Improving the Efficiency of Interpolation-based Scientific Data Compressors with Adaptive Quantization Index Prediction

Pu Jiao*, Sheng Di†, Mingze Xia*, Xuan Wu*, Jinyang Liu‡, Xin Liang*, Franck Cappello†
*Department of Computer Science, University of Kentucky, Lexington, KY, USA
{pujiao, Mingze.Xia, xuan.wu, xliang}@uky.edu
†Argonne National Laboratory, Lemont, IL, USA
sdi1@anl.gov, cappello@mcs.anl.gov
‡Department of Computer Science, University of Houston, Houston, TX, USA
jliu217@Central.UH.EDU

***Abstract*—Large-scale scientific simulations produce unprecedented amounts of data using high-performance computing systems, leading to severe problems in data storage, I/O, and communication. To address the data movement challenge, error-controlled lossy compression has been proposed to significantly reduce the data size while retaining the data quality. Recently, interpolation-based compressors, including MGARD, SZ3, QoZ, and HPEZ, have stood out due to their efficiency in obtaining relatively high compression ratios with decent compression and decompression throughput. Nevertheless, these methods focus on data decorrelation in the compression pipeline yet overlook the correlation of the quantization indices generated after decorrelation. In this paper, we develop a generic framework that can use the correlation of quantization indices to significantly improve the compression ratios for state-of-the-art interpolation-based error-bounded lossy compressors. Our contributions are three-fold: (1) We carefully characterized the quantization index array produced by the interpolation-based compressors and identified the unused correlation; (2) We designed a generic quantization index prediction method to exploit such correlation, which leads to improved compression ratio with only minor degradation in throughput; (3) We integrate our method into 4 state-of-the-art interpolation-based compressors and evaluate them using 5 real-world datasets. Experimental results demonstrate that the proposed method improves the compression ratios of the base compressors by up to 95% while keeping the same quality. It also leads to 16% improvement in end-to-end data transfer performance under a parallel setting.**

***Index Terms*—High-performance computing, scientific data, lossy compression, error control**

I. Introduction

Today's high-performance computing (HPC) systems are producing unprecedented amounts of data. High-resolution climate simulations, for example, can generate tens of terabytes of data every 16 seconds [1]. This poses grand challenges to data movement tasks as the improvement of network and I/O systems has fallen far behind the data generation speed, thus hindering downstream data analytics crucial for advancing scientific discoveries.

To mitigate this problem, error-controlled lossy compression [2]–[16] has been proposed and extensively studied in the last decade. These compressors ensure the distortion between the original and decompressed data is less than a user-specified error bound, which addresses the limitations of low compression ratios in lossless compressors and unbounded errors in traditional lossy compressors. As such, they are widely used in multiple scientific use cases, including data storage and I/O [17], data transfer [18], and data streaming [19].

Recently, interpolation methods have been applied in error-controlled lossy compressors for better compression performance and quality due to their high efficiency in decorrelating scientific data. For instance, MGARD [14], [15] leverages multilinear interpolation functions to approximate the data with finite element methods, and then applies a L^2 projection to obtain a hierarchical representation. This further enables the preservation of certain families of quantities-of-interest (QoIs) via a rigorous theory [16]. Similarly, linear and cubic spline interpolations have been directly used in the data prediction stage of SZ3 [6] and QoZ [8], which significantly improves the compression ratios when the requested error bound is relatively low. HPEZ [9] is recently proposed to investigate a versatile interpolation-based predictor to explore multi-dimensional interpolation and auto-tuning strategies. This further improves the compression ratios at the cost of some unavoidable performance degradation.

TABLE I
State-of-the-art interpolation-based compressors

Compressor	Compression Speed	Compression Ratios	Resolution reduction	GPU support	QoI support	Quality oriented
MGARD	Low	Low	✓	✓	✓	✗
SZ3	High	Medium	✗	✗	✓	✗
QoZ	High	Medium	✗	✓	✗	✓
HPEZ	Medium	High	✗	✗	✗	✓

While all these compressors use interpolation in their compression algorithms, they have unique characteristics that can benefit different scientific use cases. Table I characterizes the four compressors in terms of compression speed, compression ratio, resolution reduction, GPU support, QoI support, and quality orientation. For instance, only MGARD provides resolution reduction despite its relatively low compression speed and ratios, which is very useful when the degree of freedom in the data needs to be reduced to accelerate downstream analysis [20]. Meanwhile, both MGARD and QoZ have GPU

Corresponding author: Xin Liang, Department of Computer Science, University of Kentucky, KY 40506, USA.

support [21], [22] to accommodate use cases requiring high throughput, which is often the case for data streaming tasks in scientific instruments [23]. MGARD and SZ3 also support the preservation of several symbolistic QoIs that are essential in some application domains [16], [24]. Finally, QoZ and HPEZ feature dynamic quality orientation with an auto-tuning module, which provides the best compression ratios under varying quality metrics. In addition to the generic compressors, several customized compressors for specific applications were also developed with the interpolation-based compression framework [25].

With the wide usage of interpolations in scientific data compressors, designing new methods to improve their efficiency is important but non-trivial. The challenges are three-fold. First, existing approaches have exhausted multiple directions for optimizations, including interpolation functions [6], auto-tuning procedures [8], and interpolation orders [9]. This leaves little room for further improvement. Second, different compressors may have varied interpolation functions and compression mechanisms, which leads to diverse characteristics that are hard to unify using a single routine. Third, while high compression ratios are preferred in scientific lossy compressors, achieving that with major degradation in throughput is undesired. Having a balanced trade-off between the compression ratio and compression speed is usually not an easy task.

In the four compressors mentioned above, the original data is first decorrelated by different interpolation-based methods, and then quantized to an array of integers, also known as quantization index array. These integers are further compressed by variable-length and lossless encoding methods such as Huffman encoding [26] and ZSTD [27]. As a significant component of the compressed data, the compressibility of the quantization index array has a direct impact on the compression ratio but has not yet been well studied.

In this paper, we found that the quantization indices produced by interpolation-based methods could be highly correlated in certain regions, which can be exploited to improve the efficiency of interpolation-based compressors. As such, we carefully characterize the quantization index array from the leading interpolation-based compressors and identify the clustering regions with high correlation. We further propose a generic method to exploit such correlation using an adaptive quantization index prediction (QP) algorithm. This can significantly improve compression ratios while keeping the same quality of the decompressed data with only minor throughput degradation. Our contributions are summarized as follows.

- We comprehensively characterize the clustering phenomenon of quantization indices based on four leading interpolation-based compressors. Such phenomena will cause a high entropy in local regions and, thus, suboptimal overall compression ratios.
- We propose a lightweight prediction method to reduce the entropy of the quantization index array based on our characterization. We also explore the best-fit configuration of the prediction method to achieve high efficiency.
- We integrate the proposed method into the four leading interpolation-based compressors and evaluate them using seven real-world scientific datasets. Experimental results demonstrate that our method leads to up to 95% improvement over the base compressors in the compression ratios under the same quality. A parallel data transfer experiment using SZ3 shows that our method can improve the end-to-end data transfer performance to $1.16\times$.

The rest of this paper is organized as follows. Section II reviews the literature on lossy scientific data compressors. Section III provides an overview of the proposed method. Section IV describes the characterization of quantization indices after applying interpolation-based method. Section V details our quantization index prediction strategies and the corresponding parameter exploration. Section VI presents and analyzes the experimental results. Section VII concludes the paper with a vision for future works.

II. Related works

Data compression is regarded as a promising way to address the big data challenge in scientific applications due to its efficiency in reducing the data size. Generic lossless compressors, such as ZSTD [27], GZIP [28], and Blosc [29], are able to recover the exact data, but they usually suffer from limited compression ratios for floating-point scientific data (less than 2 in many cases [30]). While traditional lossy compression methods, such as JPEG [31] and JPEG2000 [32], can provide tunable compression ratios to accommodate different use cases, they do not enforce error bounds that are usually required to ensure the correctness of scientific analysis. Error-controlled lossy compressors [2]–[16] are proposed as an alternative way to mitigate this problem. These compressors can provide significant compression ratios while enforcing an error bound specified by the users.

