UNIVERSITY OF CALIFORNIA
RIVERSIDE

Task-Adaptive Scientific Error-Bounded Lossy Compression

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Jinyang Liu

June 2024

Dissertation Committee:

Dr. Zizhong Chen, Chairperson
Dr. Yan Gu
Dr. Rajiv Gupta
Dr. Daniel Wong
Dr. Zhijia Zhao

The Dissertation of Jinyang Liu is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

First, I would like to express my deepest gratitude to my advisor, Dr. Zizhong Chen. Dr. Chen led me to the world of research and my path of academic career. He provided his maximum support for me during the whole time of my PhD study, covering every aspect that I may need.

Secondly, I have had a great experience being mentored by Dr. Franck Cappello and Dr. Sheng Di during my internship at Argonne National Laboratory. Dr. Franck's academic insights are significantly impressive and have well guided my pursuit of excellence in research. Dr. Di is the most passionate researcher I have ever met with. He helped me a lot in my research, and I have also learned a lot from his work. I cannot be more grateful to Dr. Cappello and Dr. Di.

Moreover, I would also like to thank Dr. Yan Gu, Dr. Rajiv Gupta, Dr. Daniel Wong, and Dr. Zhijia Zhao for being my dissertation committee members. They have provided not only insightful comments on my dissertation but also support for my career path. I enjoyed and learned a lot from our communication.

Last, I also thank a lot to my colleagues and collaborators, including Dr. Kai Zhao, Dr. Xin Liang, Dr. Dingwen Tao, Dr. Sihuan Li, Dr. Yujia Zhai, Shixun Wu, Jiajun Huang, Zizhe Jian, Dr. Sian Jin, Dr. Jianan Tian, Yafan Huang, Boyuan Zhang, Dr. Robert Underwood, Arham Khan, and many more. Without your endeavors and insights, those great works would never be proposed.

Funding Acknowledgment I appreciate the funding supports from National Science Foundation (under Grants OAC-2003709, OAC-2104023, OAC-2311875, OAC-2311877,

and OAC-2153451), from the Exascale Computing Project 17-SC-20-SC (a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration), and from the U.S. Department of Energy Office of Science (under contract DE-AC02-06CH11357).

Publication Acknowledgment I acknowledge that part of the dissertation is from some of my other works, which have been published or released online previously.

- Certain contents in Chapter 1 was referred from arXiv preprint: 2404.02840 [25].
- Chapter 2 [65, 67] was published in the proceedings of SC22: International Conference for High Performance Computing, Networking, Storage and Analysis, and the proceedings of the ACM on Management of Data 2, no. 1 (2024).
- Chapter 3 [66] was published in proceedings of the 37th International Conference on Supercomputing, 2023.
- Chapter 4 [63] was published in the proceedings of the 2023 IEEE International Conference on Big Data (BigData), 2023.

To my parents and my family for all the support.

ABSTRACT OF THE DISSERTATION

Task-Adaptive Scientific Error-Bounded Lossy Compression

by

Jinyang Liu

Doctor of Philosophy, Graduate Program in Computer Science
University of California, Riverside, June 2024
Dr. Zizhong Chen, Chairperson

Modern scientific simulation applications are capable of generating petabytes of data outputs in several hours, necessitating effective compression methods for efficient data storage, analysis, and transmission. Error-bounded lossy compression has emerged as the most suitable strategy for managing these vast data volumes. It significantly reduces data size and controls point-wise data distortion according to user requirements, making it crucial for boosting the utility of scientific data.

However, existing error-bounded lossy compressors still have obvious limitations. On the one hand, they have not fully exploited the correlations in the input data points to optimize the compression rate-distortion. On the other hand, each of them cannot handle all of the diverse inputs with varying characteristics and accuracy requirements well by presenting consistent and satisfactory compression results. The core reason for the limitations is that most of the existing compressors feature fixed designs of compression techniques, frameworks, and/or pipelines, which makes them hard to adapt to diverse practical use cases and users' requirements.

Aware of those various requirements for scientific data compression in different real-world tasks, this dissertation explores multiple flexible error-bounded lossy compression techniques and strategies for scientific data across three dimensions.

First, for efficiency-aware compression tasks, this dissertation proposes an interpolation-based error-bounded lossy compressor, namely QoZ. QoZ can auto-tune its data predictor based on various quality metrics, meanwhile offering multiple optimization levels to balance compression ratio and speed for different use cases.

Secondly, for high-ratio compression, this dissertation presents FAZ, a hybrid error-bounded lossy compressor combining data transform and prediction techniques. FAZ dynamically constructs and tunes its compression pipeline for each data set, employing either wavelet-transform-based or interpolation-based data compression methods, to achieve optimal compression rate-distortion for diverse scientific datasets.

Lastly, the dissertation explores the application of Deep Learning in scientific data compression. Featuring a transformer-based super-resolution neural network for data prediction, SRN-SZ is presented in this dissertation, demonstrating superior performance on specific low-compressibility datasets compared to other scientific lossy compressors.

Contents

List of Figures **xii**

List of Tables **xiv**

1 Introduction **1**
1.1 Motivations and Challenges of Scientific Error-Bounded Lossy Compression 1
1.2 Scientific Error-Bounded Lossy Compression Strategies 4
1.2.1 Compressor archetypes 4
1.2.2 Compression techniques 5
1.3 Evaluation Metrics for Scientific Lossy Compression 9
1.3.1 Compression ratio and bit rate 9
1.3.2 Quality metrics 10
1.4 Contributions and Dissertation Organization 11
1.4.1 Quality-metric-oriented high-performance lossy compression 12
1.4.2 Adaptive high-ratio compression with pipeline auto-tuning 13
1.4.3 Improving low-compressibility scientific data compression with super-resolution neural network 14
1.4.4 Organization 14
1.5 Related Works 15

2 QoZ: Interpolation-Based High-Performance Scientific Error-Bounded Lossy Compressoion **16**
2.1 Overview 16
2.1.1 Motivations: Improving quality-oriented high-performance scientific error-bounded lossy compression 16
2.1.2 Challenges and contributions of QoZ 18
2.2 Related Work 20
2.3 Problem Formulation of Quality-Metric-Oriented Scientific Error-Bounded Lossy Compression 21
2.4 QoZ Design Framework 23
2.5 QoZ Interpolation-Based Data Predictor 26
2.5.1 Basic spline-interpolation-based predictor 27

2.5.2 Optimizations of spline-interpolation-based data prediction in QoZ . 27
2.5.3 Advancement of QoZ interpolation: QoZ 2.0 data predictor design . 33
2.6 QoZ Interpolation Auto-Tuning Module 41
2.6.1 Auto-tuning preparation: the efficient uniform sampling in QoZ . . . 41
2.6.2 Level-adapted selection of best-fit predictor 42
2.6.3 Quality metric oriented parameter auto-tuning 43
2.6.4 QoZ 2.0 auto-tuning module 46
2.7 Evaluations of QoZ 51
2.7.1 Experimental Setup 51
2.7.2 Experimental Results 54

