'_4 is derived at $t = 0.5$ ($d'_4 = B(0.5)$) as shown in Fig. 11. This Bézier curve approach is applied for each dimension separately and can be easily extended to multi-resolution scenarios. For instance, in the y direction, d_2 and d_6 would be used to process d_4.

Leverage the error-bounded feature in decompressed data. However, neglecting the error-bounded nature of decompressed data can significantly reduce the quality of the process, as illustrated in Fig. 12. Sole dependence on the Bézier curve (represented by "***Bezier***") severely impacts the quality of ZFP decompressed data, mirroring the limitations encountered with image filters. For an error-bounded compressor, the decompressed data point d_4 must stay within the error bounds eb of the original data o_4. Therefore we have: $o_4 \in [d_4 - eb, d_4 + eb]$. This condition suggests that when processing d_4 to d'_4, d'_4 should fall within $[d_4 - eb, d_4 + eb]$, guaranteeing:

$$d'_4 = \max(\min(B(0.5), d_4 + eb), d_4 - eb)$$

This formula ensures that d'_4 remains within the error limits, maintaining the decompressed data's integrity.

Further improve the process quality using dynamic limit/intensity. Nevertheless, utilizing the error-bound information is still insufficient for achieving optimal post-processing quality. To enhance the data quality, we must adaptively limit the actual error bound used in the post-process, eb' ($eb' = a \cdot eb$, $a < 1$), making it smaller to control the post-process intensity.

To clarify the necessity of a smaller eb', we'll examine a sample scenario. Assume the original data o is larger than the decompressed data d. Initially, as illustrated at the top of Fig. 13 (situation 1), there might be cases where the Bézier curve predicts in the opposite direction ($B(0.5) < d$). In such instances, a small a helps prevent d' from deviating excessively from original data o. Secondly, as depicted in the middle of Fig. 13 (situation 2), when the Bézier curve accurately predicts but overshoots beyond the original data ($B(0.5) > o$), a smaller a helps ensure d' remains closer to o. However, if the Bézier curve correctly predicts within the bounds ($o < B(0.5) < d$), overly reducing a prevents d' from getting closer to o, as shown in the bottom of Fig. 13 (situation 3).

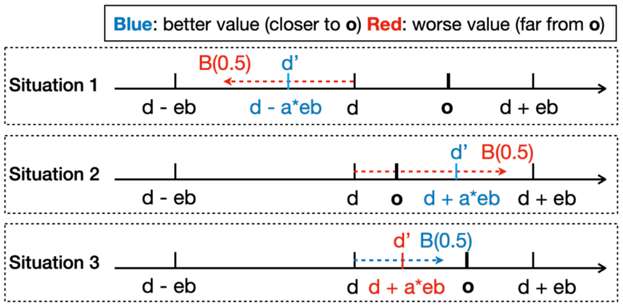


Fig. 13: Example of the impact of setting smaller limit/intensity of the post-process under different situations, blue color indicates better post-process outcome and red color indicates worse.

We seek to maximize the gain from post-processing:

$$h \cdot (|e| - |e'|) - (1-h) \cdot (|e'| - |e|) \quad 0 \le h \le 1$$

where h denotes the rate at the Bézier curve does not make opposite predictions. e and e' denote the compression errors before and after post-processing, which can be reformulated as:

$$\underset{a}{\text{maximize}} \quad h \cdot (|o-d| - |o-d'|) - (1-h) \cdot |d'-d|,$$

by adjusting a, under the constraint:

$$d' = \max(\min(B(0.5), d + a \cdot eb), d - a \cdot eb) \quad 0 \le a \le 1.$$

However, finding the optimal a analytically is not feasible due to the presence of absolute values in the objective function and the piece-wise definition of d'. Moreover, obtaining necessary parameters before compression—like the hit rate h also incurs additional costs. Thus, we employ a sampling-based numerical optimization approach to iteratively find the optimal a.

Now we present how the optimal a is dynamically determined through a lightweight compression sampling process. Extensive experimentation across various datasets has enabled us to refine our selection of the best candidate parameters for our algorithm. Specifically, for SZ2, a_{sz} is narrowed to the set $\{0.05, 0.1, 0.015, ..., 0.45, 0.5\}$, and for ZFP, a_{zfp} is set to $\{0.005, 0.01, 0.015, ..., 0.05\}$. These values can achieve optimal or near-optimal performance in most cases, while also being practical for evaluation. The candidate for ZFP is smaller due to its underestimation characteristic, which leads to a smaller max real compression error than the given error bound.

Our methodology starts by sampling i^3 data blocks of size $(j \times \text{blocksize})^3$, where blocksize refers to the compressor's block-wise compression size. We aim for a sampling rate below 1.5%, sufficient for identifying the optimal a with minimal overhead.

TABLE II: Rate-distortion comparison of original decompressed data and our post-process approach on WarpX using SZ2.

CR	*273*	*207*	*153*	*126*	*104*	*62*	*34*
PSNR-SZ2	67.8	72.8	79.6	84.8	90.0	101.9	114.4
PSNR-Proc'ed	**69.8**	**74.6**	**81.1**	**86.2**	**91.2**	**102.6**	**114.9**

Then, for each dimension, we utilize stochastic gradient descent (SGD) to find the optimal a from the candidates that minimize the overall norm2 compression error.

Fig. 12 shows that our post-processing with dynamic limit/intensity (denoted by ***"Process"***) significantly enhances ZFP decompressed data quality. The ***"a=1"*** curve represents performance without the dynamic limit, showing low performance. Fig. 9 clearly illustrates how our post-processing significantly enhances data quality through visualization. In addition to ZFP, TABLE II illustrates our approach's effectiveness in improving SZ2's compressed data quality. Furthermore, our post-processing can also improve the data quality for global compressors like SZ3 in multi-resolution scenarios. Because multi-resolution data need to be partitioned before compression, as discussed in §III-A. Detailed performance outcomes will be shown in §IV-B.

C. Uncertainty Visualisation for Compression

In this work, we employ uncertainty visualization to examine the effects of compression on data. Specifically, we explore how compression errors influence the positions of isosurfaces, which are highly sensitive to errors and can be significantly altered by compression-related inaccuracies. This sensitivity provides a valuable perspective for deepening our understanding of compression's impact on data.

Multiple previous contributions have studied the impact of uncertainty in data on isosurface visualization [40, 49, 53, 54]. In this work, we leverage the probabilistic marching cubes idea [49, 53, 55] to gain insight into the effect of compression errors on isosurface positions. The probabilistic marching cubes algorithm models per-voxel error as a probability distribution to derive the spatial probability distribution of isosurfaces. Our primary objective is to utilize the error distribution of decompressed data to analyze isosurface uncertainty. In both ZFP and SZ compressed data, errors follow a normal distribution [56], especially when the error bound is large [57].

Thus, we focus on the normal distribution in this work, given our focus on cases with larger error bounds. Modeling uncertainty per voxel as a normal distribution involves determining the mean and variance of the uncertainty (compression error) per voxel, which is challenging because the error information is lost after compression. However, as illustrated in Fig. 3, we sample the compression error during the compression process for post-processing needs. This sampled compression error can also be used to obtain the mean and variance of the error with minimal overhead by reusing the information.

Isovalue related variance. Given the fact that the data points close to the isovalue are more likely to be considered for the isosurface construction. When computing the variance, we focus on data points with values near the isovalue instead of using all the sampled points. This approach allows for a more accurate variance calculation for the given isovalue, as the compression error could depend on the data value.

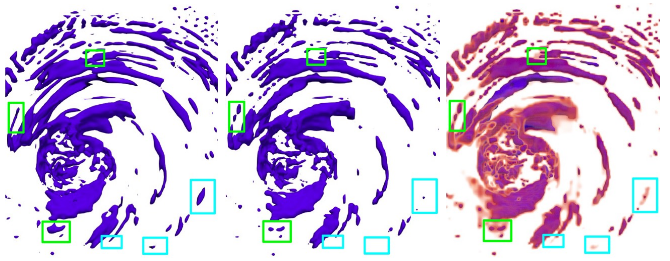


Fig. 14: Vis of original data, decompressed data (generated by our workflow using ZFP, CR = 240), and decompressed data with uncertainty, cyan/green box highlights the missing/cracking isosurface.