Error-controlled lossy compressors usually comprise three major steps: decorrelation, quantization, and encoding. They can be categorized into prediction-based model [2]–[9] and transform-based model [10]–[13] in general, depending the major decorrelation methods adopted. The SZ compressor family [3]–[6] is a set of representative prediction-based compressors. In the SZ compression pipeline, the data is first decorrelated using one or multiple pre-set prediction algorithms, and then quantized into an array of integers (a.k.a., quantization indices) based on the user-specified error bound. After that, the quantization indices are fed to a Huffman encoder [26] and ZSTD [27] for lossless compression. With several years of development, the prediction algorithms in SZ have evolved from one-dimensional curve fitting [3] to Lorenzo prediction [4], regression [5], and spline interpolation [6]. Transform-based compressors leverage specific transforms for data decorrelation, and then perform quantization and encoding in the transformed domains. ZFP [10] is a typical transform-based compressor that divides data into non-overlapped data blocks for independent compression. In each block, it converts data into fixed-point representations and applies a near-orthogonal transform for data decorrelation. The transformed data are quantized and encoded using an embedded encoding algorithm to achieve size reduction. Other notable transformed-based compressors include TTHRESH [11], SPERR [12], and FAZ [13]. While these compressors deliver high compression ratios, they usually suffer from relatively low compression/decompression throughput due to the introduction of complex operations such as singular value decomposition and wavelet transform.

MGARD [14]–[16] is a lossy compressor that lies in the middle of prediction-based and transform-based compressors. It relies on multilinear interpolation along with a L^2 projection for data decorrelation, and applies quantization in a level-wise fashion. Similar to SZ, the quantized integers, or quantization indices, are compressed using lossless methods.

To accommodate various scientific use cases [23], multiple variations of error-controlled compressors have also been proposed and developed. For instance, MGARD supports the preservation of certain families of QoIs, such as adaptive decimation and linear quantities, through a rigorous development of theory [16]. SZ3 has been equipped with similar functionality via the derivation and application of point-wise error bounds [24]. Later, QoZ [8] has been proposed to enhance SZ3 with a highly-parameterized multi-level interpolation-based data predictor and an adaptive auto-tuner, leading to optimized compression ratios according to user-specified quality metrics during online compression. Note that both MGARD and QoZ have their respective GPU supports [21], [22] to achieve high throughput. Recently, HPEZ [9] has demonstrated further improvement by incorporating multi-dimensional interpolation and tailored optimizations such as block-wise interpolation tuning and dynamic dimension freezing. While it generally leads to the best compression ratios under the same quality constraints, the complex design in the interpolation strategy and the highly serial nature of the tuning process make it hard to parallelize on GPU architectures.

In this paper, we focus on improving the compression ratios of interpolation-based compressors by investigating and optimizing the quantization indices after applying interpolation for decorrelation. Our method provides much higher compression ratios without affecting the compression quality. The proposed techniques will benefit a broad range of interpolation-based compressors, including but not limited to MGARD, SZ3, QoZ, and HPEZ, and can be simply applied to their respective variations for QoI preservation and GPU parallelization.

III. Overview

In this section, we first introduce the quality assessment metrics used in the paper, followed by an overview of the proposed techniques on quantization index prediction.

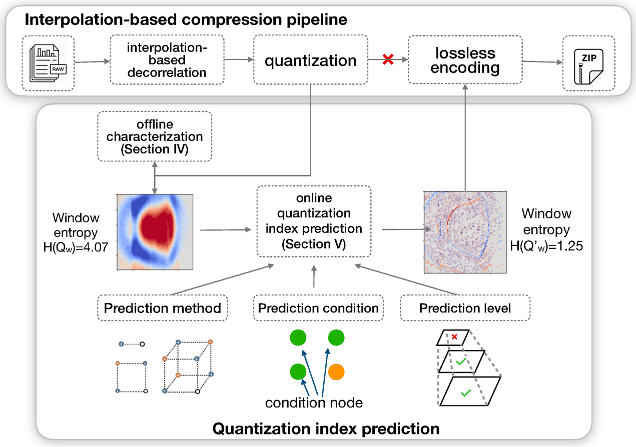
A. Quality assessment

We use compression ratios (CR) under specific error bounds to assess compression quality under the same distortion in absolute error, and the higher compression ratios indicate better quality. We also use rate-distortion graphs to assess the compression quality under the same distortion in mean squared error, which is widely used in the community [6], [9], [10]. The rate-distortion shows the relationship between the data quality metric Peak Signal-to-Noise ratio (PSNR) and the compression metric bit-rate. PSNR is computed using the following formula:

$$PSNR(d, d') = 20 \log_{10} \frac{\max(d) - \min(d)}{\sqrt{MSE(d, d')}},$$

where d is the original data, d' is the decompressed data, and $MSE(d, d')$ is the mean squared error between d and d'. Bit-rate denotes the average number of bits needed to represent a single data point in the compressed file. It can be computed by 32/64 over compression ratio for single-/double-precision floating-point data. As such, a lower bit-rate under the same PSNR indicates better compression quality.

We also use Shannon entropy [33] to assess the compressibility of the quantization index array. In particular, given the integer array Q of quantization index, the Shannon entropy is defined as $H(Q) = -\sum p_i \log p_i$, where $\{p_i\}$ is the frequency array (i.e., each entry in the array is computed by the number of occurrences for a symbol in Q divided by the total size of Q). Generally speaking, lower entropy in the quantization indices indicates better compressibility, which usually leads to a better compression ratio after lossless encoding.



Interpolation-based compression pipeline
RAW
interpolation-based decorrelation
quantization
lossless encoding
ZIP
offline characterization (Section IV)
Window entropy H(Q_W)=4.07
online quantization index prediction (Section V)
Window entropy H(Q'_W)=1.25
Prediction method
Prediction condition
Prediction level
condition node
Quantization index prediction

Fig. 1. Design overview of quantization index prediction

B. System design

We present the overview of the proposed quantization index prediction workflow for interpolation-based scientific compressors in Figure 1. The top box in the figure depicts the general interpolation-based compression pipeline, which comprises interpolation-based decorrelation, quantization, and lossless encoding. The quantization index array, or quantization indices, are produced by quantizing the coefficients from the interpolation-based decorrelation, and they will be losslessly compressed with entropy encoders and lossless compressors in the next stage.

The box below illustrates the proposed work on quantization index prediction (QP). In particular, we perform an offline characterization of the quantization index array to identify specific regions of interest with high correlation (Section IV). We then explore effective prediction methods and configurations to exploit such correlation to reduce the entropy of the quantization index array (Section V). The identified prediction method is used to transform the original quantization array Q into a lower-entropy representation Q' in the quantization stage, and then intercept the original pipeline to compress Q' instead of Q during the lossless encoding stage. This will generally lead to better compression ratios under the same distortion as will be validated in our evaluation (Section VI).

IV. Quantization Index Characterization

In this section, we carefully characterize the quantization indices of interpolation-based scientific lossy compressors and find out that they have significant **clustering effect** with multiple highly-correlated data regions. This motivates and lays the foundation of the quantization index prediction methods that will be introduced in the next section. In the following, we use SZ3 as an example to introduce how interpolations are used in scientific data compressors for decorrelation, followed by our characterization results.

A. SZ3 Recap

We first introduce the multilevel interpolation procedure in SZ3, followed by the quantization operation that produces the quantization indices. These concepts are closely related to our motivation and design of quantization index prediction methods and are essential for a better understanding of the proposed design.

Multilevel interpolation and strides: SZ3 decomposes the data into different levels with different strides and performs the interpolation level by level in a top-down order, and the stride is defined as the distance between two adjacent data points in a level. SZ3 starts data decorrelation from the highest level l with stride 2^{l-1} until the lowest level with stride 1 is reached. As long as the number of data points is sufficient, SZ3 performs intra-level interpolation to predict data at the current level, as detailed below.

Intra-level interpolation: We demonstrate how SZ3 performs linear interpolation inside a level on a $3 \times 3 \times 3$ cube using an example illustrated Figure 2. The same procedure applies to cubic interpolation, which requires a larger number of data points inside the level.

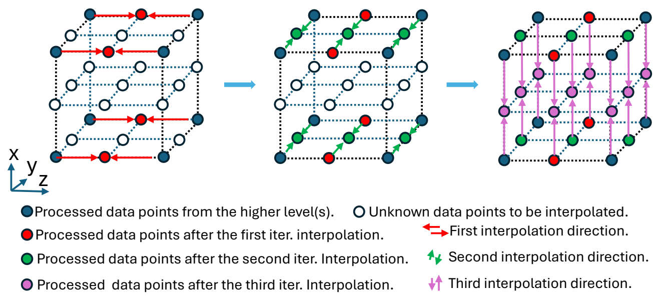

x
y
z
Processed data points from the higher level(s).
Unknown data points to be interpolated.
Processed data points after the first iter. interpolation.
First interpolation direction.
Processed data points after the second iter. Interpolation.
Second interpolation direction.
Processed data points after the third iter. Interpolation.
Third interpolation direction.