3 FAZ: Pipeline-Auto-Tuning-Based High-Ratio Scientific Error-Bounded Lossy Compression 66
3.1 Overview 66
3.1.1 Motivations and challenges: Pursuing extremely high-ratio scientific error-bounded lossy compression 66
3.1.2 Contributions of FAZ 68
3.2 Related Work 69
3.3 Problem Formulation 71
3.4 Analysis of Existing Compressors 72
3.4.1 Introduction and investigation of high-ratio error-bounded lossy compression strategies 72
3.4.2 Limitations of the existing strategies 75
3.5 Overview of FAZ Design 77
3.5.1 Framework 77
3.5.2 Modules in FAZ 78
3.6 Pipeline Auto-Tuning in FAZ 80
3.6.1 Overall process of pipeline auto-tuning 81
3.6.2 Online statistical analysis for wavelets 82
3.6.3 Variance-based block-wise sampling 83
3.6.4 Rate-distortion tuning 84
3.7 Performance Evaluation 89
3.7.1 Experimental setup 89
3.7.2 Experimental results and analysis 91

4 SRN-SZ: Scientific Error-Bounded Lossy Compression with Super-Resolution Neural Network 100
4.1 Overview 100
4.1.1 Motivations and challenges: Addressing the low-compressibility scientific data compression with new technical routines 100
4.1.2 Contributions of SRN-SZ 102
4.2 Related Work 103
4.2.1 Traditional scientific error-bounded lossy compression 103
4.2.2 Deep-learning-based scientific lossy compression 104
4.2.3 Super-resolution neural networks 105

4.3 Problem Formulation and Backgrounds . . . 105
4.3.1 Research target . . . 105
4.3.2 Challenge for error-bounded lossy compression: low-compressibility datasets . . . 106
4.4 SRN-SZ Design Overview . . . 107
4.5 SRN-SZ Compression Pipeline . . . 109
4.5.1 Data grid sparsification . . . 109
4.5.2 Data grid expansion . . . 112
4.6 SRN-SZ Network Training . . . 116
4.6.1 Training data collection and preprocessing . . . 117
4.6.2 Domain-specific fine-tuning . . . 118
4.6.3 Denoise training with Gaussian random noise . . . 118
4.7 Performance Evaluation . . . 119
4.7.1 Experimental setup . . . 120
4.7.2 Evaluation results and analysis . . . 122

5 Conclusions and Future Work **129**
5.1 Conclusions . . . 129
5.2 Future Work . . . 131
5.2.1 New data prediction scheme for scientific compression . . . 132
5.2.2 Efficiency-aware compression auto-tuning . . . 132
5.2.3 More efficient neural networks for scientific compression . . . 133

Bibliography **134**

List of Figures

1.1 Visualized comparisons among Miranda-pressure data (a), non-error-bounded decompressed data (b) (by downsampling and tricubic interpolation), and error-bounded decompressed data (c) (by QoZ). The compression ratios for (b) and (c) are both 64. 2

2.1 The error-bounded quality-metric-oriented compression framework. 23
2.2 QoZ framework. 25
2.3 Illustration of the basic interpolation-based data predictor. 28
2.4 Visualization of original data and compression error (Hurricane Cloud). . . 29
2.5 Illustrating the key difference between QoZ vs. SZ3 (using 2D dataset as an example). 32
2.6 Illustration of 1D cubic spline interpolation. 34
2.7 Comparison of 1D-style interpolation and QoZ 2.0 multi-dimensional interpolation (an 2D example). 37
2.8 Comparison of QoZ 1.0 and QoZ 2.0 interpolation orders (Dim1 is the fastest-varying dimension). 39
2.9 Illustration of same-level cubic interpolation. 39
2.10 Illustrating QoZ data sampling using CESM-ATM dataset (field FSUTOA). 42
2.11 QoZ 2.0 auto-tuning module. 47
2.12 Illustration of dimension freezing. 49
2.13 Block-wise interpolation tuning. 50
2.14 Error bound-compression speed plots. 56
2.15 Rate-distortion (PSNR) plots of high-performance compressors. 59
2.16 SSIM of high-performance compressors. 60
2.17 Rate-PSNR of QoZ 2.0 and high-ratio compressors (QoZ 2.0's speed is 2x-17x of high-ratio compressors). 61
2.18 Parallel data transfer time approximation and decompression PSNR (simulation on the Anvil supercomputer, $p = 2048$, $s = 1GB/s$). 63
2.19 Visualization of SCALE-QS field (logarithmized) and the decompressed data. 65

3.1 Rate-distortion (PSNR) of compressors. In (a) and (b), SPERR is the best. In (c) and (d), QoZ is the best. 74

3.2 Interpolations (QoZ) vs Lorenzo (SZ2.1). 76
3.3 Comparing QoZ (interpolations), SPERR (CDF9/7), and other wavelets (higher PSNR is better). 76
3.4 FAZ compressor framework. 78
3.5 FAZ pipeline auto-tuning module. Yellow blocks indicate existing techniques and blue blocks are proposed ones. 81
3.6 Sym13 vs CDF9/7. Blue circles (red squares) are better compressed with CDF9/7 (Sym13). 83
3.7 The rate-distortion tuning process. Blue blocks are related to newly proposed techniques. 85
3.8 Comparison of estimated bit rate (in compression test) and real bit rate with different compression pipelines on SCALE-LetKF dataset (QS field). 88
3.9 Bit-rate adjustment on Hurricane dataset. 88
3.10 Rate-distortion (PSNR) of the compressors. 94
3.11 Rate-distortion (SSIM) of the compressors. 95
3.12 Visualization of data distortion under different compressors (SEGSalt dataset). 96
3.13 Ablation studies. 97
3.14 Parallel performance evaluation on Hurricane dataset (compression ratios on bottom-left). 99

4.1 Rate-distortion (PSNR) of several existing error-bounded compressors. . . . 108
4.2 SRN-SZ compression framework. 109
4.3 Data grid sparsification. 111
4.4 Data grid expansion. 112
4.5 HAT network. 114
4.6 3D super-resolution with 2D slices. 115
4.7 Interpolations in SRN-SZ. 116
4.8 SRN-SZ network training pipeline. 117
4.9 Histograms of decompression errors from SRN-SZ. 123
4.10 Rate-distortion evaluation (PSNR). 125
4.11 Visualization of reconstructed data (CESM-CLDHGH). 127
4.12 Ablation study for the domain-specific fine-tuning. 128
4.13 Ablation study for the denoise training. 128

List of Tables

1.1 Predictors used in different lossy compressors 6

2.1 Comparison cases of two compression results (B_I, M_I) and (B_{II}, M_{II}) for solution I and II under the error bound e 46
2.2 Information of the datasets in experiments 52
2.3 Execution speeds (MB/s per CPU core) with ϵ=1e-3 55
2.4 Compression ratios of high-performance compressors (SZ, ZFP, QoZ 1.1 and QoZ 2.0) 57
2.5 Compression ratios of QoZ 2.0 and high-ratio compressors (SPERR, FAZ, and TTHRESH) 58
2.6 Compression-based parallel data transfer throughput time (in seconds, 2048 cores, under PSNR=80). Inter-machine speed is the transfer speed of compressed data between 2 machines. 62