Having characterized uncertainty with error distribution near isovalue in decompressed data, we apply the probabilistic marching cubes techniques [49, 53] to gain insight into spatial uncertainty in isosurface arising from compression. Fig. 14 illustrates how uncertainty visualization helps in understanding the error in decompressed data. Specifically, Fig. 14a and Fig. 14b visualize the isosurfaces for the Hurricane dataset [58] extracted from the original data and decompressed data, respectively. Fig. 14c visualizes uncertainty in red using our approach for the decompressed data. The boxes in Fig. 14 highlight the topological features that are missed/broken in visualization without uncertainty (Fig. 14b) but are successfully recovered by the one with uncertainty visualization (Fig. 14c). For example, the cyan boxes illustrate features that disappear from the original data in Fig. 14a because of the compression errors, but whose potential presence is denoted by the red regions in Fig. 14c. Thus, the visualization of spatial uncertainty mitigates data misrepresentation arising from compression errors.

This phenomenon occurs because the isosurface is prone to being pruned due to compression errors, attributed to its binary nature. A moderate compression error can cause the isosurface to disappear completely; for example, if all the corresponding data values fall below the isovalue after compression. On the other hand, the isosurface uncertainty visualization, as described in [49, 53], employs a more informative approach. It enhances visualization by incorporating the uncertainty (i.e., error distribution) of the decompressed data, rather than solely considering the decompressed data itself.

IV. Experimental Evaluation

A. Experimental Setup.

Applications and datasets. We conducted both in-situ and offline experiments. For the in-situ experiments, we selected two real-world applications: the Nyx cosmology simulation [59] and the WarpX electromagnetic simulation [4, 60]. These were conducted on the Bridges-2 [61, 62], where each node is equipped with two AMD EPYC 7742 CPUs and 256 GB RAM. Our experiments utilized 128 cores. Nyx serves as an AMR application, fully supporting AMR features. WarpX is utilized for experiments involving adaptive data (derived from uniform-resolution data) as WarpX does not yet fully support AMR.

In addition to in-situ evaluation, we also evaluated our solution using five different offline datasets from four distinct applications to demonstrate our solution's broad applicability. The offline evaluation included multi-resolution data with different resolution levels and density (density refers to the proportion of data within the entire domain) and uniform-resolution data as specified in TABLE III. Specifically, we tested the Rayleigh-Taylor (denoted as"RT") dataset generated by the IAMR fluid dynamics simulation [63], the S3D combustion simulation, the Hurricane Isabel dataset [64], and two additional Nyx datasets (denoted as "T2" and "T3") from different timesteps.

TABLE III: Our tested datasets

Dataset	Property	(Size, Density) per Level Fine to Coarse	Per-Timestep Data Size
Nyx-T1	In-situ, AMR 2 levels	fine: $(512^3, 18\%)$ coarse: $(256^3, 82\%)$	3.1 GB
WarpX	In-situ, Adpt 2 levels	fine: $(256^2 \times 2048, 50\%)$ coarse: $(128^2 \times 1024, 50\%)$	6.3 GB
RT	Offline, AMR 3 levels	finest: $(512^3, 15\%)$ medium: $(256^3, 31\%)$ coarse: $(128^3, 54\%)$	2 GB
Nyx-T2	Offline, AMR 2 levels	fine: $(512^3, 58\%)$ coarse: $(256^3, 42\%)$	7.1 GB
Hurri	Offline, Adpt 2 levels	fine: $(500^2 \times 100, 35\%)$ coarse: $(250^2 \times 50, 65\%)$	1.1 GB
Nyx-T3	Offline, Uni	$(512^3, 100\%)$	10 GB
S3D	Offline, Uni	$(512^3, 100\%)$	11 GB

Comparison baseline. We evaluate our SZ3MR on both AMR data and adaptive data generated from uniform-resolution data. In terms of AMR data, we benchmark our improved approach against AMRIC's SZ3 (referred to as "AMRIC-SZ3" [33]) and TAC's SZ3 (referred to as "TAC-SZ3", only for offline evaluation as it lacks an in-situ option [31]), and the original SZ3 (denote as "Baseline-SZ3"). For adaptive data, our improved SZ3 is evaluated against the original SZ3 as TAC and AMRIC do not offer SZ3 implementation for adaptive data.

Further, we first conduct offline evaluations on our adaptive post-processing technique utilizing both SZ2 and ZFP across multi-resolution and uniform datasets. It is important to note that we employ AMRIC's SZ2 for multi-resolution data due to its superior compression capabilities compared to zMesh [30] and TAC. Additionally, we have integrated our post-processing technique into the AMR application Nyx for in-situ evaluation, demonstrating that our approach significantly enhances the data quality of AMRIC-SZ2 through post-processing.

B. In-situ Evaluation

In-situ Evaluation on AMR data compression. As illustrated in Fig. 15, our SZ3MR (with "pad" and "eb" detailing the performance of our two-step optimization) outperforms both the baseline and AMRIC across both refinement levels on Nyx, particularly at higher compression ratios. However, at the coarse level and with smaller compression ratios, our SZ3MR's performance is slightly worse than the baselines. This is due to the high padding overhead given the smaller unit block size at the coarse level, as discussed in §III-A.

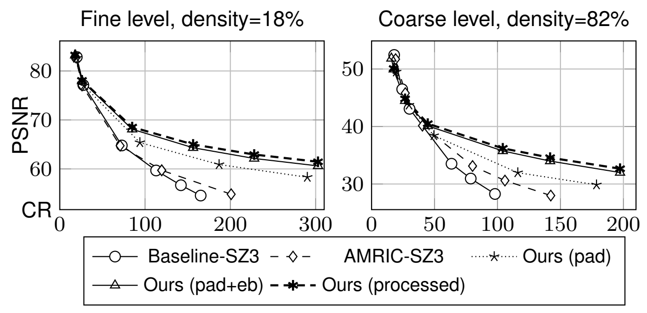


Fig. 15: Rate-distortion comparison on AMR data of our SZ3MR approaches and baselines using Nyx AMR simulation (Nyx-T1).

We also compare the overall output time of our SZ3MR with that of AMRIC on Nyx. The overall output time consists of (1) pre-processing (i.e., collecting data to the compression buffer) and (2) compression and writing the compressed data to the file system. As shown in TABLE IV, although our compression speed is slightly lower than AMRIC because of the padding overhead, our SZ3MR achieves a faster total output speed in both large and small error-bound settings. The improvement is primarily attributed to our more efficient pre-processing stage, as AMRIC's stacking process is more complex and computationally intensive, requiring significant data rearrangement.

TABLE IV: Output time of AMRIC and our SZ3MR on Nyx-T1.

EB_abs	Time (Sec)	Pre-process	Comp. & Writing	Total Time
5.4E+9 (big)	AMRIC	1.22	**1.62**	2.85
	Ours	**0.49**	1.69	**2.18**
2.7E+8 (small)	AMRIC	1.23	**2.30**	3.52
	Ours	**0.47**	2.38	**2.85**

Our post-processing solution, as shown in TABLE V, significantly improves the quality of decompressed data for AMRIC-SZ2 on Nyx simulation at both resolution levels, with the degree of improvement being notably greater at higher compression ratios. Furthermore, as outlined in §III-A and illustrated by the "**Ours (processed)**" curve in Fig. 15, our post-processing also improves the data quality of SZ3 on multi-resolution data due to the need for partition. However, the improvement is less substantial than those achieved with block-wise compressors SZ2/ZFP. This is because the partition size (unit block size) for multi-resolution data is larger than the block sizes used by SZ/ZFP (16 vs. 4), resulting in less room for improvement.

TABLE V: Rate-distortion comparison of decompressed data and our post-process solution on both levels of Nyx-T1 using AMRIC-SZ2.