Fig. 2. Illustration of the intra-level linear interpolation in SZ3 on 3D data.

To complete the intra-level interpolation on 3D data, SZ3 needs to perform interpolation for 3 iterations, each along a different dimension. It starts with only 8 available data points (colored in blue) that have already been processed in the previous levels and aims to interpolate all the other data in the $3 \times 3 \times 3$ cube. The first interpolation is performed along the z direction, which produces prediction results on the 4 intermediate data (colored in red). These prediction results are then used to quantize the corresponding data and update them to decompressed values for future prediction. After that, the second interpolation is applied along the y direction to predict and then update the 6 intermediate data (colored in green). Finally, the third interpolation along the x direction produces the results for the remaining 9 data points (colored in magenta). If we look at the data updated by each interpolation direction separately, their strides are 2×2 (red data points), 1×2 (green data points), and 1×1 (magenta data points), respectively, assuming the base stride in the level is 1.

Quantization: Quantization is a key step in most error-controlled data compressors, which alters the data for better compressibility while enforcing error control. SZ3 adopts the following quantization function to quantize the difference between the original value d and its predicted value p: $q = round(\frac{d-p}{2\epsilon})$, where ϵ is the user-specified error bound and q is the resulting integer (a.k.a., quantization index). The decompressed data d' can be intermediately interpreted as $d' = p + 2q\epsilon$ after q is known, and it is easy to prove the enforcement of error bound as $|d - d'| \leq \epsilon$. Quantization is performed for each data point, so the final quantization index array will be of the same size as that of the original data. As such, its compressibility will significantly impact the overall compression ratio.

B. Visualization and characterization of quantization indices

We visualize the quantization indices of the four leading interpolation-based compressors using the Pressure2000 field from the SegSalt dataset (with dimensionality $1008 \times 1008 \times 352$; see Table III for details) as an example to understand the correlation in quantization indices. We align the PSNR of all the candidate compressors to 75, and the detailed statistics are listed in Table II. We use SZ3 as an example to describe how we select the visualization regions, and similar procedures apply to the other interpolation-based compressors. Under the specified setting, SZ3 performs interpolation first along the z direction, followed by y and x directions. To investigate the correlations that are perpendicular to the prediction direction, we visualize the slices along xy, xz, and yz planes, respectively, as shown in Figure 3. We select slice 211 along the xy plane, slice 221 along the xz plane, and slice 51 along the yz plane for demonstration purposes. This selection is without loss of generality since the selected slides have only medium entropy, as shown in Figure 4, which indicates that higher correlations are expected to occur in the other slices. We use stride to indicate the distance between two adjacent data points shown in the plot, and the zoomed-in regions are set to [450:550, 50:150], [400:600, 50:150], and [320:420,

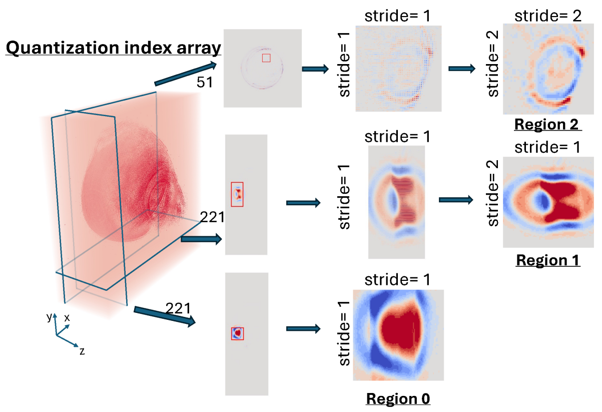


Fig. 3. Selection of the visualization regions along different planes. The visualization value range of the quantization index is set to [-8, 8].

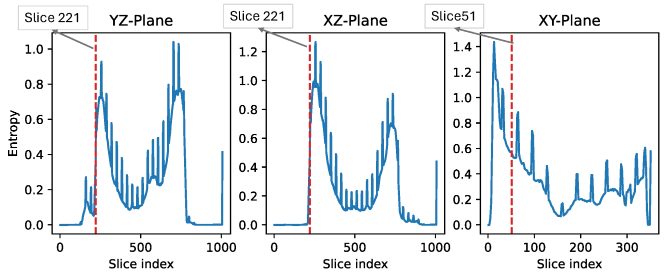


Fig. 4. Entropy of quantization indices by slice in three planes with SegSalt Pressure2000 data and SZ3. The stride in the X-axis is set to 2 to focus on the quantization indices produced by the last level of interpolation.

500:600], respectively, to locate critical regions with relatively high regional entropy. We referred to them as Region 0, Region 1, and Region 2 in the rest of the text.

TABLE II
COMPRESSION STATISTICS ON SEGSALT PRESSURE2000

	MGARD	SZ3	QoZ	HPEZ
Max Relative Error	0.0013	0.0006	0.00094	0.0011
PSNR	75.60	75.02	75.20	75.14
CR (original)	46.52	119.69	162.59	277.70
CR with QP*	54.73	144.29	179.55	286.64

* This indicates the compression ratios after applying the proposed quantization index prediction methods to be detailed in Section V.

According to Figure 3, the **clustering effect of quantization indices** can be easily observed in all three regions. Interestingly, the different regions exhibit clustering with different strides. This is mainly caused by the order of interpolation: when the interpolation is performed along z direction, only $1/8$ data is processed with a stride of 2×2 (analogous to the red data points in Figure 2); then $1/4$ data is handled at the stride of 1×2 (analogous to the green data points in Figure 2) for interpolation along the y direction, and $1/2$ data is interpolated with stride 1×1 (analogous to the magenta data points in Figure 2) along the x direction.

We then visualize the quantization index array in the three regions with the respective strides for the four interpolated-based compressors in Figure 5(a). According to this figure, MGARD and SZ3 exhibit the most severe clustering effect, especially in Region 0 and Region 1. QoZ is slightly better than these two probably because of its auto-tuning procedure, but it still has a large region of correlated quantization indices. HPEZ has the least correlation in the quantization index because of two reasons. On the one hand, it introduces a multi-dimensional interpolation scheme to leverage the correlation that is orthogonal to the interpolation direction; on the other hand, it adaptively selects the interpolation direction for each 32×32 block independently. In particular, the highlighted block in Region 0 is the only block in HPEZ that first interpolates along the z direction, which leads to the clustering effect; the other blocks are interpolated by the order $x \rightarrow y \rightarrow z$ and thus need a stride of 2×2 to observe the clustering effect. Such observations motivate us to explore a generic method to exploit the correlation in quantization indices to achieve higher compression ratios with interpolation-based compressors.

V. Quantization Index Prediction

In this section, we formulate the research problem for exploiting the correlation in quantization indices as a data prediction problem and design a generic algorithm to solve that. We then propose to use the Lorenzo predictor to perform the task and explore a suitable configuration to achieve high efficiency. Since this approach will not introduce any change in the decompressed data, it is expected to produce higher compression ratios without sacrificing compression quality in state-of-the-art interpolation-based scientific data compressors.

A. Formulation

Given the multi-dimensional quantization index array Q from an interpolation-based compressor, our research problem can be mathematically formulated as follows:

$$\min_{f}(H(f(Q)) \quad s.t. \quad f^{-1}(f(Q)) = Q,$$

where f is a reversible transform and H is the Shannon entropy. Let $Q' = f(Q)$ denote the quantization indices after applying the function f, our goal is to minimize the entropy $H(Q')$ as much as possible. This formulation is generic and can be easily plugged in as a postprocessing stage for quantization indices in existing interpolation-based compressors after f is identified.

We propose to use reversible prediction methods in prediction-based compressors [2]–[4] as candidates of f, because they have proven to be effective in data decorrelation. As such, the original problem reduces to a quantization index prediction (QP) problem.