3.1 Information of the datasets in experiments 90
3.2 Compression ratios under the same error bounds 92
3.3 Sequential speeds (MB/s) with ϵ=1e-3 98

4.1 Information of the scientific simulations for training data of SRN-SZ 118
4.2 Information of the datasets in experiments 120
4.3 Compression ratio comparison based on the same error bounds 124

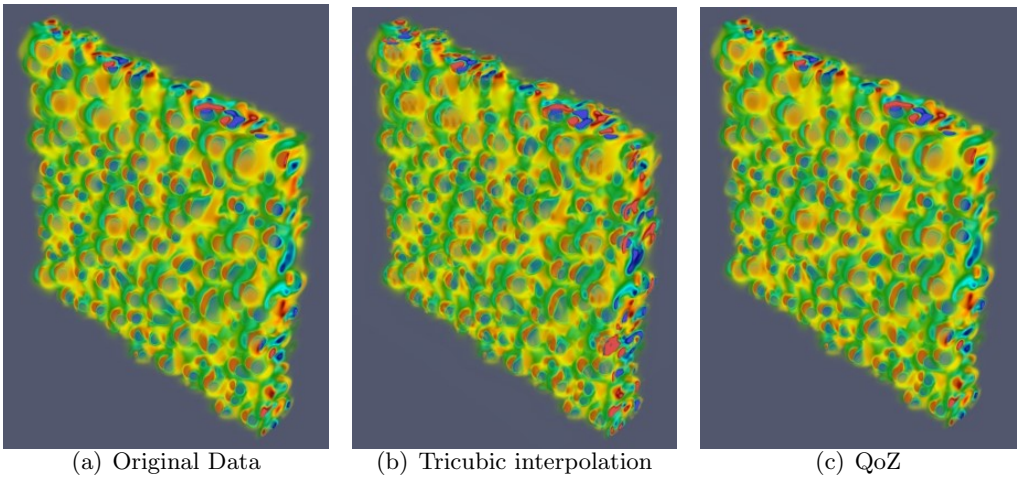
Chapter 1

Introduction

1.1 Motivations and Challenges of Scientific Error-Bounded Lossy Compression

The gigantic scale and exceptionally intense computation power of modern supercomputers have empowered the exascale scientific simulation applications to generate tremendous amounts of data in short periods, bringing up significant burdens for distributed scientific databases and cloud data centers. For instance, A one-trillion particle Hardware/Hybrid Accelerated Cosmology Code (HACC) [31] can harness approximately 22PB output data in a single simulation, and Community Earth System Model (CESM) [42] simulation may generate 2.5PB data for a simulation task [78]. To this end, error-bounded lossy compression techniques have been developed for those scientific data, and they have been recognized as the most proper strategy to manage the extremely large amount of data. The advantage of error-bounded lossy compression is primarily two-fold. On the one hand, it

can reduce the original data to an incredibly shrunken size which is much smaller than the compressed data size generated by a lossless compressor. In an analysis provided by [101], on representative scientific datasets, state-of-the-art lossless compressors can only present compression ratios [1] which are less than 3, but a typical error-bounded lossy compressor SZ3 can have tens or hundreds of compression ratio on those datasets. On the other hand, the error-bounded lossy compression can constrain the point-wise data distortion strictly upon the users' requirements. Figure 1.1 presents an example of compressing the pressure field of Miranda dataset [1] by both non-error-bounded method (downsampling and tricubic interpolation) and error-bounded method (by QoZ [65]). Under the same compression ratio of 64 (so the same compressed data size), the decompression data from error-bounded exhibits far better visualization quality than the one from tricubic interpolation.



(a) Original Data (b) Tricubic interpolation (c) QoZ

Figure 1.1: Visualized comparisons among Miranda-pressure data (a), non-error-bounded decompressed data (b) (by downsampling and tricubic interpolation), and error-bounded decompressed data (c) (by QoZ). The compression ratios for (b) and (c) are both 64.

[1] Compression ratio: The division of input data size by compressed data size.

Attributed to the strength of error-bounded scientific lossy compression, it is leveraged in diverse scientific data management use cases, such as reducing memory footprint, accelerating I/O, and reducing streaming intensity. In those different circumstances, the requirements for error-bounded scientific lossy compressors also diverge. For example, in cases of reducing storage space costs, users concentrate on optimizing compression ratios with considerable tolerance of compression speed. But in order to reduce streaming intensity, the compressors need to be fast enough to avoid introducing additional latency, whereas the compression ratios can just be moderate. Intuitively, it would be a great idea to propose a compressor design with both highly optimized compression ratios and fast compression speed, so that it can satisfy a large variety of compression requirements. However, designing a versatile error-bounded lossy compressor that delivers high compression ratios with sufficient efficiency is quite challenging. First, to reach a high compression performance, general techniques have to perform relatively simple data transform [61] or prediction within short-range areas [55, 101, 5], which cannot take advantage of long-range data correlations, thus leading to limited compression ratios. Second, to reach a high compression ratio, general solutions are applying sophisticated techniques such as wavelet transform [49] or higher-order SVD [11] on the full data input, which suffers from high computational costs, conflicting with the high-performance objective. Last, due to the diversity of scientific domains and varied characteristics, Many of the existing compressors failed to deliver stable and consistent compression outcomes on a large range of scientific datasets. Therefore, existing compressors have not covered all the requirements raised in essential practical use cases, which calls for new research.

In the rest of this chapter, this dissertation will first feature the technical backgrounds of scientific lossy compression for readers' better understanding of this dissertation, and then stress the research routines and contributions of this dissertation.

1.2 Scientific Error-Bounded Lossy Compression Strategies

Multiple existing works in the field of scientific error-bounded lossy compression have proposed a large diversity of compressor designs, and they contain various data processing techniques. As an important background, in this section this dissertation presents brief demonstrations for them.

1.2.1 Compressor archetypes

Existing error-bounded lossy compressors fall in diverse archetypes, such as prediction-based – SZ3 [101, 58], transform-based – ZFP [61] and SPERR [49], and dimension-reduction-based – TTHRESH [11]. In the following, those archetypes are featured.

Prediction-based compressor

A prediction-based scientific compressor [62, 47, 55, 58, 65] normally involves four steps in its pipeline: point-wise data prediction, quantization, variable-length encoding, and dictionary encoding. Data prediction is the most critical step in prediction-based compressors because higher prediction accuracy can significantly reduce the size of the encoded bitstream of prediction errors.

Transform-based compressor

Data transform methods (e.g. Discrete Cosine Transform and Discrete Wavelet Transform) have been adopted in several scientific lossy compressors [61, 48, 49]. After transforming the original data into a new domain, the transformed coefficients often exhibit substantially more compressibility than the original data. This fact enables the effective usage of data transforms into scientific data compression.