Fine	CR	*270*	*165*	*113*	*73*	*28*
	PSNR-AMRIC-SZ2	48.1	54.6	59.7	64.8	77.1
	PSNR-Post-SZ2	**50.1**	**56.9**	**61.8**	**66.5**	**77.6**
Coarse	CR	*128*	*98*	*63*	*36*	*24*
	PSNR-AMRIC-SZ2	25.3	28.3	33.5	40.7	46.5
	PSNR-Post-SZ2	**27.8**	**31.0**	**36.0**	**41.9**	**46.9**

In-situ evaluation on adaptive data compression. Regarding adaptive data derived from uniform data, Fig. 17 (left) shows our

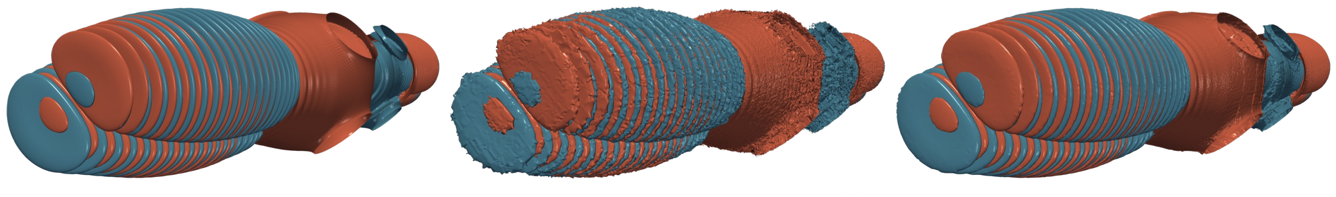

(a) Original data (b) SZ3, CR=147, SSIM=0.662, PSNR=75.5 (c) Ours, CR=147, SSIM=0.904, PSNR=86.9

Fig. 16: Visual comparison (iso-surface) of original data and decompressed data produced by original SZ3 and our SZ3MR on WarpX ("Ez" field).

in-situ experiments with WarpX, demonstrating that our SZ3MR outperforms the original SZ3 baseline in most cases, except at lower compression ratios. It's important to note that AMRIC-SZ3 and TAC-SZ3 were not compared in this context due to their lack of support for adaptive data. Furthermore, as shown in Fig. 16, our SZ3MR notably enhances the compression quality (in terms of both PSNR and SSIM) and reduces visualization artifacts, offering a clear improvement over the baseline.

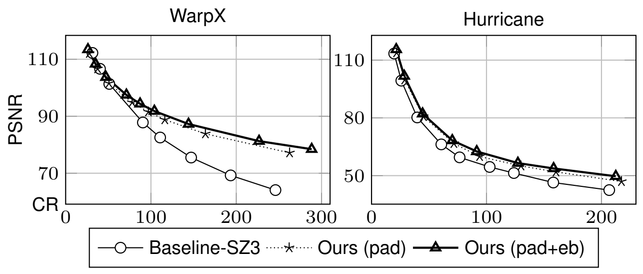


Fig. 17: Rate-distortion comparison on adaptive data of our SZ3MR and baselines using WarpX (in-situ) and Hurricane (offline) datasets.

C. *Offline Evaluation*

SZ3-MR on multi-resolution data. As illustrated in Fig. 18, our method, after the two-step optimizations, outperforms all three baselines for both the Nyx-T2 and RT AMR datasets. It's observed that the AMRIC solution underperforms compared to the baseline on the RT dataset. We attribute this to the RT dataset having an additional refinement level compared to Nyx-T2, resulting in sparser data and more unsmooth boundaries due to merging non-adjacent blocks, leading to increased mispredictions. Also, note that when the compression ratio is low, TAC yields slightly better performance than our solution on Nyx-T2, but its advantage is almost negligible on the RT dataset. This is because the RT has a smaller data size for each resolution level. Since TAC must compress the processed blocks with different shapes separately for each level (as mentioned in §III-A), the smaller data size will severe the encoding overhead issue of TAC and lead to a low compression ratio.

Regarding adaptive data derived from uniform data, as shown in Fig. 17 (right), our adaptive error-bound solution offers limited enhancements until the high compression ratio. However, our padding technique consistently delivers significant improvements over the baseline across all compression ratios in the Hurricane dataset. We attribute this performance to the dataset's relative sparsity (i.e., numerous zero points), which enhances compressibility and offsets the padding overhead.

We also evaluated SZ3MR using application-specific power spectrum analysis on the Nyx-T2 dataset (see [31, 65] for more

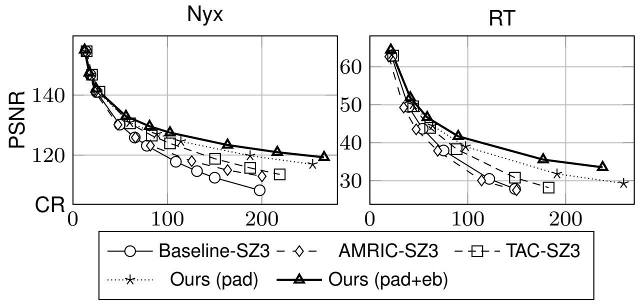


Fig. 18: Rate-distortion comparison on offline AMR data of our SZ3MR and baselines using Nyx-T2 and RT datasets.

details on power spectrum analysis in Nyx). We compared the power spectrum $p'(k)$ of decompressed data with the original $p(k)$. Typically, a maximum relative error threshold of 1% is considered acceptable for all $k < 10$. Table VI shows that under the same compression ratio, SZ3MR achieves a lower power spectrum error (including both the max and average errors for all $k < 10$) compared to all three baselines. Specifically, SZ3MR reduces the max power spectrum error by 75%, 76%, and 73%, and reduces the average error by 74%, 60%, and 62% compared to the original SZ3, AMRIC, and TAC, respectively, at the same compression ratio.

TABLE VI: Max and average power spectrum error comparison of our SZ3MR and baselines on Nyx-T2 under same CR for all $k < 10$.

	Baseline-SZ3	AMRIC-SZ3	TAC-SZ3	**Ours(pad+eb)**
Avg Rel Error	8.8E-03	5.7E-03	6.0E-03	**2.3E-03**
Max Rel Error	2.7E-02	2.8E-02	2.5E-02	**6.7E-03**

Post process for multi-resolution data. As illustrated in TABLE VII, our post-process approach enhances the data quality in terms of PSNR for both the Hurricane and RT datasets across all compression ratios, with both SZ2 (optimized by AMRIC for multi-resolution data) and ZFP. Note that PSNR improvement is relatively modest at low compression ratios (e.g., under 30) because a lower CR indicates higher decompressed data quality, leaving limited room for improvement. When the compression ratio is low, our dynamic post-process approach can apply a conservative degree of post-processing intensity to ensure the original data quality remains uncompromised.

Post process for uniform resolution-data. Our post-processing method, as previously mentioned, demonstrates broad applicability, making it suitable for processing both uniform-resolution data and multi-resolution data from block-wise compressors. As shown in TABLE VIII, and in alignment with our previous observations, our post-processing consistently enhances the data quality of the original SZ2 and ZFP outputs

TABLE VII: Rate-distortion comparison of original decompressed data and our post-process approach on multiresolution datasets Hurricane and RT using ZFP and AMRIC-SZ2.

RT	ZFP	CR	*184*	*143*	*118*	*72*	*43*	*27*
		PSNR-Ori	34.2	41.2	45.3	54.1	63.6	74.2
		PSNR-Post	**36.7**	**43.9**	**47.7**	**55.4**	**64.2**	**74.5**
	SZ2	CR	*257*	*180*	*122*	*75*	*40*	*22*
		PSNR-Ori	35.2	40.5	45.8	53.3	64.8	78.2
		PSNR-Post	**37.2**	**42.5**	**47.6**	**54.6**	**65.6**	**78.6**
Hur	ZFP	CR	*240*	*147*	*94*	*64*	*27*	*18*
		PSNR-Ori	40.1	43.7	47.8	52.6	68.5	80
		PSNR-Post	**42.1**	**45.6**	**49.5**	**53.8**	**69.2**	**80.5**
	SZ2	CR	*170*	*121*	*108*	*73*	*38*	*23*
		PSNR-Ori	41.9	44.3	45.3	49.9	62.4	75.8
		PSNR-Post	**43.2**	**45.9**	**47**	**51.5**	**63.3**	**76.4**

for both the uniform resolution datasets Nyx-T3 and S3D.

TABLE VIII: Rate-distortion comparison of original decompressed data and our post-process approach on uniform resolution dataset S3D and Nyx-T3 using ZFP and SZ2.