B. Algorithm and implementation

We describe our lightweight QP algorithm in Algorithm 1 for any subroutine `quant_pred` that can predict the current quantization index using prior processed ones, which incurs minimal modification to the base compressor. While we present the algorithm using SZ3 as the base compressor,

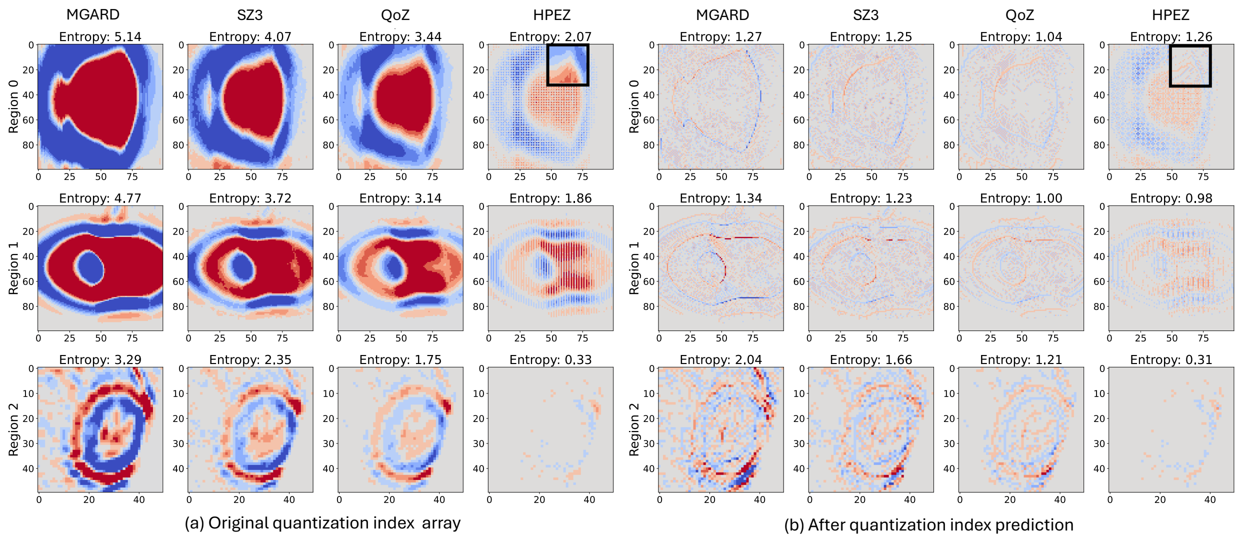


Fig. 5. Quantization index visualization of interpolation-based compressors on SegSalt Pressure2000 for the three regions with (a) original interpolation-based compressors and (b) the modified ones with quantization index prediction. Regions 1 and 2 are plotted with stride 1×2 and 2×2, respectively, and regional entropy is attached above each subfigure. The visualization value range is set to [-4, 4] for better comparison.

Algorithm 1 SZ3+QP: SZ3 COMPRESSION WITH INTEGRATED QUANTIZATION INDEX PREDICTION

Input: input data d of size n, error bound eb, interpolation predictor p, linear-scaling quantizer q, entropy encoder enc, lossless encoder l
Output: compressed data cc

```
1: init($Q$, $n$) /*allocate memory for quantization index array*/
2: init($Q'$, $n$) /*allocate memory for updated quantization index array*/
3: for $i = 1 \to n$ do
4:     $p \leftarrow predictor$.predict$(d, i)$ /*perform prediction*/
5:     $Q[i], d'[i] \leftarrow quantizer$.quantize$(d[i], p, eb)$ /*perform quantization to obtain quantization index and decompressed data*/
6:     $d[i] \leftarrow d'[i]$ /*overwrite original data with decompressed data for future prediction*/
7:     Q'[i] ← Q[i] - quant_pred(Q[1:i-1]) /*record the difference between original and predicted quantization index*/
8: end for
9: $c \leftarrow$ allocate_memory()
10: $p$.save(c) /*save predictor*/
11: $q$.save(c) /*save data quantizer*/
12: $enc$.encode(Q', c) /*perform entropy encoding*/
13: $enc$.save(c) /*save encoder*/
14: $cc \leftarrow l$.compress$(c)$ /*perform lossless compression*/
15: return $cc$
```

the same procedure applies to all the other interpolation-based compressors, including MGARD, QoZ, and HPEZ. The key modifications are highlighted in blue. In particular, we initialize an array to store the transformed quantization indices Q' (line 2). Then, we iterate each data point to perform the prediction and quantization as SZ3 does, but add an additional step to predict the current quantization index $Q[i]$ using the currently processed data and record its difference (line 7). Note that the restriction on using currently processed data is required because this is only available information during decompression. While this step can be moved outside of the loop, we keep it there to perform the prediction in a level-wise fashion to make it consistent with the workflow of interpolation-based compressors. This incurs less performance overhead by reusing the cache to access the original quantization indices while delivering better efficiency by preventing cross-level prediction because different levels may have different error bounds [6], [20]. Finally, we use the entropy encoder to encode the updated quantization index array Q' instead of the original quantization index array Q (line 12) after storing the metadata for the predictor and quantizer, and pass the result to the lossless encoder for further size reduction.

C. *Prediction method and configuration exploration*

While many candidate prediction methods exist, we propose to leverage Lorenzo predictor [34], which is both computationally lightweight and proven to be effective for data decorrelation [4]. It is also easily parallelizable for lossless prediction with CPU multi-threading or GPUs, as has been demonstrated in [35]. Lorenzo aims to predict a data point using its neighboring points, and the key idea is to assume data points in a local neighborhood follow a specific multivariate function as shown in Figure 6. This function can be solved using the neighboring points that have already been processed and applied to the current data points to obtain the prediction. The analytical expression of prediction is surprisingly simple, since it only involves additions and subtractions of the neighboring data as noted in the figure. In the following, we explore three settings for the Lorenzo prediction on the quantization indices to identify the best-fit configuration. While we use the Pressure2000 field in SegSalt and the Velocityx field in Miranda (see Table III for details) as two representative data fields and SZ3 to demonstrate the design process of our quantization index prediction method, similar behaviors are observed in the all the other data fields that we test.

1) Prediction dimension: We first explore the efficacy of using 1D, 2D, and 3D Lorenzo predictors to predict the

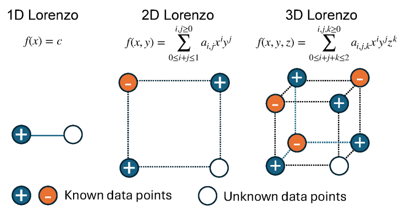

1D Lorenzo
$f(x) = c$
2D Lorenzo
$f(x,y) = \sum_{0\le i+j\le 1}^{i,j\ge 0} a_{i,j}x^i y^j$
3D Lorenzo
$f(x,y,z) = \sum_{0\le i+j+k\le 2}^{i,j,k\ge 0} a_{i,j,k}x^i y^j z^k$
Known data points
Unknown data points

Fig. 6. Lorenzo prediction.

quantization index, and the results of the compression ratio change are presented in Figure 7. For 1D prediction, we evaluate the prediction along three directions: 1D-Back for interpolation direction, and 1D-Top and 1D-Left for axes in the plane that is orthogonal to the interpolation direction. It is observed that 2D prediction yields significantly better compression quality when compared with the 3D one, although it uses a lower-order approximation function. This is because the points used to perform the 3D prediction are not contiguous along the interpolation direction when we do it in a level-wise fashion, which is also partially verified by the quality degradation when 1D prediction along this direction (1D-Back) is used. As such, we propose to use 2D Lorenzo for the quantization index prediction.

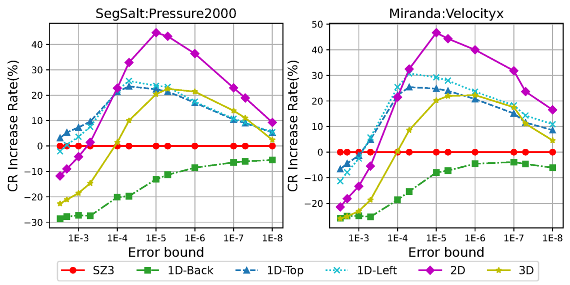


Fig. 7. Compression ratio increase rate for different prediction dimensions.

2) *Prediction condition:* While it is feasible to perform the quantization index prediction everywhere, it may not lead to the best overall compression quality. A typical example is the neighborhood of unpredictable data. Unpredictable data in these interpolation-based compressors refer to data points that fall out of the range of quantization, i.e., the obtained quantization integer exceeds the maximal allowable range. These data points are usually assigned to the same specific quantization value (e.g., the minimal quantization value in SZ3) and encoded separately. As such, the quantization indices of unpredictable data may not be related to their original values and thus could negatively impact the efficiency of quantization index prediction.

We define four conditions for performing quantization index prediction and evaluate how they impact the overall compression quality using the 2D Lorenzo predictor. In addition to taking unpredictable data into consideration, we also include additional constraints on the signs of the involved neighbors, which indicates whether the clustering phenomena occur around the current data point. The selected conditions are listed below, and the results are shown in Figure 8.

- Case I: No restrictions (perform prediction everywhere).
- Case II: Skip when unpredictable data is used for prediction.
- Case III: Case II holds, and the signs of left and top neighbors are the same.
- Case IV: Case II holds, and the signs of the three involved neighbors are the same.