Dimension-reduction-based compressor

Dimension reduction techniques such as Higher-order singular value decomposition (HOSVD) can effectively decompose the high-dimensional (3D, 4D, or even more) input data to a set of 2D matrices and a small core tensor, meanwhile bringing low errors. Therefore, a scientific lossy compressor that integrates those dimension-reduction techniques can work significantly well for high-dimensional scientific datasets.

Deep-learning-based compressor

Several different types of deep neural networks have also been leveraged in scientific lossy compression. For example, some works [64, 33] apply autoencoders [12, 45] for creating compact representations of data. Others [32, 36, 71] propose predictive neural networks for prediction-based scientific lossy compression.

1.2.2 Compression techniques

Several important data processing techniques for scientific lossy compression are demonstrated as follows. They serve as critical modules in scientific error-bounded lossy

compression pipelines. For the conciseness of the dissertation, only the ones closely relevant to the proposed compressor designs in this dissertation will be illustrated.

Data prediction

In prediction-based scientific error-bounded lossy compressors, data prediction modules reconstruct the input data (both in compression and decompression) by different methods, and as mentioned before it is the most critical module in the pipeline. With this technique, the compressor can efficiently store the prediction error offsets instead of the original data values. Supported by the error quantization technique (to be detailed later), a high-accuracy data predictor can make the error offsets to be converted to mostly zeros and close-to-zero integers, which leads to greatly optimized compression ratios for them.

Many existing data prediction methods leveraged in scientific error-bounded lossy compressors are listed in Table 1.1. The most popular ones include Lorenzo predictor [24, 82], linear regression [55], and spline interpolation [101].

Table 1.1: Predictors used in different lossy compressors

Predictor	Compressor	Domain	# Values Used
Lorenzo	SZ1-3 [24, 82, 58], FPZIP [62]	General	1 or 3 or 7
Mean-value	SZ2 [55]	General	Many
Linear Regression	SZ2 [55]	General	216
Spline Interpolation	SZ3 [58], QoZ [65], FAZ [66]	General	2 to 4
Scaled-Pattern	Pastri [30]	Quantum Chemistry	Many
Temporal Smoothness	MDZ [100]	Molecular Dynamics	1
Multi-level	MDZ [100]	Molecular Dynamics	Many
Mask-based	CliZ [40]	Climate	2 to 4

Error quantization

Quantization means mapping continuous values into countable sets each represented by a discrete value. Rounding real values into integers is a typical example. Error quantization is a popular technique for scientific error-bounded lossy compressors, especially prediction-based ones because it can be used for both converting the data reconstruction errors into easy-to-encode integers and guaranteeing the compression error bound. For example, given a data reconstruction error r and an error bound e, the linear-scale quantization computes the quantized error $q = Round(\frac{r}{2e})$, so that the difference between the recovered error $r^{'} = 2eq$ and r will no larger than e. According to this fact, an error-bounded lossy compressor can store q instead of r for adjusting the data reconstruction and respecting the error bound.

Wavelet Transform

Wavelet transform, specifically the hierarchical multidimensional discrete wavelet transform, is a useful data transform method for scientific data compression. In many cases, it can effectively de-correlate and sparsify the input data to coefficients with higher compressibilities. Example wavelet transforms leveraged in existing scientific lossy compressors are the CDF9/7 [21] wavelet in SPERR [49] and Sym13 [23] wavelet in FAZ [66]. In those compressors, the input data array is first preprocessed with wavelet transforms. Next, the transformed coefficient array is further encoded with certain encoding algorithms such as the SPECK [73] encoding algorithm for wavelet coefficients. The encoded bitstream usually exhibits a significantly reduced size compared with the original data. One core limitation of

wavelet transform is that the effective transforms often have a relatively high computational cost and therefore apparently slow down the compression process.

Lossless encoding

Lossless encoding is also a critical technique in error-bounded lossy compression that can help obtain a fairly high compression ratio in general because the intermediate data from the previous steps (such as quantized errors) tends to be very sparse. Applying lossless encoding on those intermediate data often brings much more reduced data size with zero information loss. The lossless encoders used in scientific lossy compressors include but are not limited to Huffman Encoding, Arithmetic Encoding, RunLength Encoding, Fixed-length Encoding, and Embedded Encoding.

Deep neural networks

Since neural-network-based compression has been well developed and practicalized for natural images and videos, several attempts have also been made to leverage neural networks for the lossy compression of scientific data. In neural-network-based scientific lossy compressors, the neural networks can serve as both data encoders and data predictors. They can also be either offline-pre-trained by pre-acquired datasets or online-trained by input data. Neural network-based compressors with online-trained networks can achieve much better compression ratios and/or distortions than offline-trained networks but suffer from lower throughputs due to the requirement of training for each separate input. Neural-network-based compressors with offline-trained networks are free from per-input training but also need to address the challenge of collecting trustworthy training datasets. No matter

which scheme is used, neural-network-based scientific lossy compressors still must overcome the limitation of low-speed neural networks, presenting a better balance of quality and performance to fit the practical usage in high-performance scientific computing systems.

1.3 Evaluation Metrics for Scientific Lossy Compression

To quantitatively analyze and evaluate scientific lossy compressors, several well-defined metrics have been widely adopted in the community of scientific data compression. There are two main types of them: one is the compression ratio and bit rate which represents the compressed data size, and the other contains multiple quality metrics for evaluating the decompression data quality.

1.3.1 Compression ratio and bit rate

Compression ratio and bit rate directly measure the size of the compressed data. Compression ratio is defined by the input data size divided by the compressed data size. Specifically, for input data X and compressed data Z, compression ratio ρ is:

$$\rho = \frac{|X|}{|Z|} \tag{1.1}$$

According to Eq. 1.1, a higher compression ratio means better (smaller) compressed size, and vice versa. In the visualization of experimental results, researchers often plot curves with another metric closely related to the compression ratio, namely the bit rate. Bit rate is defined by the average number of bytes used in the compressed data to store each data element for the input data, which can be expressed as (denote bit rate by b):

$$b = \frac{sizeof(x)}{|Z|} \tag{1.2}$$

in which x is an element of the input X, and sizeof() returns the byte size. Since the bit rate is reciprocal to the compression ratio, a lower bit rate is better.

1.3.2 Quality metrics

Quality metrics work for measuring different aspects of the decompression data quality by different comparison methods between the original data and the decompressed data. There are quite a few mathematically defined metrics to model the data quality, and the most representative and important ones among them are listed here.

PSNR

PSNR is a metric commonly used in the rate-distortion evaluation [83]. PSNR measures the data distortion (i.e., compression errors) with mean-square errors according to the following formula ($vrange$ means the value range = max(X)-min(X)):

$$PSNR = 20 \log_{10} \frac{vrange(X)}{\sqrt{mse(X, X')}} \tag{1.3}$$

SSIM

SSIM [93] is a significant metric commonly used to measure the visual quality of decompressed data. The formula for calculating SSIM with input data X and decompressed data $X^{'}$ is:

$$SSIM = \frac{1}{N} \sum_{i=1}^{N} SSIM_i(X, X') \tag{1.4}$$

$SSIM_i(X, X')$ is the calculation of SSIM for a local sliding window i, which is calculated as follows:

$$SSIM_i(X, X') = \frac{(2\mu_X\mu_{X'}+c_1)(2\sigma_{XX'}+c_2)}{(\mu_X^2+\mu_{X'}^2+c_1)(\sigma_X^2+\sigma_{X'}^2+c_2)} \tag{1.5}$$

where μ is the mean, σ is the standard variance/covariance. For details, this dissertation refers readers to read Wang et al.'s papers [93].