S3D	ZFP	CR	*138*	*106*	*87*	*70*	*55*	*32*
		PSNR-Ori	48.4	62.7	73.4	83.7	94	115.6
		PSNR-Post	**51**	**65.9**	**75.4**	**84.8**	**94.7**	**115.9**
	SZ2	CR	*229*	*180*	*135*	*81*	*59*	*40*
		PSNR-Ori	64.9	67.7	72.8	90.8	104.2	115.8
		PSNR-Post	**67.5**	**70.2**	**74.7**	**91.4**	**104.4**	**116.0**
Nyx	ZFP	CR	*149*	*116*	*73*	*56*	*41*	*22*
		PSNR-Ori	107.3	112.1	120.5	124.8	129.2	138.3
		PSNR-Post	**109.3**	**114.2**	**122.8**	**126.8**	**130.9**	**139.2**
	SZ2	CR	*214*	*143*	*94*	*53*	*34*	*13*
		PSNR-Ori	112.5	116	119.7	124.8	128.9	140.3
		PSNR-Post	**114.5**	**118.1**	**121.7**	**126.7**	**130.6**	**141.3**

Post process overhead. Our post-processing solution is efficient and highly parallelizable, as mentioned in §III-B, thereby introducing minimal overhead to the compression workflow. We employ OpenMP to accelerate our post-processing approach and assess its overhead using both SZ2 and ZFP, which are also optimized with OpenMP. It is important to note that using OpenMP with SZ2 can lead to a lower compression ratio due to the embarrassingly parallel. Thus, we have also conducted evaluations using the serial SZ2. As demonstrated in the last column of TABLE IX, our post-processing introduces an overhead of only about 1.3% for serial SZ2 and 3.5% for SZ2/ZFP with OpenMP acceleration, respectively, under various compression ratios, utilizing 64 cores.

TABLE IX: Execution time of original compression workflow (columns 1 and 2) and post-processing (columns 3 and 4) on S3D.

	CR	1. I/O	2. Comp& Decomp	3. Sample + Model	4. Process	5. Ori (c1+c2)	6. Extra (c3+c4)	**Overhead (c6/c5)**
ZFP (OpenMP)	Small	0.77	1.403	0.009	0.050	2.175	0.059	**0.027**
	Mid	0.79	1.081	0.010	0.051	1.876	0.061	**0.033**
	Large	0.80	0.948	0.012	0.049	1.749	0.061	**0.035**
SZ2 (OpenMP)	Small	0.78	0.411	0.010	0.034	1.190	0.044	**0.037**
	Mid	0.84	0.371	0.008	0.034	1.208	0.042	**0.035**
	Large	0.79	0.283	0.007	0.033	1.072	0.039	**0.037**
SZ2 (Serial)	Small	0.82	5.199	0.031	0.042	6.015	0.073	**0.012**
	Mid	0.81	4.585	0.028	0.041	5.399	0.069	**0.013**
	Large	0.85	3.637	0.021	0.039	4.485	0.061	**0.013**

Specifically, the original compression workflow (columns 1 and 2) includes reading the original file, compression and decompression, and writing the decompressed file. Our post-processing involves sampling, (de)compressing the sampled data, modeling the optimal parameter before compression (column 3), and post-processing after decompression (column 4). The efficiency of our approach is due to the high parallel efficiency of the Bézier curve and our effective implementation. As detailed in column 3 of TABLE IX, our sampling and modeling process incurs very low overhead for SZ2/ZFP with OpenMP. For serial SZ2, the sampling and modeling times are higher due to slower (de)compression speed, which, further minimizes our relative overhead. Moreover, our post-processing speed is notably fast, as shown in column 4. Note that the post-processing speed for ZFP is slower due to its smaller block size compared to SZ2, which increases processing intensity.

V. Conclusion and Future Work

This paper introduces a workflow for multi-resolution data compression, applicable to both uniform and AMR simulations. Initially, the workflow employs a compression-oriented ROI extraction approach to enable multi-resolution methods for uniform data. We further propose adaptive padding and dynamic processing to improve the efficiency of three distinct compressors for multi-resolution data and improve the compression ratio of SOTA approaches by up to 3.3× under the same data quality. In addition, an advanced uncertainty visualization method is integrated to evaluate the compression impacts. In the future, we aim to investigate how to effectively apply our workflow to sparse data, given that each individual level of multi-resolution data essentially constitutes sparse data. We will also study how our workflow can preserve application-specific post-analysis quality such as Halo-finder. Additionally, we plan to explore post-processing curves beyond the Bézier curve and incorporate other visualization methods (e.g., volume rendering) to expand the scope of our uncertainty visualization for compression.

Acknowledgement

James Ahrens, Pascal Grosset, and Jesus Pulido are employees of Triad National Security, LLC, which operates Los Alamos National Laboratory under Contract No. 89233218CNA000001 with the U.S. Department of Energy (DOE) and National Nuclear Security Administration (NNSA), and their work on this project was funded by the U.S. DOE Office of Science (SC), Office of Advanced Scientific Computing Research (ASCR), under contracts DE-AC02-06CH11357 and DE-AC02-05CH11231 and the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of the DOE SC and NNSA. Tushar Athawale's funding was supported by the U.S. Department of Energy (DOE) RAPIDS-2 SciDAC project under contract number DE-AC0500OR22725. Daoce Wang and Dingwen Tao's work on this project was supported by the National Science Foundation (Grant Nos. 2312673, 2247080, and 2303064). Daoce Wang was also supported by the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of the DOE SC and NNSA during his summer internship at Los Alamos National Laboratory. Dingwen Tao was also supported by the National Natural Science Foundation of China (Grant Nos. 62032023 and T2125013), and the Innovation Funding of ICT, CAS (Grant No. E461050). This material is also based upon work supported by the CAMPA collaboration, a project of the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research and Office of High Energy Physics, Scientific Discovery through Advanced Computing (SciDAC) program.

References

[1] W. Zhang, A. Almgren, V. Beckner, J. Bell, J. Blaschke, C. Chan, M. Day, B. Friesen, K. Gott, D. Graves, *et al.*, "AMReX: A framework for block-structured adaptive mesh refinement," *Journal of Open Source Software*, vol. 4, no. 37, pp. 1370–1370, 2019.

[2] J. M. Stone, K. Tomida, C. J. White, and K. G. Felker, "The athena++ adaptive mesh refinement framework: Design and magnetohydrodynamic solvers," *The Astrophysical Journal Supplement Series*, vol. 249, no. 1, p. 4, 2020.

[3] A. Dubey, K. Weide, J. O'Neal, A. Dhruv, S. Couch, J. A. Harris, T. Klosterman, R. Jain, J. Rudi, B. Messer, M. Pajkos, J. Carlson, R. Chu, M. Wahib, S. Chawdhary, P. M. Ricker, D. Lee, K. Antypas, K. M. Riley, C. Daley, M. Ganapathy, F. X. Timmes, D. M. Townsley, M. Vanella, J. Bachan, P. M. Rich, S. Kumar, E. Endeve, W. R. Hix, A. Mezzacappa, and T. Papatheodore, "Flash-X: A multiphysics simulation software instrument," *SoftwareX*, vol. 19, p. 101 168, 2022, ISSN: 2352-7110. DOI: https://doi.org/10.1016/j.softx.2022.101168. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352711022001030.

[4] L. Fedeli, A. Huebl, F. Boillod-Cerneux, T. Clark, K. Gott, C. Hillairet, S. Jaure, A. Leblanc, R. Lehe, A. Myers, C. Piechurski, M. Sato, N. Zaim, W. Zhang, J. Vay, and H. Vincenti, "Pushing the frontier in the design of laser-based electron accelerators with groundbreaking mesh-refined particle-in-cell simulations on exascale-class supercomputers," in *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, **Winning Paper, 2022 ACM Gordon Bell Prize**, Los Alamitos, CA, USA: IEEE Computer Society, 2022, pp. 1–12. DOI: 10.1109/SC41404.2022.00008.

[5] J.-L. Vay, J.-C. Adam, and A. Héron, "Asymmetric pml for the absorption of waves. application to mesh refinement in electromagnetic particle-in-cell plasma simulations," *Computer Physics Communications*, vol. 164, no. 1, pp. 171–177, 2004, Proceedings of the 18th International Conferene on the Numerical Simulation of Plasmas, ISSN: 0010-4655. DOI: 10.1016/j.cpc.2004.06.026.

[6] J.-L. Vay, D. P. Grote, R. H. Cohen, and A Friedman, "Novel methods in the particle-in-cell accelerator code-framework warp," *Computational Science & Discovery*, vol. 5, no. 1, p. 014 019, 2012. DOI: 10.1088/1749-4699/5/1/014019.

[7] S. Kumar, J. Edwards, P.-T. Bremer, A. Knoll, C. Christensen, V. Vishwanath, P. Carns, J. A. Schmidt, and V. Pascucci, "Efficient i/o and storage of adaptive-resolution data," in *SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, IEEE, 2014, pp. 413–423.