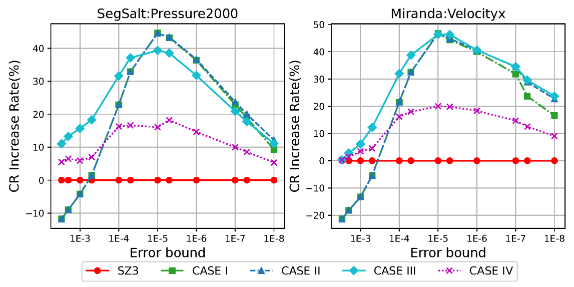


Fig. 8. Compression ratio increase rate of different condition cases.

According to the figure, Case III leads to the best overall compression. Case I suffers from slight degradation in compression ratios when the error bound is relatively small because the number of unpredictable starts to increase in those cases. Also, Case I and Case II show negative impacts on the compression ratio when the error bound is relatively large, because they perform the prediction even when the number of nonzero quantization indices is too small to provide useful information. In contrast, Case IV is so conservative that it seldom invokes the prediction procedure, which does not fully take advantage of the correlation in the quantization indices. Based on these findings, we use Case III in our algorithm.

3) *Prediction levels:* We also explore how to determine the levels to perform the quantization index prediction. This is inspired by the fact that high levels (1) possess only a small portion of data that may not have a huge impact on the overall compression ratios and (2) have large strides that may negatively impact the quality of prediction. As such, performing the prediction on the whole data may be inefficient in terms of both compression ratio and computational efficiency.

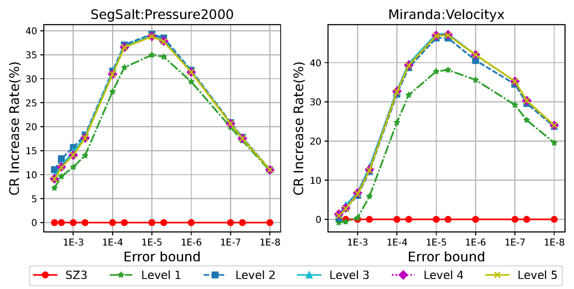


Fig. 9. Compression ratio increase rate of different start level settings.

We present the results with different starting levels in Figure 9. Substantial improvements are observed for levels 1

Algorithm 2 Best-fit `quant_pred` subroutine

```
1: Input: Quantization index array $Q$, current index $i$, 2D strides $s_1, s_2$,
   unpredictable quantization label $u$, and current interpolation level $l$.
2: Ootput: quantization index compensation factor $c$
3: if $l \leq 2$ then
4:     if $Q'[i-s_1] \neq u \wedge Q[i-s_2] \neq u \wedge Q[i-s_1-s_2] \neq u$ then
5:         if $(Q[i-s_1] > 0 \wedge Q[i-s_2] > 0) \vee (Q[i-s_1] < 0 \wedge Q[i-s_2] < 0)$
           then
6:             $c = Q[i-s_1] + Q[i-s_2] - Q[i-s_1-s_1]$
7:             return $c$
8:         end if
9:     end if
10: end if
11: return 0
```

and 2, which contain over 98% of the data points in total. Modest improvements or even degradation are noticed by including levels from 3 and above while introducing additional computational overhead. As such, we only perform quantization index prediction in level 1 and level 2 in our methods.

Best-fit configuration: Putting things together, we describe our final quantization index prediction algorithm in Algorithm 2. The inputs include the original quantization index array, the current data index, the strides in the prediction plane, the tag for unpredictable data, and the current interpolation level. When the current interpolation level is either 1 or 2 (line 3), we check if the unpredictable data appear in the neighborhood (line 4). If none of the 3 neighbors are unpredictable, we proceed to check the signs of the left and top neighbors (line 5). When the two neighbors' quantization indices have the same signs as Case III describes, we perform 2D Lorenzo prediction to compute and return a compensation (line 6), which will be applied to Q to obtain Q'. After integrating the QP algorithm and with this `quant_pred` subroutine to the four base compressors, the clustering phenomena can be significantly mitigated as shown in Figure 5(b).

VI. Evaluation

We integrate the proposed QP algorithm into four leading interpolation-based compressors—MGARD, SZ3, QoZ, and HPEZ—and evaluate them with all these base compressors and three state-of-the-art transform-based compressors—ZFP, TTHRESH, and SPERR—using 7 real-world datasets. This section details the evaluation results.

A. Experiment setup

We use 5 datasets from multiple domains as the benchmark datasets, as detailed in Table III. This includes Miranda from large turbulence simulations (hydrodynamics) [36], Hurricane from weather simulation [37], SegSalt from SEG/EAGE Salt and Overthrust models (geology) [38], SCALE from the SCALE-RM weather model (weather) [39], S3D from combustion simulation(chemistry) [40], CESM-3D from the CESM-ATM climate model (climate) [41], and RTM from a reserve time migration application (seismic) [42]. Note that we evaluate the first four datasets for generic comparisons and use RTM only for the data transfer evaluation. All the experiments are conducted on the Morgan Comput Cluster (MCC) [43], a medium-scale cluster where each node is equipped with two AMD EPYC 7763 64-core CPUs and 512GB DDR4 memory.

TABLE III
Benchmark datasets

Dataset	#Field	Dimension	Size	Type
Miranda	7	$256 \times 384 \times 384$	0.98GB	Float
Hurricane	13	$100 \times 500 \times 500$	1.21GB	Float
SegSalt	3	$1008 \times 1008 \times 352$	3.99GB	Float
SCALE	12	$98 \times 1200 \times 1200$	6.31GB	Float
S3D	11	$500 \times 500 \times 500$	10.24GB	Double
CESM-3D	33	$26 \times 1800 \times 3600$	20.71GB	Float
RTM	1	$3600 \times 449 \times 449 \times 235$	635.36GB	Float

B. Rate-distortion comparison with the base compressors

We first present the improvement of the proposed QP algorithm on the base compressors in terms of rate distortion. The results on the four datasets are displayed in Figures 10–13, respectively, with the maximum compression ratio increase rate and the corresponding PSNR annotated in the place where significant improvement is observed. Since QP increases the compression ratios without changing the decompressed data (which indicates the same PSNR), evaluation points in QP are always left shifts of the corresponding evaluation points in the base compressor in the rate-distortion graph.

According to these figures, improvement in compression ratios has been seen in all the base compressors on most of the datasets (except HPEZ on CESM and MGARD, SZ3, and HPEZ on Hurricane). Interestingly, the base compressors with the highest improvement vary with the dataset. Specifically, the maximum improvement on the Miranda dataset comes from SZ3, where a 45% increase in the compression ratio (from 26.95 to 39.25) has been observed when the PSNR is 101.23. On the SegSalt dataset, QP exhibits the best effect on QoZ with a 47% increase in the compression ratio (from 14.62 to 21.55) when the PSNR is 108.9. The improvements over MGARD become the most significant on the SCALE and CESM datasets, and the largest improvement appears on the CESM dataset, where the compression ratio is increased by 95% (from 23.59 to 45.99) when the PSNR is 75.76. QP has only minor improvement on HPEZ, which yields only 7%, 10%, 5%, and 0.31% in the four datasets. This is because QP mainly exploits the correlation along the plane that is orthogonal to the interpolation direction, while HPEZ has already taken advantage of this partially through its highly optimized multidimensional interpolation techniques. Nevertheless, QP still improves the quality of HPEZ and will not have any negative impact on the compression ratios.

While the improvement of QP is highly dependent on the datasets, one can observe a general trend from these figures: if we observe high improvement of QP for a base compressor at a specific evaluation point, it generally indicates that QP will have decent improvement for this compressor on this dataset. Similarly, if QP yields a minor benefit for an evaluation point, it is highly likely that QP cannot have high improvement for other evaluation points using this base compressor on this dataset. The only exception happens on SZ3 for the

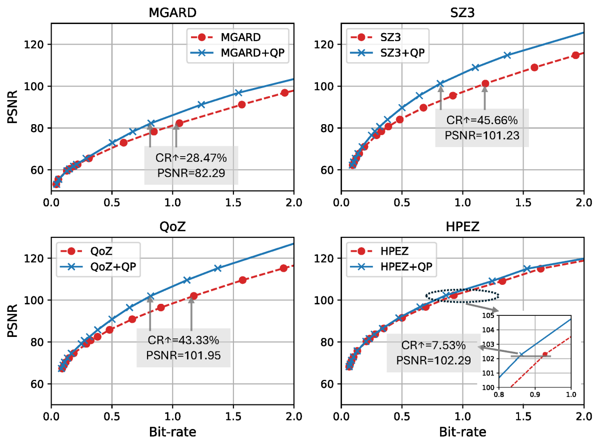


Fig. 10. Rate-distortion results on Miranda dataset.