Auto-correlation of compression error (AC)

Auto-correlation of compression error (AC) is a very important metric concerned by many application users. It is defined as follows.

$$AC = \frac{E(e_i-\mu_i)(e_{i+k}-\mu_{i+k})}{\sigma^2} \tag{1.6}$$

where e_i denotes the compression error at data point i and k is the lag (or offset) used to calculate the auto-correlation. The lower the AC value, the higher the randomness of the error correlation at adjacent data points. In general, the users expect to have a random error correlation between adjacent data points (i.e., low AC values).

1.4 Contributions and Dissertation Organization

After deeply analyzing existing techniques and artifacts for scientific error-bounded lossy compression, this dissertation believes that the best strategy for designing an effective scientific error-bounded lossy compressor is to make it task-adaptive. In other words, its design should be based on the awareness of its potential use cases, and only with this can it be optimized in the most practical way for real-world usage. For example, there are several

important users' requirements for the scientific error-bounded lossy compression, for which the existing works failed to fill in the blanks:

- Maximally reducing the data size with fast enough speed (whereas existing high-speed compressors present under-optimized compression ratios);
- Optimize compression according to different quality metrics (whereas existing compressors only deliver fixed outputs with certain inputs and parameters);
- Present consistently excellent compression ratio among a diversity of datasets (whereas many delicately designed high-ratio compressors only perform well on a limited number of datasets);

To address the issues including but not limited to the above-mentioned ones, in this dissertation, several scientific error-bounded lossy compressors with heterogeneous designs and frameworks are proposed. They feature different trade-offs between compression ratios and speeds, covering most of the existing scientific data compression use requirements.

1.4.1 Quality-metric-oriented high-performance lossy compression

The existing error-bounded lossy compressors are all developed based on inflexible designs or compression pipelines, which cannot adapt to diverse compression quality requirements and/or metrics favored by different application users. To resolve this issue, in Chapter 2, this dissertation proposes a novel dynamic quality-metric-oriented error-bounded lossy compression framework, namely *QoZ*. The detailed contribution is multi-fold. (1) QoZ features a novel highly-parameterized multi-level interpolation-based data predictor with dynamic interpolation schemes, which can significantly improve the overall compression

quality with the same compressed size. (2) QoZ integrates an advanced auto-tuning module, which can auto-determine the critical parameters of its interpolation-based data predictor for optimizing the compression ratio and user-specified quality metrics during online compression. (3) QoZ has 2 major released versions (1.0 and 2.0) and both of them have been systematically evaluated by comparing its compression quality with multiple other state-of-the-art compressors on various real-world scientific application datasets. Experiments show that, compared with the second-best lossy compressor, QoZ 1,0 can achieve up to 70% compression ratio improvement under the same error bound, up to 150% compression ratio improvement under the same PSNR, or up to 270% compression ratio improvement under the same SSIM. Moreover, QoZ 2.0 can further improve the compression ratio by up to 140% under the same error bound, and by up to 360% under the same PSNR. In parallel data transfer experiments on the distributed database, QoZ 2.0 achieves a significant performance gain with up to 40% time cost reduction over other compressors.

1.4.2 Adaptive high-ratio compression with pipeline auto-tuning

Existing high-ratio scientific error-bounded lossy compressors are facilitated with highly advanced and complicated data processing techniques, but none of them consistently present outperforming compression ratios among scientific datasets of different characteristics. In Chapter 3, this dissertation develops FAZ, a flexible and adaptive error-bounded lossy compression framework, which projects a fairly high capability of adapting to diverse datasets. FAZ can always keep the compression ratio and quality at the best level compared with other state-of-the-art compressors for different datasets. Experiments show that compared with the other existing lossy compressors, FAZ can improve the compression ratio by

up to 120%, 190%, and 75% when setting the same error bound, the same PSNR, and the same SSIM, respectively.

1.4.3 Improving low-compressibility scientific data compression with super-resolution neural network

Among the diverse datasets generated by various scientific simulations, certain datasets cannot be effectively compressed by any existing error-bounded lossy compressors with traditional techniques. The recent success of Artificial Intelligence has inspired several researchers to integrate neural networks into error-bounded lossy compressors. However, those works still suffer from limited compression ratios and/or extremely low efficiencies. To address those issues and improve the compression on the low-compressibility data, in Chapter 4, this dissertation proposes SRN-SZ, which is a deep learning-based scientific error-bounded lossy compressor leveraging the hierarchical data grid expansion paradigm implemented by super-resolution neural networks. SRN-SZ applies the most advanced super-resolution network HAT for its compression, which is free of time-costing per-data training. In experiments compared with various state-of-the-art compressors, SRN-SZ achieves up to 75% compression ratio improvements under the same error bound and up to 80% compression ratio improvements over the second-best compressor under the same PSNR.

1.4.4 Organization

The next chapters of this dissertation are organized as follows: Chapter 2 proposes a high-performance interpolation-based scientific error-bounded lossy compressor QoZ, which is capable of auto-tuning its compression according to users' targets. Chapter 3 demon-

strates FAZ, a pipeline-auto-tuning-based scientific lossy compressor that achieves optimized compression ratios on a wide range of scientific datasets. In Chapter 4, this dissertation presents a novel exploration of leveraging deep learning techniques, specifically speaking the transformer-based super-resolution network, in scientific error-bounded lossy compression. Facilitated with the Hybrid-attention Transformer, the designed compressor SRN-SZ shows excellent compression outcomes for low-compressibility scientific data.

1.5 Related Works

The related works of this dissertation are categorized into different specific topics, and are covered in different sections of the following chapters, including Section 2.2, Section 3.2, and Section 4.2.

Chapter 2

QoZ: Interpolation-Based High-Performance Scientific Error-Bounded Lossy Compressoion

2.1 Overview

2.1.1 Motivations: Improving quality-oriented high-performance scientific error-bounded lossy compression

The first critical research problem this dissertation would like to address is establishing and optimizing the quality-oriented high-performance scientific error-bounded lossy compression. Existing high-performance error-bounded lossy compressors all have in-

flexible designs, which cannot adapt to users' diverse requirements for reconstructed data quality. In addition to the strict error-bound constraint, the users may care about the rate-distortion (i.e., the relationship between compression ratio vs. some specific quality metric value) according to their post hoc analysis. Rate distortion may involve different quality metrics in practice. For instance, peak signal-to-noise ratio (PSNR) [83] (equivalent with normalized root mean squared error (NRMSE)) is a common quality metric to assess the overall statistical distortion of the data [47, 82, 81, 10, 13, 75]. Structural Similarity Index (SSIM) [93] is a perceptual metric that quantifies the visualization quality for a reconstructed data snapshot, which has also been widely used to assess the reconstructed data quality [9, 10, 8, 13, 83]. Low auto-correlation (AC) of compression errors [83, 103] is often highly preferred by users because it is consistent with the white noise nature. In practice, under the same error bound, the reconstructed data generated by various lossy compression methods often exhibit different levels on these distortion quality metrics.