[8] Y. Tian, S. Klasky, W. Yu, B. Wang, H. Abbasi, N. Podhorszki, and R. Grout, "Dynam: Dynamic multiresolution data representation for large-scale scientific analysis," in *2013 IEEE Eighth International Conference on Networking, Architecture and Storage*, IEEE, 2013, pp. 115–124.

[9] H. Bhatia, D. Hoang, N. Morrical, V. Pascucci, P.-T. Bremer, and P. Lindstrom, "Amm: Adaptive multilinear meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 6, pp. 2350–2363, 2022.

[10] W. Usher, X. Huang, S. Petruzza, S. Kumar, S. R. Slattery, S. T. Reeve, F. Wang, C. R. Johnson, and V. Pascucci, "Adaptive spatially aware i/o for multiresolution particle data layouts," in *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, IEEE, 2021, pp. 547–556.

[11] D. Tao, S. Di, Z. Chen, and F. Cappello, "Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization," in *2017 IEEE International Parallel and Distributed Processing Symposium*, IEEE, 2017, pp. 1129–1139.

[12] S. Di and F. Cappello, "Fast error-bounded lossy HPC data compression with SZ," in *2016 IEEE International Parallel and Distributed Processing Symposium*, IEEE, 2016, pp. 730–739.

[13] X. Liang, S. Di, D. Tao, S. Li, S. Li, H. Guo, Z. Chen, and F. Cappello, "Error-controlled lossy compression optimized for high compression ratios of scientific datasets," in *2018 IEEE International Conference on Big Data*, IEEE, 2018, pp. 438–447.

[14] P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2674–2683, 2014.

[15] M. Ainsworth, O. Tugluk, B. Whitney, and S. Klasky, "Multilevel techniques for compression and reduction of scientific data—the univariate case," *Computing and Visualization in Science*, vol. 19, no. 5–6, pp. 65–76, 2018.

[16] J. Tian, S. Di, K. Zhao, C. Rivera, M. H. Fulp, R. Underwood, S. Jin, X. Liang, J. Calhoun, D. Tao, and F. Cappello, "Cusz: An efficient gpu-based error-bounded lossy compression framework for scientific data," pp. 3–15, 2020.

[17] J. Tian, S. Di, X. Yu, C. Rivera, K. Zhao, S. Jin, Y. Feng, X. Liang, D. Tao, and F. Cappello, "Optimizing error-bounded lossy compression for scientific data on gpus," in *2021 IEEE International Conference on Cluster Computing (CLUSTER)*, IEEE, 2021, pp. 283–293.

[18] cuZFP, https://github.com/LLNL/zfp/tree/develop/src/cuda_zfp, Online, 2023.

[19] S. Jin, S. Di, F. Vivien, D. Wang, Y. Robert, D. Tao, and F. Cappello, "Concealing compression-accelerated i/o for hpc applications through in situ task scheduling," in *EuroSys 2024*, 2024.

[20] A. Huebl, R. Widera, F. Schmitt, A. Matthes, N. Podhorszki, J. Y. Choi, S. Klasky, and M. Bussmann, "On the scalability of data reduction techniques in current and upcoming hpc systems from an application perspective," in *High Performance Computing*, J. M. Kunkel, R. Yokota, M. Taufer, and J. Shalf, Eds., Cham: Springer International Publishing, 2017, pp. 15–29. DOI: 10.1007/978-3-319-67630-2_2.

[21] T. Lu, Q. Liu, X. He, H. Luo, E. Suchyta, J. Choi, N. Podhorszki, S. Klasky, M. Wolf, T. Liu, *et al.*, "Understanding and modeling lossy compression schemes on HPC scientific data," in *2018 IEEE International Parallel and Distributed Processing Symposium*, IEEE, 2018, pp. 348–357.

[22] H. Luo, D. Huang, Q. Liu, Z. Qiao, H. Jiang, J. Bi, H. Yuan, M. Zhou, J. Wang, and Z. Qin, "Identifying latent reduced models to precondition lossy compression," in *2019 IEEE International Parallel and Distributed Processing Symposium*, IEEE, 2019.

[23] D. Tao, S. Di, X. Liang, Z. Chen, and F. Cappello, "Optimizing lossy compression rate-distortion from automatic online selection between sz and zfp," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 8, pp. 1857–1871, 2019.

[24] F. Cappello, S. Di, S. Li, X. Liang, A. M. Gok, D. Tao, C. H. Yoon, X.-C. Wu, Y. Alexeev, and F. T. Chong, "Use cases of lossy compression for floating-point data in scientific data sets," *The International Journal of High Performance Computing Applications*, 2019.

[25] S. Jin, P. Grosset, C. M. Biwer, J. Pulido, J. Tian, D. Tao, and J. Ahrens, "Understanding gpu-based lossy compression for extreme-scale cosmological simulations," *arXiv preprint arXiv:2004.00224*, 2020.

[26] P. Grosset, C. Biwer, J. Pulido, A. Mohan, A. Biswas, J. Patchett, T. Turton, D. Rogers, D. Livescu, and J. Ahrens, "Foresight: Analysis that matters for data reduction," in *2020 SC20: International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, IEEE Computer Society, 2020, pp. 1171–1185.

[27] S. Jin, D. Tao, H. Tang, S. Di, S. Byna, Z. Lukic, and F. Cappello, "Accelerating parallel write via deeply integrating predictive lossy compression with hdf5," *arXiv preprint arXiv:2206.14761*, 2022.

[28] A. H. Baker, D. M. Hammerling, and T. L. Turton, "Evaluating image quality measures to assess the impact of lossy data compression applied to climate simulation data," in *Computer Graphics Forum*, Wiley Online Library, vol. 38, 2019, pp. 517–528.

[29] S. Di, J. Liu, K. Zhao, X. Liang, R. Underwood, Z. Zhang, M. Shah, Y. Huang, J. Huang, X. Yu, *et al.*, "A survey on error-bounded lossy compression for scientific datasets," *arXiv preprint arXiv:2404.02840*, 2024.

[30] H. Luo, J. Wang, Q. Liu, J. Chen, S. Klasky, and N. Podhorszki, "zMesh: Exploring application characteristics to improve lossy compression ratio for adaptive mesh refinement," in *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, IEEE, 2021, pp. 402–411.

[31] D. Wang, J. Pulido, P. Grosset, S. Jin, J. Tian, J. Ahrens, and D. Tao, "TAC: Optimizing error-bounded lossy compression for three-dimensional adaptive mesh refinement simulations," in *Proceedings of the 31st International Symposium on High-Performance Parallel and Distributed Computing*, 2022, pp. 135–147.

[32] D. Wang, J. Pulido, P. Grosset, S. Jin, J. Tian, K. Zhao, J. Ahrens, and D. Tao, "Tac+: Optimizing error-bounded lossy compression for 3d amr simulations," *IEEE Transactions on Parallel and Distributed Systems*, vol. 35, no. 3, pp. 421–438, 2024.

[33] D. Wang, J. Pulido, P. Grosset, J. Tian, S. Jin, H. Tang, J. Sexton, S. Di, K. Zhao, B. Fang, *et al.*, "Amric: A novel in situ lossy compression framework for efficient i/o in adaptive mesh refinement applications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2023, pp. 1–15.

[34] D. Tao, S. Di, Z. Chen, and F. Cappello, "Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization," in *2017 IEEE International Parallel and Distributed Processing Symposium*, IEEE, 2017, pp. 1129–1139.

[35] K. Zhao, S. Di, M. Dmitriev, T.-L. D. Tonellot, Z. Chen, and F. Cappello, "Optimizing error-bounded lossy compression for scientific data by dynamic spline interpolation," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, IEEE, 2021, pp. 1643–1654.

[36] Q. Gong, J. Chen, B. Whitney, X. Liang, V. Reshniak, T. Banerjee, J. Lee, A. Rangarajan, L. Wan, N. Vidal, Q. Liu, A. Gainaru, N. Podhorszki, R. Archibald, S. Ranka, and S. Klasky, "MGARD: A multigrid framework for high-performance, error-controlled data compression and refactoring," *SoftwareX*, vol. 24, p. 101 590, 2023.

[37] R. Ballester-Ripoll, P. Lindstrom, and R. Pajarola, "Tthresh: Tensor compression for multidimensional visual data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 9, pp. 2891–2903, 2020. DOI: 10.1109/TVCG.2019.2904063.