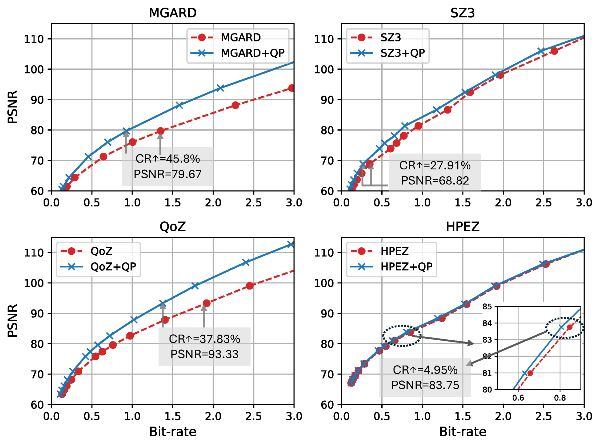


Fig. 12. Rate-distortion results on SCALE dataset.

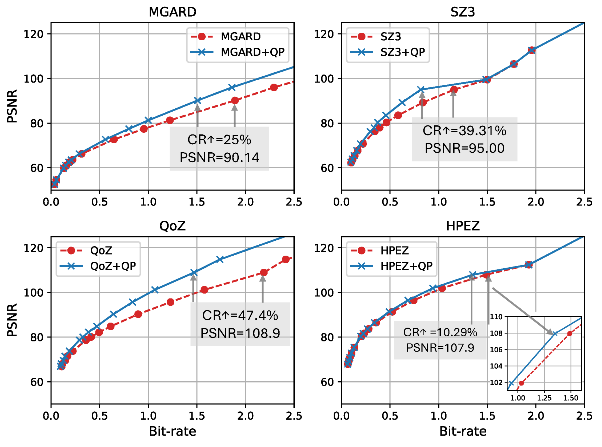


Fig. 11. Rate-distortion results on SegSalt dataset.

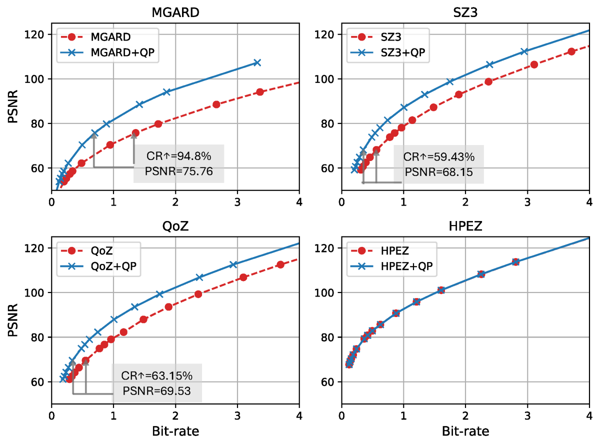


Fig. 13. Rate-distortion results on CESM dataset.

SegSalt data, where one can see a 39% improvement in the compression ratios when the PSNR is 95 but a modest gain when it is higher than 100. This is because SZ3 switches to the multidimensional Lorenzo predictor by its design, which does not have the clustering phenomena in the quantization index array.

C. Speed comparison with the base compressors

Since QP introduces an additional stage to predict the quantization indices during compression and recover them during decompression, it introduces additional overhead. In this section, we present the speed comparison of the base compressors and their QP-integrated versions to study the overhead in both compression and decompression. We showcase 3 error bounds, namely 1E-3, 1E-4, and 1E-5, on the four datasets because of limited space, and the speed under other error bounds shares similar trends.

We present the compression speed in Figure 16 and the decompression speed in Figure 17. According to these figures, one can see speed degradation on all the compressors after integrating QP, which is as expected. The lowest overhead is observed on MGARD, because the vanilla MGARD has relatively low compression and decompression speed and dominates the running time. For the other base compressors, different levels of speed degradations are seen on different datasets with varying error bounds. For instance, QP incurs around 20% compression overhead for SZ3 on all the evaluated error bounds, and this number increases to around 24% on SegSalt for 1E-3 and 1E-4. Note that QP has 0% compression overhead for SZ3 on SegSalt with 1E-5, because SZ3 has switched to Lorenzo under this error bound, where QP will not be invoked. On SCALE, the compression overhead of QP for SZ3 reduces from 17% to 5% because several fields (8 out of 12 when the error bound is 1E-5) have switched to the Lorenzo predictor. The compression overhead of QP is much more steady on QoZ because QoZ does not make the Lorenzo switch. The overhead across all the datasets and all the error bounds is between 14% to 19%.

Another worth noting point is that QP incurs a higher overhead on decompression than compression. This is mainly because decompression, especially at relatively high error bounds, has a higher speed than compression, and the overhead can vary with error bounds. For instance, the decompression overhead of QP on QoZ decreases from 44% to 27% when the

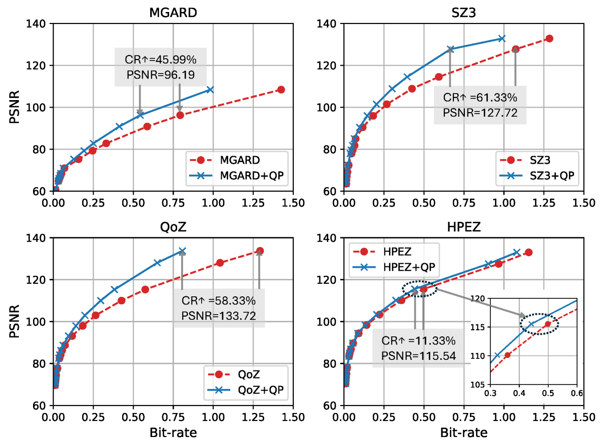


Fig. 14. Rate-distortion results on S3D dataset.

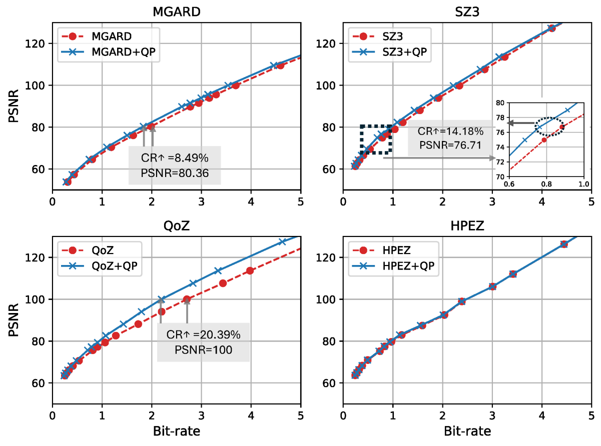


Fig. 15. Rate-distortion results on Hurricane dataset.

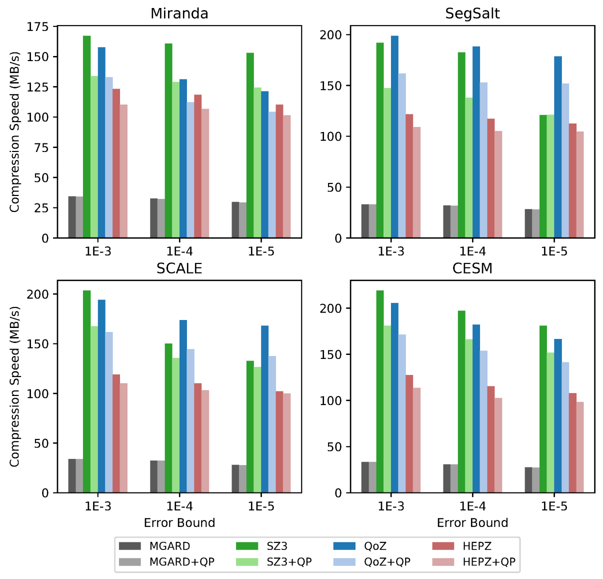


Fig. 16. Compression speed.