However, although some lossy compressors (such as MGARD [5] and Fixed-PSNR based compression [81]) support preserving different quantity of interest (QoI) metrics (such as L-infinity, L1-norm and L2-norm errors), they are just preserving a threshold of the metric and none of them can dynamically optimize the compression based on diverse quality metrics under a certain error bound. That is, given a particular error bound, each existing high-performance lossy compressor always generates fixed compression and decompression outputs, which leaves a significant gap for users to control the compression quality on demand. Moreover, due to the concern for compression efficiency, the data modeling and processing techniques integrated into existing high-performance compressors are presenting

a nature of under-optimization of data reconstruction quality, which also brings a problem to resolve. The research in this dissertation will target this two-fold research topic: First, it will improve the decompression data quality of high-performance error-bounded scientific lossy compression; Second, it will enable the scientific error-bounded lossy compression to support optimization of different quality metric targets.

2.1.2 Challenges and contributions of QoZ

This chapter proposes a novel quality-metric-oriented error-bounded lossy compression framework, namely *QoZ*, which faces several challenging issues. (1) Combining the user-specified quality metric with error-bounded lossy compression requires an in-depth investigation of various lossy compression models. (2) Based on a specified quality metric, determining which steps or what parameters in the compression are tunable and critical to the overall compression quality is non-trivial. (3) How to optimize the rate-distortion with respect to the user-specified quality metric is non-trivial, since in this case, the compression result regards a co-optimization of the compression ratio and the quality metric instead of just maximizing the compression ratio. A straightforward method is using trial-and-error search to run the compressor multiple times with different tunable parameters, which inevitably introduces expensive computation costs [89, 88]. To the best of the author's knowledge, the developed QoZ compressor is a fresh attempt to adaptively optimize compression quality based on different quality metrics online under a particular error bound.

In order to make QoZ feature both high compression ratios and satisfactory speeds, this dissertation has also developed a brand-new auto-tuning strategy and an anchor-based level-wise hybrid interpolation predictor. Integrating extensively optimized interpolation

predictors and auto-tuning modules, QoZ attains far better compression ratios and lower distortions than other high-performance error-bounded lossy compressors with limited compression speed degradation. QoZ is also substantially faster than other high-ratio compressors. It achieves optimized throughput performance in a variety of use cases such as parallel data transfer for large (distributed) databases. The contributions are attributed as follows:

- This dissertation carefully explores and designs the best-fit data predictor for building the error-bounded lossy compression framework QoZ. Founded on theoretical analysis and algorithmic optimizations, QoZ substantially upgrades the most critical step in the quality-oriented compression – interpolation prediction, leading to an immensely improved data prediction accuracy.

- This dissertation develops an efficient error-bounded lossy compression framework that can dynamically optimize different inclined quality metrics in online compression. To this end, it leverages multiple advanced techniques, including block-wise interpolation tuning, dynamic dimension freezing, and Lorenzo tuning, which can substantially improve the adaptability of the auto-tuning for compression across a broad spectrum of inputs.

- Solid experiments are performed using 6 real-world scientific datasets. QoZ significantly outperforms state-of-the-art high-performance error-bounded lossy compressors in terms of rate-distortion, while still having a decent speed beating other high-ratio compressors. Consequently, it achieves the best throughput in distributed data transfer over WAN based on the experiments. QoZ exhibits the least time cost in data transfer for most scientific datasets with up to 40% time reduction.

2.2 Related Work

In general, scientific data compression techniques can be divided into two categories - lossless compression and lossy compression. Examples of existing lossless compressors for databases are Gorilla [74] and AMMO [96] for time-series data, and traditional lossy data compression methods include ModelarDB [39, 44] for time-series data and [47, 98, 28, 51] for Geology spatial-temporal data. Besides that, error-bounded lossy compression has been preferred and crafted to serve various scientific data reduction applications [13] and scientific databases. To meet the requirement of scientists, the error-bounded lossy compression needs to constrain the point-wise compression errors within a certain value, which differs from compression techniques for traditional data such as JPEG-2000 [85] for image data and h.265 [79] for video data. The error-bounded scientific compressors are classified into four main categories: prediction-based, transform-based, dimension-reduction-based, and neural-network-based. They also essentially utilize approaches to manage the data distortion in line with user-specified error bounds.

The prediction-based compressors use data prediction techniques, like linear regression [55] and dynamic spline interpolations [101]. Well-known examples are SZ2 [55] and SZ3 [58, 101]. Transform-based compressors use data transformations to de-correlate the data, then switch to compress the more compressible transformed coefficients. ZFP [61], for example, is a typical example that employs exponent alignment, orthogonal discrete transform, and embedded encoding. SPERR [49], a more recent work, leverages wavelet transform for data compression. Dimension-reduction-based compressors apply dimension reduction techniques, with (high-order) singular vector decomposition (SVD) being a case

in point (for instance, TTHRESH [11]). Neural-network-based compressors [64, 29, 68, 33] utilize neural network models like the autoencoder family [12, 45, 46].

Each compressor has its strengths and weaknesses, depending on the nature of the input data and user needs. To enhance scientific error-bounded lossy compression, two emerging approaches are raised to further refine the specialization of the compressor or to boost its versatility. Regarding compressor specialization, MDZ [100], a prediction-based compressor, is specifically tailored for molecular dynamics simulation data. SZx [95] offers low-ratio lossy compression at incredibly high speeds. CuSZ [87], CuSZ+ [86] and FZ-GPU [97] delve into GPU-based scientific lossy compression to quicken the compression process. [41] aims at maintaining the quantities of interest (QoI) of the input data.

With all those evolving works taken into insight, there is still a lack of broad-spectrum scientific error-bounded lossy compressors that can achieve both top-tier compression quality and adequate compression speed. In this chapter, the proposed solution endeavors to fill this gap: it pursues both high compression quality (by optimizing the rate-distortion) and high execution throughput across a wide range of scientific datasets.

2.3 Problem Formulation of Quality-Metric-Oriented Scientific Error-Bounded Lossy Compression

In this section, the research problem of quality-metric-oriented scientific error-bounded lossy compression is mathematically formulated to clarify the research objective for this chapter. Basically, it is a dual-objective lossy compression problem: meeting the necessary condition (error-bound constraint) meanwhile optimizing the compression result

in terms of the user-specified quality metrics. For example, *Rate distortion* is a very common method to assess the lossy compression quality. *Rate* here refers to the *bit-rate*, which is defined as the average number of the bits used to represent a data point after compression. Obviously, the lower the bit rate, the better the compression result. *Distortion* measures the difference between the original data and the decompressed data, and in literature, it is mainly referred to as peak signal-to-noise ratio (PSNR) [83] (to be detailed later). Here, the concept of the rate-distortion is extended to fit more generic distortion metrics such as SSIM [93] and autocorrelation (AC) [83] of compression errors, which is a critical advancement to optimize compression quality based on user's requirement on data fidelity in practice.