[38] K. Zhao, S. Di, X. Liang, S. Li, D. Tao, Z. Chen, and F. Cappello, "Significantly improving lossy compression for hpc datasets with second-order prediction and parameter optimization," in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, 2020, pp. 89–100.

[39] C. R. Johnson and A. R. Sanderson, "A next step: Visualizing errors and uncertainty," *IEEE Computer Graphics and Applications*, vol. 23, no. 5, pp. 6–10, 2003. DOI: 10.1109/MCG.2003.1231171.

[40] F. Ferstl, M. Kanzler, M. Rautenhaus, and R. Westermann, "Visual analysis of spatial variability and global correlations in ensembles of iso-contours," *Computer Graphics Forum*, vol. 35, no. 3, pp. 221–230, 2016. DOI: 10.1111/cgf.12898.

[41] T. M. Athawale, B. Ma, E. Sakhaee, C. R. Johnson, and A. Entezari, "Direct volume rendering with nonparametric models of uncertainty," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1797–1807, 2021. DOI: 10.1109/TVCG.2020.3030394.

[42] M. Otto, T. Germer, and H. Theisel, "Uncertain topology of 3D vector fields," in *IEEE Pacific Visualization Symposium (PacificVis)*, 2011, pp. 67–74. DOI: 10.1109/PACIFICVIS.2011.5742374.

[43] J. Wang, S. Hazarika, C. Li, and H.-W. Shen, "Visualization and visual analysis of ensemble data: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 9, pp. 2853–2872, 2019. DOI: 10.1109/TVCG.2018.2853721.

[44] F. Jiao, J. M. Phillips, Y. Gur, and C. R. Johnson, "Uncertainty visualization in HARDI based on ensembles of ODFs," in *IEEE Pacific Visualization Symposium*, 2012, pp. 193–200. DOI: 10.1109/PacificVis.2012.6183591.

[45] K. Brodlie, R. A. Osorio, and A. Lopes, "A review of uncertainty in data visualization," in *Expanding the Frontiers of Visual Analytics and Visualization*, J. Dill, R. Earnshaw, D. Kasik, J. Vince, and P. C. Wong, Eds., Springer Verlag London, 2012, pp. 81–109. DOI: 10.1007/978-1-4471-2804-5_6.

[46] K. Potter, P. Rosen, and C. R. Johnson, "From quantification to visualization: A taxonomy of uncertainty visualization approaches," in *Uncertainty Quantification in Scientific Computing*, A. M. Dienstfrey and R. F. Boisvert, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 226–249, ISBN: 978-3-642-32677-6. DOI: 10.1007/978-3-642-32677-6_15.

[47] D. Günther, J. Salmon, and J. Tierny, "Mandatory critical points of 2D uncertain scalar fields," *Computer Graphics Forum*, vol. 33, no. 3, pp. 31–40, 2014. DOI: 10.1111/cgf.12359.

[48] D. Vietinghoff, M. Böttinger, G. Scheuermann, and C. Heine, "Detecting critical points in 2D scalar field ensembles using Bayesian inference," in *IEEE Pacific Visualization Symposium (PacificVis)*, 2022, pp. 1–10. DOI: 10.1109/PacificVis53943.2022.00009.

[49] T. M. Athawale, S. Sane, and C. R. Johnson, "Uncertainty visualization of the marching squares and marching cubes topology cases," in *2021 IEEE Visualization Conference (VIS)*, 2021, pp. 106–110. DOI: 10.1109/VIS49827.2021.9623267.

[50] M. Davis, G. Efstathiou, C. S. Frenk, and S. D. White, "The evolution of large-scale structure in a universe dominated by cold dark matter," *The Astrophysical Journal*, vol. 292, pp. 371–394, 1985.

[51] J. Liu, S. Di, K. Zhao, X. Liang, Z. Chen, and F. Cappello, "Dynamic quality metric oriented error bounded lossy compression for scientific datasets," in *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, Los Alamitos, CA, USA: IEEE Computer Society, 2022, pp. 1–15. DOI: 10.1109/SC41404.2022.00067. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SC41404.2022.00067.

[52] D. Wang, J. Pulido, P. Grosset, J. Tian, J. Ahrens, and D. Tao, "Analyzing impact of data reduction techniques on visualization for amr applications using amrex framework," in *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*, 2023, pp. 263–271.

[53] K. Pöthkow, B. Weber, and H.-C. Hege, "Probabilistic marching cubes," *Computer Graphics Forum*, vol. 30, no. 3, pp. 931–940, 2011. DOI: https://doi.org/10.1111/j.1467-8659.2011.01942.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-

8659.2011.01942.x. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2011.01942.x.

[54] T. M. Athawale, E. Sakhaee, and A. Entezari, "Isosurface visualization of data with nonparametric models for uncertainty," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 777–786, 2015.

[55] Z. Wang, T. M. Athawale, K. Moreland, J. Chen, C. R. Johnson, and D. Pugmire, "FunMC^2: A Filter for Uncertainty Visualization of Marching Cubes on Multi-Core Devices," in *Eurographics Symposium on Parallel Graphics and Visualization*, R. Bujack, D. Pugmire, and G. Reina, Eds., The Eurographics Association, 2023, ISBN: 978-3-03868-215-8. DOI: 10.2312/pgv.20231081.

[56] P Lindstrom, "Error distributions of lossy floating-point compressors," Oct. 2017. [Online]. Available: https://www.osti.gov/biblio/1526183.

[57] S. Jin, S. Di, J. Tian, S. Byna, D. Tao, and F. Cappello, "Improving prediction-based lossy compression dramatically via ratio-quality modeling," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, 2022, pp. 2494–2507. DOI: 10.1109/ICDE53745.2022.00232.

[58] IEEE VISUALIZATION, http : / / vis . computer . org / vis2004contest/data.html, Online, 2014.

[59] NYX simulation, https://amrex-astro.github.io/Nyx/, Online, 2019.

[60] Oak Ridge Leadership Computing Facility. "WarpX, granted early access to the exascale supercomputer Frontier, receives the high-performance computing world's highest honor." Online. (2023), [Online]. Available: https://www.olcf.ornl.gov/2022/11/17/plasma-simulation-code-wins-2022-acm-gordon-bell-prize/.

[61] Bridges-2, https://www.psc.edu/resources/bridges-2/, Online, 2021.

[62] S. T. Brown, P. Buitrago, E. Hanna, S. Sanielevici, R. Scibek, and N. A. Nystrom, "Bridges-2: A platform for rapidly-evolving and data intensive research," ser. PEARC '21, Boston, MA, USA: Association for Computing Machinery, 2021, ISBN: 9781450382922. DOI: 10.1145/3437359.3465593. [Online]. Available: https://doi.org/10.1145/3437359.3465593.

[63] IAMR. (2023).

[64] Scientific Data Reduction Benchmarks, https://sdrbench.github.io/, Online, 2019.

[65] S. Jin, J. Pulido, P. Grosset, J. Tian, D. Tao, and J. Ahrens, "Adaptive configuration of in situ lossy compression for cosmology simulations via fine-grained rate-quality modeling," *arXiv preprint arXiv:2104.00178*, 2021.

Appendix: Artifact Description

Artifact Description (AD)

I. Overview of Contributions and Artifacts

A. Paper's Main Contributions

C_1 We propose **SZ3MR**, an optimization of the state-of-the-art global lossy compressor SZ3 for multi-resolution data, using dynamic padding and adaptive error bounds within the SZ3 compressor to improve prediction accuracy and compression quality.

C_2 We develop an efficient and effective error-bounded **post-processing solution** that leverages spatial information across each compressed block to significantly enhance the quality of block-wise compressors.

C_3 We investigate the uncertainty introduced by lossy compression by integrating a cutting-edge **uncertainty visualization** technique. To better understand how compression affects the data via visualization.

B. Computational Artifacts

A_{1-3} https://github.com/FabioGrosso/SC24-MRZ-image
A_{1-3} https://github.com/FabioGrosso/SC24-MRZ
A_3 https://github.com/wangzhezhe/UCV

Artifact ID	Contributions Supported	Related Paper Elements
A_1	C_1	Figures 15, 17, 18 Table 4
A_2	C_2	Tables 6-8
A_3	C_3	Figure 14

II. Artifact Identification

A. Computational Artifact A_1

Relation To Contributions

See TABLE I-B for relations among each artifact, contribution supported, and related paper elements.