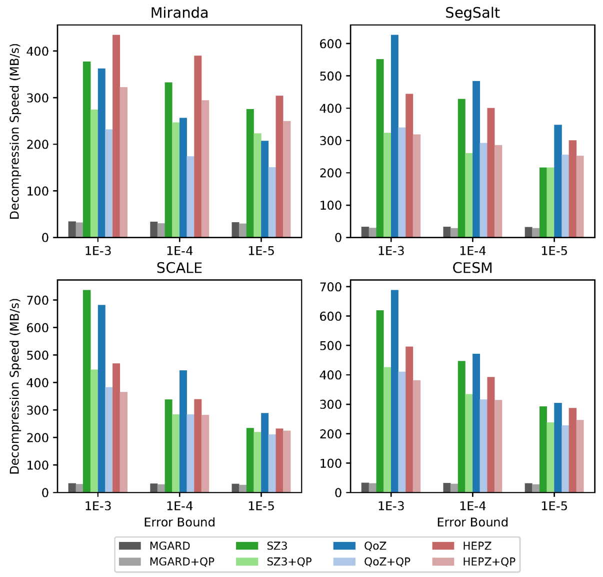


Fig. 17. Decompression speed.

error bound decreases from 1E-3 to 1E-5, because the vanilla QoZ has a much lower decompression speed at the error bound 1E-5 compared with that at the error bound 1E-3.

D. Comparison with state of the arts

We further compare QP-integrated interpolation-based compressors with state-of-the-art error-controlled lossy compressors in the community. In addition to MGARD, SZ3, QoZ, and HPEZ, we also compare with ZFP [10], TTHRESH [11] and SPERR [12] in terms of compression ratios and speed in Table IV.

From the table, it is observed that HPEZ+QP and SPERR always lead the compression ratios, but SPERR is relatively slow due to the wavelet transform it uses. While the improvement of QP to HPEZ is modest, we see huge benefits in integrating QP with MGARD, SZ3, and QoZ. Furthermore, integrating QP with SZ3 and QoZ has turned them into very competitive and even better compressors when compared with the state of the arts. For instance, QoZ+QP achieves similar compression ratios to HPEZ at the error bound 1E-5 on the SegSalt dataset while delivering 35% improvement in compression speed; similarly, it has the same level of compression ratios as SPERR at the error bound 1E-3 on the Miranda dataset while featuring 2.4× compression speed. This demonstrates the potential of QP in complementing interpolation-based compressors.

E. End-to-end data transfer

We further perform an end-to-end data transfer experiment in a parallel setting using the RTM dataset, to demonstrate how QP can improve the base compressors for data transfer tasks. The whole dataset has the dimension of $3600 \times 449 \times 449 \times 235$ with a total size of 635.54 GB, and the data transfer is performed between MCC at the University of Kentucky and Anvil [44] at Purdue University via Globus [45]. A vanilla

TABLE IV
PERFORMANCE COMPARISON WITH THE STATE-OF-ART COMPRESSORS (S_C: COMPRESSION SPEED, S_D: DECOMPRESSION SPEED, IN MB/S)

Miranda								
	1E-3				1E-5			
Compressor	**CR**	**PSNR**	S_C	S_D	**CR**	**PSNR**	S_C	S_D
MGARD	53.65	72.99	34.03	34.34	11.12	109.9	33.22	33.15
MGARD+QP	63.57	72.99	33.77	32.29	13.03	109.9	33.37	30.88
SZ3	168.0	71.05	167.3	377.3	20.11	108.9	152.9	275.4
SZ3+QP	183.55	71.05	133.9	274.2	29.03	108.9	124.4	223.0
QoZ	172.6	74.57	157.6	362.4	20.29	109.6	121.2	207.47
QoZ+QP	192.4	74.57	132.9	231.9	28.69	109.6	104.4	150.7
HPEZ	241.6	75.71	123.3	434.2	24.11	109.1	110.2	304.0
HPEZ + QP	**245.1**	75.71	110.2	322.3	25.75	109.1	101.6	249.6
ZFP	25.62	**95.64**	**426.0**	**907.5**	9.67	**134.4**	**280.8**	**559.2**
TTHRESH	137.6	68.62	30.15	109.3	24.74	110.0	22.93	67.54
SPERR	200.2	79.69	55.34	121.6	**32.74**	114.8	43.25	90.11
SegSalt								
	1E-3				1E-5			
Compressor	**CR**	**PSNR**	S_C	S_D	**CR**	**PSNR**	S_C	S_D
MGARD	49.45	72.68	32.86	33.10	9.49	109.8	32.50	32.13
MGARD+QP	57.15	72.68	33.02	31.56	11.36	109.8	32.19	29.56
SZ3	140.6	70.71	191.9	551.2	18.04	106.5	120.9	216.4
SZ3+QP	159.61	70.71	147.4	323.5	18.04	106.5	121.1	216.4
QoZ	143.4	73.71	198.9	626.1	14.62	108.9	178.7	348.6
QoZ+QP	167.4	73.71	161.7	339.7	21.55	108.9	151.9	255.6
HPEZ	239.3	75.35	121.7	444.1	21.49	107.9	112.6	300.6
HPEZ + QP	**245.5**	75.35	109.2	318.4	23.71	107.9	104.7	252.1
ZFP	24.96	96.97	**668.6**	**1032**	9.73	132.8	**462.8**	**648.5**
TTHRESH	99.53	**98.21**	11.49	99.25	11.77	**139.6**	9.74	48.28
SPERR	188.1	78.47	69.72	150.4	**26.33**	113.7	51.49	100.0

transfer of the original data using Globus takes 23 minutes and 29 seconds, which indicates a bandwidth of 461.75 MB/s.

The dataset is compressed in an embarrassingly parallel fashion. In particular, we split the entire dataset into 3600 slices along the first dimension and distributed the slices to different processors for independent compression. We report the time for compression, writing compressed data to the file systems, transferring the compressed data over Globus, reading decompressed from the file system, and decompression on a strong-scaling test with 225, 450, 900, and 1800 cores. The evaluation results on SZ3 and SZ3+QP are shown in Figure 18, and their respective compression ratios are 21.54 and 25.06. Both methods have the same PSNR of 108.51 in this setting.

According to the figure, one can clearly see that QP significantly reduces the data transfer time due to the reduced data size. Overall, integrating QP with SZ3 leads to a 16% performance gain in the end-to-end data transfer experiment, and this benefit is expected to grow when the size of data and the number of cores continue to increase. This may be impacted by the data transmission bandwidth, though, because higher bandwidth will lead to a shorter transmission time and, thus, fewer benefits in improving compression ratios. For instance, if the bandwidth of the transfer link doubles, the expected performance gain will decrease to 11%. However, considering that data movement time is usually the major performance bottleneck in many scientific applications, the proposed methods can have benefits for end-to-end data transfer in a wide range of use cases.

VII. Conclusion and Future Work

In this paper, we explore the correlation in the quantization index array for interpolation-based compressors, which is complementary to existing research on optimizing the interpolation methods in the data decorrelation stage. In particular, we

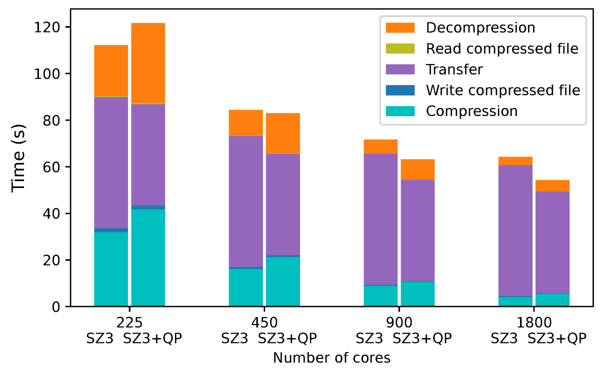


Fig. 18. End-to-end data transfer test results.

carefully characterize the quantization index array produced by four leading interpolation-based compressors and identify the clustering phenomena that can be utilized for potential optimizations. We then propose a generic quantization index prediction (QP) algorithm that is compatible with most interpolation-based compressors and explore the best-fit configuration to achieve high compression efficiency. We further integrate the QP algorithm into the four leading interpolation-based compressors and evaluate them using 7 real-world datasets. Experiments demonstrate that the QP algorithm can improve the compression ratios of the base compressors by up to 95%. Integrating it with SZ3 leads to up to 1.16× performance in a data transfer task.

Admittedly, the current proposed design of QP has certain limitations. For example, it is just oriented from interpolation-based error quantization, so it has not been well adapted to other archetypes of lossy compressors. Moreover, it exhibits limited compression ratio improvement under large error bound and low bit rate use cases. In the future, we will endeavor to address those limitations. Our detailed work plan includes 1) Proposing a more generalized design for compressors besides interpolation-based ones; 2) Guaranteeing the compression ratio improvement consistency in more diverse use cases; 3) Further optimizing the implementation of QP, reducing the computational overhead.