The problem of quality-metric-oriented error-bounded lossy compression is formulated as follows. Given an input data array (denoted by X) and a user-specified absolute error bound e, the error-bounded lossy compression consists of a compressor C and a decompressor D. It generates the compressed data (denoted Z) and the decompressed data (denoted $X^{'}$), which strictly respects the error bound (denoted e) on each data point. For each data value d_i, $|d_i - d_i^{'}| \leq e$ must be satisfied, where $d_i \in X$ and $d_i' \in X'$ represent the original data value and decompressed data value, respectively. This chapter aims to develop a highly parameterized error-bounded lossy compression framework, which can auto-tune the parameters to obtain the best rate-distortion in terms of different quality metrics such as PSNR, SSIM, and AC (please refer to Section 1.3 for descriptions). This chapter denotes the error bound by e and the user-specified quality metric by T. For a specific parameter set θ and parameterized compressor C_θ and decompressor D_θ, QoZ can automatically determine θ according to the optimization problem formulated as follows:

$$\theta = \arg \underset{\theta}{\mathrm{OPT}}\ T(X, X^{'}, Z)$$

$$s.t.\ \ Z = C_\theta(X)$$

$$X^{'} = D_\theta(Z)$$

$$|x_i - x_i^{'}| \leq e, \forall x_i \in X$$

where OPT refers to a optimization operation (e.g., max, min) according to the specific quality metric. Serving for this research target, the proposed error-bounded quality-metric-oriented compression framework is demonstrated in Figure 2.1.

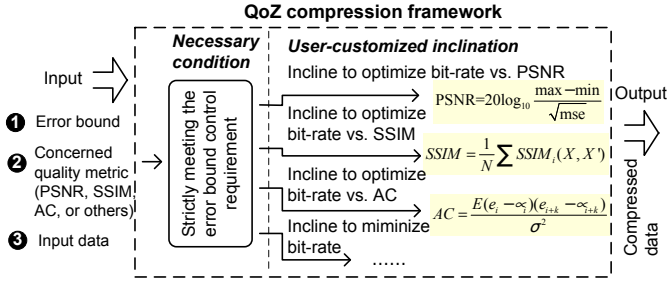


Figure 2.1: The error-bounded quality-metric-oriented compression framework.

2.4 QoZ Design Framework

This section proposed an overview of the QoZ compressor. As an interpolation-based scientific error-bounded lossy compressor, QoZ is designed for structured data grids in types of floating points and integers. QoZ is adaptive to either one-dimensional (1D) or multi-dimensional (2D, 3D, 4D ...) inputs, and exploits the dimension-wise spatial cor-

relations and smoothness of them. The compression framework of QoZ is illustrated in Figure 2.2. QoZ takes advantage of the SZ3 modular framework [58], which contains the auto-tuning module, data prediction module, error quantization module, Huffman encoding module, and the Zstd lossless module. The detailed demonstration of the QoZ compression pipeline is as follows:

- **Step 1: Auto-tuning**. With a user-specified quality metric optimization target, QoZ first auto-tunes its predictor configurations (to be featured in Section 2.6).
- **Step 2: Data prediction**: QoZ applies the auto-tuned data predictor on the whole input, acquiring the prediction errors.
- **Step 3: Linear quantization (error control)**: A linear error quantization module quantizes the data prediction errors in step 2 to control the element-wise decompression error. For example, for each data value x and its prediction $x^{'}$, the original error is $e = x - x^{'}$ and the quantized error e_q satisfies $|e_q - e| <= \epsilon$ (ϵ is the error bound). In this way, QoZ can use $x^{'} + e_q$ as the decompression of x which is bounded by ϵ.
- **Step 4: Huffman encoding**: The quantized prediction errors acquired from Step 3 are further encoded with Huffman encoding. A more concentrated distribution of quantization errors will lower the encoded tree size, therefore the reduction of prediction error is key to improving the compression ratio.
- **Step 5: Lossless postprocessing**: The encoded quantized errors and other metadata are losslessly compressed by Zstd [22] to further reduce the compressed size.

QoZ leverages existing modules in stereotype prediction-based error-bounded compression model (orange ones in Figure 2.2) and interpolation techniques (yellow ones in Figure 2.2). Most importantly, our QoZ framework introduces several new modules and significantly improved components (as marked in blue and pink), including interpolation designs and auto-tuning techniques. In the data prediction module and the auto-tuning module, new designs have been incorporated in QoZ to enhance the compression rate-distortion substantially. With those new designs, first, QoZ has significantly improved the interpolation-based data predictors in SZ3 [101, 58], introducing multiple refinements upon the existing dynamic spline interpolation; Second, QoZ facilitates a novel auto-tuning module for handling the optimization of interpolation configurations and boosting adaptability for more datasets. Third, the compression speed of QoZ still maintains at a high level, empowering it to well-fit efficiency-oriented tasks. Those newly proposed designs will be demonstrated in the next sections, including Section 2.5 and Section 2.6.

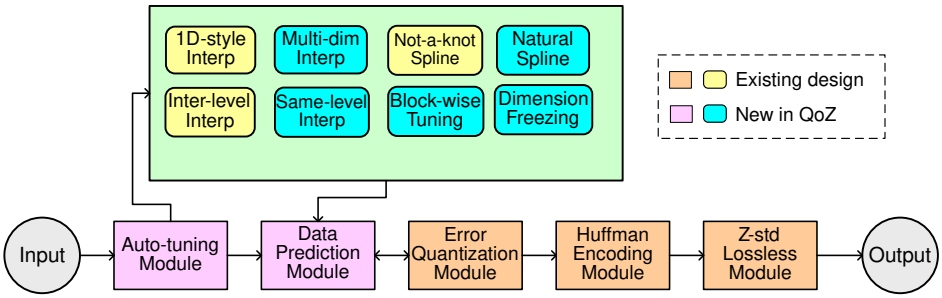


Figure 2.2: QoZ framework.

2.5 QoZ Interpolation-Based Data Predictor

This section presents the developed interpolation-based data predictor used in the QoZ framework. To optimize the compression with various compression quality requirements in terms of different quality metrics, the compression framework needs to be flexible enough to provide multiple compression results under the same error-bound constraint or the same compression ratio. As such, a flexible prediction method is critical to the QoZ framework.

In the design, QoZ data predictor falls in the archetype of spline-interpolation-based data predictor [58], because of the following two reasons:

- According to Zhao et al.'s study [58], the spline-interpolation-based predictor can obtain outstanding compression qualities over many other existing lossy compressors such as ZFP and SZ in most cases.
- The spline-interpolation-based prediction is executed based on a level-wise architecture, which provides great potential for parameterization and auto-tuning.