Expected Results

The expected results of the experiments should be consistent with those of the main paper. Specifically, our SZ3MR yields better compression performance than all three baselines (Original SZ3, TAC's SZ3, and AMRIC's SZ3) on multi-resolution data, especially at high compression ratios. Additionally, our SZ3MR should offer improved output speed over AMRIC's SZ3 in in-situ AMR data compression.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is approximately 17 min, including around 5 min for Artifact Setup, 10 min for Artifact Execution, and 2 min for Artifact Analysis.

Artifact Setup (incl. Inputs)

Hardware: Processor: $\geq$ 96 cores; Storage: $\geq$ 128 GBs; Memory: $\geq$ 192 GB RAM

Software: gcc/9.4.0 (or 9.3.0); cmake ($\geq$3.23); OpenMPI/4.1.1; python/3.8; hdf5/1.12.2;
Our SZ3MR, baselines AMR compressors (TAC and AMRIC), original SZ3, Nyx and WarpX simulations (all packed): (https://github.com/FabioGrosso/SC24-MRZ)

Datasets / Inputs: To save the user's time on in-situ evaluation, we have pre-run the Nyx and WarpX simulations to obtain checkpoint files from later timesteps. These checkpoint files have been included in the artifact. Other needed datasets for the offline evaluation can be obtained at (https://sdrbench.github.io/) and in the artifact (https://github.com/FabioGrosso/SC24-MRZ-image).

Installation and Deployment: To streamline the review process, we prepared a Singularity image for the evaluation. The user need to install Singularity, download our pre-built Singularity image file, and then run the Singularity image file to start the evaluation. Authors can also build from source (or we can provide a pre-built version on Chameleon Cloud), which requires first installing CMake, numpy, and OpenMPI. Then, install the original SZ3 compressor, our SZ3MR, and baselines (TAC and AMRIC). Lastly, the Nyx and WarpX simulations need to be installed.

Artifact Execution

The workflow consists of three tasks: T_1, T_2, and T_3, with dependencies $T_1 \rightarrow T_2$ and $T_1 \rightarrow T_3$. Task T_1 is first performed to generate decompressed data using our SZ3MR and the baselines. Subsequently, T_2 evaluates the compression ratio and decompressed data quality in terms of PSNR for our SZ3MR and the baselines. Task T_3, following T_1, measures the execution speed of our SZ3MR and the AMRIC baseline. Regarding experimental parameters, we utilize the same input data size and error bound as in the main paper, given the use of identical input data. The detailed input parameter will be included in scripts for ease of use. The workflow will be executed multiple times on different applications using different error bounds to generate the rate-distortion curve.

Artifact Analysis (incl. Outputs)

T_2 will output the ratio-distortion of our SZ3MR and the baselines, with our SZ3MR expected to demonstrate superior performance compared to the baselines, as evidenced by the results in Figures 15, 17, and 18. T_3 will measure the execution speed of our SZ3MR and the baseline, with our SZ3MR achieving faster speeds, replicating the findings in Table 4.

B. Computational Artifact A_2

Expected Results

The expected results of the experiments are consistent with the main paper. Specifically, our post-processing solution can enhance the data quality from SZ2 and ZFP compressors under different compression ratios. Our post-processing solution also introduces low overhead. We will provide scripts to extract and output the results of the experiment.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is approximately 8 min, including around 5 min for Artifact Setup, 2 min for Artifact Execution, and 1 min for Artifact Analysis.

Artifact Setup (incl. Inputs)

Hardware: Same as A_1.

Software: Part of the required software is already fulfilled by A_1. The new software needed includes our post-processing solution and the SZ2/ZFP compressors, also available at https://github.com/FabioGrosso/SC24-MRZ.

Datasets / Inputs: Same as A_1.

Installation and Deployment: Similar to A_1, A_2 is also included in the pre-built Singularity image. To build A_2 from the source, the users need to install the SZ2 and ZFP compressors after completing the evaluation of A_1.

Artifact Execution

The workflow consists of three tasks: T_1, T_2, and T_3. Such that $T_1 \rightarrow T_2$ and $T_1 \rightarrow T_3$. Task T_1 is initially performed to generate original decompressed datasets and post-processed datasets using our post-processing solution. Subsequently, T_2 outputs the data quality of the original decompressed data and our post-processing solution. Task T_3, following T_1, outputs the execution time of our post-processing solution and the SZ2/ZFP compressor. For experimental parameters, we utilize the same input data size due to the identical input data. We also apply the same error bounds as in the paper. The detailed input parameter will be included in scripts. The workflow will be executed multiple times on different datasets using different error bounds.

Artifact Analysis (incl. Outputs)

T_2 will output the post-processed data quality, consistently surpassing that of the original decompressed data, thus replicating the results presented in Tables 6-7. T_3 will assess the efficiency of our post-processing solution, demonstrating its minimal overhead, in alignment with Table 8. We will provide scripts to extract and output the results of the experiment.

C. Computational Artifact A_3

Expected Results

The expected results of the experiments are consistent with the main paper. Specifically, our uncertainty visualization can reproduce Figure 14 in the paper.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is approximately 8 min, including around 5 min for Artifact Setup, 2 min for Artifact Execution, and 1 min for Artifact Analysis.

Artifact Setup (incl. Inputs)

Hardware: Same as A_1.

Software: Part of the required software is already fulfilled by A_1. The new software needed are ParaView: https://github.com/Kitware/ParaView/ and UCV: https://github.com/wangzhezhe/UCV.

Datasets / Inputs: The dataset will be fulfilled by A_1, andA_2.

Installation and Deployment: Download and compile ParaView first, then download and compile the UCV using ParaView's library. After that, run ParaView and load the UCV plugin to visualize the decompressed data with uncertainty.

Artifact Execution

The workflow consists of three tasks: T_1 and T_2, with the sequence $T_1 \rightarrow T_2$. Task T_1 is performed to generate decompressed data and model the uncertainty of the decompressed data. Then, T_2 involves visualizing the iso-surface of both the original and decompressed data. Subsequently, T_2 outputs the uncertainty visualization (Fig. 14c) of the decompressed data using the modeled uncertainty. For experimental parameters, we adhere to the same error bound (i.e., 0.0005) and iso-value (i.e., 0.0006) as in the paper.

Artifact Analysis (incl. Outputs)

T_2 will visualize the original, decompressed, and decompressed data with uncertainty, aiming to replicate Fig. 14 in the paper. The visualization will be saved in PNG files under the current directory.

Artifact Evaluation (AE)

Artifact DOI: 10.5281/zenodo.13272693

We provide a Singularity container image to enhance reproducibility. While preparing the artifacts, we executed them on a single node from the Chameleon Cloud (https://www.chameleoncloud.org/), equipped with two Intel(R) Xeon(R) Platinum 8380 CPUs and 256 GB RAM (specifically, the `compute_icelake_r650` instance). To ensure a clearer and more streamlined workflow, as well as good reproducibility, we focus on reproducing the principal parts of the evaluation (see artifact analysis for more details).

A. Computational Artifact A_1

Artifact Setup (incl. Inputs)

1) Install Singularity (https://singularity-tutorial.github.io/01-installation/).
2) Download the pre-built Singularity image via Github.
```
$ git clone https://github.com/FabioGrosso\
/SC24-MRZ-image
$ cat SC24-MRZ-image/mrz.part.* > mrz.sif
```
3) Build the image file (need root privilege).
```
$ sudo singularity build --sandbox mrz mrz.sif
```