Acknowledgment

The material was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research (ASCR), under contract DE-AC02-06CH11357. This material was also supported by the National Science Foundation under Grant Nos. CSSI/OAC# 2311875, CSSI/OAC# 2104023, OAC-2330367, OAC-2311756, OAC-2313122, and OIA-2327266. We would like to thank the University of Kentucky Center for Computational Sciences and Information Technology Services Research Computing for its support and use of the Lipscomb Compute Cluster, Morgan Compute Cluster, and associated research computing resources.

References

[1] I. Foster, M. Ainsworth, B. Allen, J. Bessac, F. Cappello, J. Y. Choi, E. Constantinescu, P. E. Davis, S. Di, W. Di, H. Guo, S. Klasky, K. K. Van Dam, T. Kurc, Q. Liu, A. Malik, K. Mehta, K. Mueller, T. Munson, G. Ostouchov, M. Parashar, T. Peterka, L. Pouchard, D. Tao, O. Tugluk, S. Wild, M. Wolf, J. M. Wozniak, W. Xu, and S. Yoo, "Computing just what you need: Online data analysis and reduction at extreme scales," in *European conference on parallel processing*. Springer, 2017, pp. 3–19.

[2] S. Lakshminarasimhan, N. Shah, S. Ethier, S.-H. Ku, C.-S. Chang, S. Klasky, R. Latham, R. Ross, and N. F. Samatova, "Isabela for effective in situ compression of scientific data," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 524–540, 2013.

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

[4] 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.

[5] 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.

[6] 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.

[7] X. Liang, K. Zhao, S. Di, S. Li, R. Underwood, A. M. Gok, J. Tian, J. Deng, J. C. Calhoun, D. Tao *et al.*, "Sz3: A modular framework for composing prediction-based error-bounded lossy compressors," *IEEE Transactions on Big Data*, vol. 9, no. 2, pp. 485–498, 2022.

[8] 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 *2022 SC22: International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*. IEEE Computer Society, 2022, pp. 892–906.

[9] J. Liu, S. Di, K. Zhao, X. Liang, S. Jin, Z. Jian, J. Huang, S. Wu, Z. Chen, and F. Cappello, "High-performance effective scientific error-bounded lossy compression with auto-tuned multi-component interpolation," *Proceedings of the ACM on Management of Data*, vol. 2, no. 1, pp. 1–27, 2024.

[10] P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 12, pp. 2674–2683, 2014.

[11] 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, 2019.

[12] S. Li, P. Lindstrom, and J. Clyne, "Lossy scientific data compression with sperr," in *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2023, pp. 1007–1017.

[13] J. Liu, S. Di, K. Zhao, X. Liang, Z. Chen, and F. Cappello, "Faz: A flexible auto-tuned modular error-bounded compression framework for scientific data," in *Proceedings of the 37th International Conference on Supercomputing*, 2023, pp. 1–13.

[14] 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.

[15] ——, "Multilevel techniques for compression and reduction of scientific data—the multivariate case," *SIAM Journal on Scientific Computing*, vol. 41, no. 2, pp. A1278–A1303, 2019.

[16] ——, "Multilevel techniques for compression and reduction of scientific data-quantitative control of accuracy in derived quantities," *SIAM Journal on Scientific Computing*, vol. 41, no. 4, pp. A2146–A2171, 2019.

[17] X. Liang, S. Di, D. Tao, S. Li, B. Nicolae, Z. Chen, and F. Cappello, "Improving performance of data dumping with lossy compression for scientific simulation," in *2019 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE, 2019, pp. 1–11.

[18] Y. Liu, S. Di, K. Chard, I. Foster, and F. Cappello, "Optimizing scientific data transfer on globus with error-bounded lossy compression," in *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2023, pp. 703–713.

[19] R. Underwood, C. Yoon, A. Gok, S. Di, and F. Cappello, "Roibin-sz: Fast and science-preserving compression for serial crystallography," *Synchrotron Radiation News*, vol. 36, no. 4, pp. 17–22, 2023.

[20] X. Liang, B. Whitney, J. Chen, L. Wan, Q. Liu, D. Tao, J. Kress, D. R. Pugmire, M. Wolf, N. Podhorszki, and S. Klasky, "Mgard+: Optimizing multilevel methods for error-bounded scientific data reduction," *IEEE Transactions on Computers*, 2021.

[21] J. Chen, L. Wan, X. Liang, B. Whitney, Q. Liu, D. Pugmire, N. Thompson, J. Y. Choi, M. Wolf, T. Munson, I. Foster, and S. Klasky, "Accelerating multigrid-based hierarchical scientific data refactoring on gpus," in *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2021, pp. 859–868.

[22] J. Liu, J. Tian, S. Wu, S. Di, B. Zhang, Y. Huang, K. Zhao, G. Li, D. Tao, Z. Chen *et al.*, "cusz-i: High-fidelity error-bounded lossy compression for scientific data on gpus," *arXiv preprint arXiv:2312.05492*, 2023.

[23] 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*, vol. 33, no. 6, pp. 1201–1220, 2019.

[24] P. Jiao, S. Di, H. Guo, K. Zhao, J. Tian, D. Tao, X. Liang, and F. Cappello, "Toward quantity-of-interest preserving lossy compression for scientific data," *Proc. VLDB Endow.*, vol. 16, no. 4, p. 697–710, dec 2022.

[25] Z. Jian, S. Di, J. Liu, K. Zhao, X. Liang, H. Xu, R. Underwood, S. Wu, J. Huang, Z. Chen, and F. Cappello, "Cliz: Optimizing lossy compression for climate datasets with adaptive fine-tuned data prediction," in *2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2024, pp. 417–429.

[26] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[27] Y. Collet, "Zstandard - real-time data compression algorithm," http://facebook.github.io/zstd/, online.

[28] P. Deutsch, "Gzip file format specification version 4.3," 1996.

[29] F. Alted, "Blosc compressor," http://blosc.org/, online.

[30] P. Lindstrom, "Error distributions of lossy floating-point compressors," Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), Tech. Rep., 2017.

[31] G. K. Wallace, "The jpeg still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[32] M. Rabbani, "Jpeg2000: Image compression fundamentals, standards and practice," *Journal of Electronic Imaging*, vol. 11, no. 2, p. 286, 2002.

[33] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[34] L. Ibarria, P. Lindstrom, J. Rossignac, and A. Szymczak, "Out-of-core compression and decompression of large n-dimensional scalar fields," in *Computer Graphics Forum*, vol. 22, no. 3. Wiley Online Library, 2003, pp. 343–348.

[35] 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," in *Proceedings of the ACM International Conference on Parallel Architectures and Compilation Techniques*, 2020, pp. 3–15.

[36] Miranda application. https://wci.llnl.gov/simulation/computer-codes/miranda.

[37] H. I. dataset, http://sciviscontest-staging.ieeevis.org/2004/data.html, online.

[38] SEG/EAGE salt and overthrust models. https://wiki.seg.org/wiki/SEG/EAGE_Salt_and_Overthrust_Models/.

[39] G.-Y. Lien, "gylien/scale-letkf." [Online]. Available: https://github.com/gylien/scale-letkf

[40] J. Chen, "S3d-legion: An exascale software for direct numerical simulation of turbulent combustion with complex multicomponent chemistry," in *Exascale Scientific Applications*. Chapman and Hall/CRC, 2017, pp. 257–278.

[41] J. E. Kay, C. Deser, A. Phillips, A. Mai, C. Hannay, G. Strand, J. M. Arblaster, S. Bates, G. Danabasoglu, J. Edwards, M. Holland, P. Kushner, J.-F. Lamarque, D. Lawrence, K. Lindsay, A. Middleton, E. Munoz, R. Neale, K. Oleson, L. Polvani, and M. Vertenstein, "The community earth system model (cesm) large ensemble project:

A community resource for studying climate change in the presence of internal climate variability," *Bulletin of the American Meteorological Society*, vol. 96, no. 8, pp. 1333–1349, 2015.

[42] S. N. Kayum, T. Tonellot, V. Etienne, A. Momin, G. Sindi, M. Dmitriev, and H. Salim, "Geodrive-a high performance computing flexible platform for seismic applications," *First Break*, vol. 38, no. 2, pp. 97–100, 2020.

[43] "Morgan Compute Cluster," https://docs.ccs.uky.edu, 2023, online.

[44] X. C. Song, P. Smith, R. Kalyanam, X. Zhu, E. Adams, K. Colby, P. Finnegan, E. Gough, E. Hillery, R. Irvine *et al.*, "Anvil-system architecture and experiences from deployment and early user operations," in *Practice and experience in advanced research computing*, 2022, pp. 1–9.

[45] I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit," *The International Journal of Supercomputer Applications and High Performance Computing*, vol. 11, no. 2, pp. 115–128, 1997.