Artifact Execution

0) Run the image file (need root privilege) and set up environmental variables.
```
$ sudo singularity shell --writable mrz
export OMPI_DIR=/opt/ompi
export OMPI_VERSION=4.1.1
export PATH=$OMPI_DIR/bin:$PATH
export LD_LIBRARY_PATH=$OMPI_DIR/lib:\
$LD_LIBRARY_PATH
export LD_LIBRARY_PATH=/opt/zfp/build/lib:\
$LD_LIBRARY_PATH
export MANPATH=$OMPI_DIR/share/man:$MANPATH
export C_INCLUDE_PATH=/opt/ompi/include:\
$C_INCLUDE_PATH
export CPLUS_INCLUDE_PATH=/opt/ompi/include:\
$CPLUS_INCLUDE_PATH
export OMPI_ALLOW_RUN_AS_ROOT=1
export OMPI_ALLOW_RUN_AS_ROOT_CONFIRM=1
export DISPLAY=:99
```
1.1) Run NYX with no compression, original SZ3 compression, AMRIC, and ours (2 iterations, 8 mins).
```
$ cd /home/nyx256/
$ . go.sh
```
1.2) Compare I/O performance between AMRIC and ours in NYX.
```
$ cd /home/nyx256/otfile
$ . io.sh
```
1.3) Evaluate NYX's data quality and compression ratio for Baseline-SZ3, AMRIC-SZ3, and ours.
```
$ cd /home/nyx256/run
$ . decomp.sh
$ . qualityCR.sh
```
2.1) Run WarpX with no compression, original SZ3 compression, and ours (2 iterations, 3 mins).
```
$ cd /home/wpx/
$ . go.sh
```
2.2) Evaluate WarpX's data quality and compression ratio for Baseline-SZ3, and ours.
```
$ cd /home/wpx/diags
$ . decomp.sh
$ . qualityCR.sh
```
3.1) Compress RT data using with Baseline-SZ3, AMRIC-SZ3, TAC-SZ3, and ours.
```
$ cd /home/rtamr/
$ . go.sh
```
3.2) Evaluate RT data quality and CR for Baseline-SZ3, AMRIC-SZ3, TAC-SZ3, and ours.
```
$ . qualityCR.sh
```

Artifact Analysis (incl. Outputs)

1) The expected results for step 1.2 are:
```
*************** Error Bound 1 ***************
---------- Writing Time for Ours ----------
Ours Preprocess time = 0.29 seconds
Ours Compression+Writing time = 1.76 seconds
Ours Total time = 2.06 seconds

---------- Writing Time for AMRIC ---------
AMRIC Preprocess time = 0.90 seconds
AMRIC Compression+Writing time = 1.52 seconds
AMRIC Total time = 2.39 seconds
...
```
This should be consistent with Table 4 in the paper, that our solution has lower pre-process and total time than AMRIC.

2) The expected results for step 1.3 are:
```
*************** Error bound 1 ***************
---------- Data Quality and CR for Baseline-SZ3
PSNR: 54.56 | Compression Ratio: 165.22
---------- Data Quality and CR for AMRIC-SZ3
PSNR: 54.83 | Compression Ratio: 200.96
---------- Data Quality and CR for Ours(pad+eb)
PSNR: 60.62 | Compression Ratio: 302.75
...
```
The results replicate Figure 15 (left, Fine level) for high compression ratios (CR) in the paper, demonstrating that our solution (pad+eb) achieves higher PSNR and CR compared to Baseline and AMRIC-SZ3.
We focus on the fine level of AMR data, which is more critical within the dataset. Additionally, we emphasize high CR, as our solution is particularly efficient towards high CR scenarios.

3) The expected results for step 2.2 are:
```
*************** Error bound 1 ***************
---------- Data Quality and CR for Baseline-SZ3
PSNR: 69.24 | Compression Ratio: 193.45
---------- Data Quality and CR for Ours(pad+eb)
PSNR: 81.27 | Compression Ratio: 226.75
...
```
The results replicate Figure 17 (left, WarpX) for high CR, demonstrating that our solution (pad+eb) achieves higher PSNR and CR compared to Baseline-SZ3.
We focus on WarpX as it enables in-situ compression using real-world applications. We also emphasize high CR as our solution focuses more on high CR scenarios.

4) The expected results for step 3.2 are:
```
*************** Error bound 1 ***************
---------- Data Quality and CR for Baseline-SZ3
PSNR: 30.48 | Compression Ratio: 121.95
---------- Data Quality and CR for AMRIC-SZ3 --
PSNR: 30.06 | Compression Ratio: 114.08
```

```
---------- Data Quality and CR for TAC-SZ3 ----
PSNR: 30.80 | Compression Ratio: 148.15
---------- Data Quality and CR for Ours(pad+eb)
PSNR: 35.56 | Compression Ratio: 176.98
...
```

These should replicate the result in Figure 18 (right, RT) for high CR, demonstrating that our solution (pad+eb) achieves higher PSNR and CR compared to Baseline-SZ3, AMRIC-SZ3, and TAC-SZ3.
We focus on the RT dataset to encompass a broader range of dataset types for artifact evaluation.

B. Computational Artifact A_2

Artifact Setup (incl. Inputs)

Same as A_1.

Artifact Execution

0) Run the image file (**skip if already run for A_1**).
```
$ sudo singularity shell --writable mrz
```
1.1) Compress/decompress multi-resolution dataset Hurricane using SZ2/ZFP with our post-process.
```
$ cd /home/post-mr/
$ . go.sh
```
1.2) Evaluate Hurricane data quality and CR for original SZ2, ZFP, and with our post-process.
```
$ . qualityCR.sh
```
2.1) Compress/decompress uniform-resolution dataset S3D using SZ2/ZFP with our post-process.
```
$ cd /home/post-uni/
$ . go.sh
```
2.2) Evaluate S3D data quality and CR for original SZ2, ZFP, and with our post-process.
```
$ . qualityCR.sh
```
2.3) Evaluate the run time of the original compression and post-processing on S3D using ZFP and SZ2 (Serial).
```
$ . time.sh
```

Artifact Analysis (incl. Outputs)

Artifact Analysis (incl. Outputs)

1) The expected results for step 1.2 are:
```
******* Data Quality and CR for ZFP *******
------------- Error bound 1 ------------
Compression Ratio: 240
PSNR-Ori:  40.11
PSNR-Post: 42.19
...
******* Data Quality and CR for SZ2 *******
------------- Error bound 1 ------------
Compression Ratio: 170
PSNR-Ori:  41.87
PSNR-Post: 43.27
```
This should replicate the bottom half of Table 6 (Hur) in the paper (first two columns), showing that our post-processing improves the PSNR of both ZFP and SZ2.

2) The expected results for step 2.2 are:
```
******* Data Quality and CR for ZFP *******
------------- Error bound 1 ------------
Compression Ratio: 106
PSNR-Ori:  62.74
PSNR-Post: 65.90
...
******* Data Quality and CR for SZ2 *******
------------- Error bound 1 ------------
Compression Ratio: 180
PSNR-Ori:  67.72
PSNR-Post: 70.16
...
```
This should replicate the upper half of Table 7 (S3D, 2nd and 3rd columns) in the paper, showing that our post-processing improves the PSNR of ZFP and SZ2.
We chose the S3D dataset to cover a broader range of datasets, and its output will also be used to reproduce Table 8 in step 2.3.

3) The expected results for step 2.3 are:
```
Singularity> . time.sh
*************** Error Bound 1 ***************
--Time of post-processing and ZFP(OpenMP)--
...
5. Time taken by Ori is: 1.883 sec
6. Time taken by Extra is: 0.096 sec
Overhead: 0.051

---Time of post-processing and SZ2(Serial)---
...
5. Time taken by Ori is: 5.790 sec
6. Time taken by Extra is: 0.094 sec
Overhead: 0.016
*************** Error Bound 2 ***************
--Time of post-processing and ZFP(OpenMP)--
...
5. Time taken by Ori is: 4.469 sec
6. Time taken by Extra is: 0.093 sec
Overhead: 0.021
---Time of post-processing and SZ2(Serial)---
...
5. Time taken by Ori is: 8.121 sec
6. Time taken by Extra is: 0.091 sec
Overhead: 0.011
```
The results are consistent with Table 8, specifically for the rows ZFP(OpenMP) and SZ2(Serial), demonstrating that our post-processing introduces minimal overhead, less than 5% for most cases. Note that the speed may fluctuate due to potential system performance instability. We focus on the Serial SZ2 because SZ2 with OpenMP results in a lower compression ratio, owing to its embarrassingly parallel nature.

C. Computational Artifact A_3

Artifact Setup (incl. Inputs)

Same as A_1. A_3 should be executed after step 2.1 of A_2.

Artifact Execution

Compute the uncertainty of the decompressed hurricane data. And generate visualizations of original data (fig-14a.png), decompressed data (fig-14b.png), and decompressed data with uncertainty (fig-14c.png).

```
$ cd /home/vis
$ . go.sh
```

Artifact Analysis (incl. Outputs)

Three images will be will generated: fig-14a.png, fig-14b.png, and fig-14c.png. These images reproduce the uncertainty visualization presented in Figure 14 of the paper.