In what follows, this section first introduces the theoretical and technical fundamentals of the spline-interpolation-based data predictor, then describes the developed QoZ data predictor. These newly proposed designs are not only for the target of quality-metric-driven compression but also improve the data prediction accuracy in lossy compression, which will be detailed later.

2.5.1 Basic spline-interpolation-based predictor

Compared to the traditional extrapolation methods such as the Lorenzo predictor and regression models such as Linear regression, the interpolation-based predictor can improve the prediction accuracy prominently, especially for smooth datasets, as presented in Zhao et al.'s recent studies [58]. In the interpolation-based predictor, the data points in a data array are predicted based on a fixed interpolation method with varied strides, following a fixed propagation policy (as demonstrated in Figure 2.3 based on a 2D example). The entire prediction procedure starts with the first data point (see Stage 1 in the figure), which will be used to predict large-stride data points through the whole data array, followed by a linear-scale quantization to make sure the reconstructed value is close to the true data value within the expected error bound. Then, more data points would be predicted and quantized along another dimension alternatively, as demonstrated in the figure, until all the data points are covered (see Stage K in the figure). Note that each interpolation operation has to use the reconstructed data values (i.e., the approximated values after prediction+quantization on that data point) instead of the original data values, in order to make sure that the reconstructed data during the decompression would definitely respect the expected error bound.

2.5.2 Optimizations of spline-interpolation-based data prediction in QoZ

The developed interpolation-based data predictor in QoZ eliminates several critical limitations of the basic interpolation-based predictor.

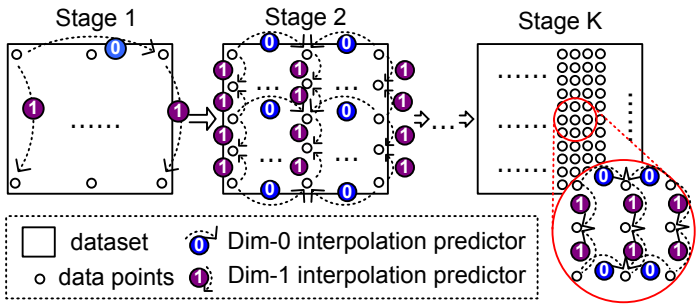


Figure 2.3: Illustration of the basic interpolation-based data predictor.

Anchor points: improving prediction by avoiding long-range interpolation

The first serious issue in the basic interpolation-based predictor is that it suffers from considerably low accuracy in long-range interpolation. As mentioned previously, the basic interpolation-based prediction method is executed in the order from large strides to small strides. Since it does not control the maximum stride length, the prediction accuracy would be fairly low when the interpolation spans a long distance in the data array. This situation turns even worse especially when the data exhibits different smoothness or patterns in different areas. In Figure 2.4, Here is an example to illustrate this serious situation of the SZ3 which adopts the basic interpolation-based prediction method. It can clearly be observed that there are more artifacts in the compression errors generated by SZ3 [58] than by SZ2.1 [55] (using block-wise linear regression and Lorenzo predictor for data prediction) under the same absolute error bound of 1E-2. This is mainly due to the fact that the interpolation method cannot predict the distant data values accurately in SZ3, while SZ2 always predicts data points with their neighbors.

QoZ leverages grid-wise anchor points to avoid those inaccurate long-range interpolations, which mitigates the inaccurate prediction issue effectively. Specifically, for

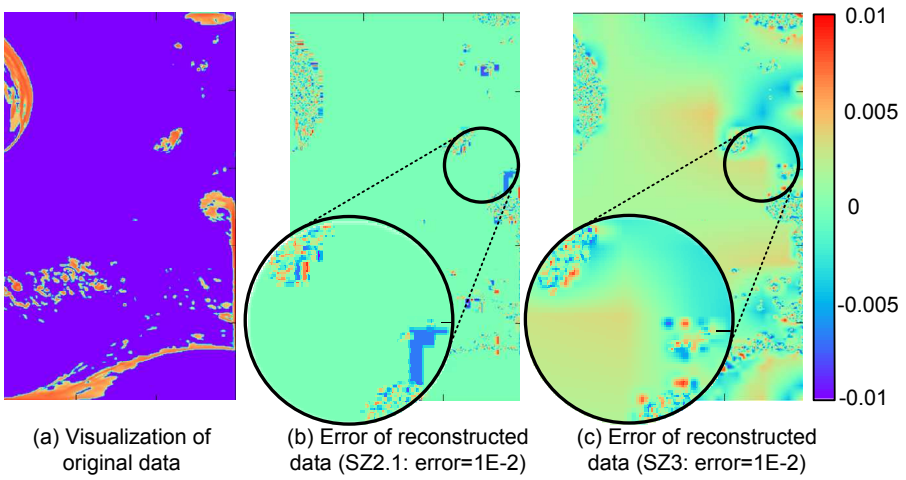


Figure 2.4: Visualization of original data and compression error (Hurricane Cloud).

interpolation, anchor points are data points that are considered to be known in advance and losslessly encoded and saved. These anchor points split the whole data array into many blocks and all other data points would be predicted/reconstructed by other points within a certain range, using the multi-level based interpolation method. Note that the storage overhead introduced by saving the losslessly compressed anchor points would be nearly negligible if an appropriate stride is set for the anchor point grid. The key advantage of utilizing anchor points is that it may greatly improve the quality of the distortion metric, which will be presented later on in Section 2.7.

Level-adapted interpolation and error bound auto-tuning

To have better prediction accuracy, the QoZ data predictor selects the best interpolation method at corresponding levels during the compression. As described previously, the basic interpolation-based prediction method can be decomposed into multiple stages (as shown in Figure 2.3). In QoZ design, These stages are performed on non-overlapping

levels: stage 1 corresponds to level K, $\cdots$, and stage K corresponds to level 1. There are two important takeaways regarding these different interpolation levels.

- These interpolation levels may have different data patterns or characteristics from each other, which motivates us to adopt different predictors on different levels.
- The compression quality of points at higher levels may affect the compression quality of points at lower levels significantly, in that the prediction of points always relies on the decompressed data points at higher levels.

Based on the above two critical takeaways, this dissertation develops the level-adapted interpolation-based predictor as follows.

First, the QoZ interpolation-based predictor adopts diverse interpolation methods at different levels. Specifically, the interpolation type includes both linear interpolation and cubic spline interpolation. As mentioned in Section 2.5.1, the multi-dimensional interpolation method is actually composed of multiple 1D interpolation operations. In a high-dimensional data array (such as 3D), even for the same interpolation type, each interpolation level may also involve multiple dimensions. As such, different sequences of the dimensions may lead to different prediction qualities. As an example, for 3D data, there are 6 different dimensional sequences based on the three dimensions (dim0, dim1, and dim2): 012, 021, 102, 120, 201, and 210. Accordingly, considering the two types of interpolation, there are a total of 12 prediction methods to select at each level.

Second, the QoZ interpolation-based predictor sets different error bounds for different levels. Such a design is motivated by the following important observation. In the whole interpolation-based prediction, a large majority of the data points (75% in 2D case

or 87.5% in 3D case) are at the lowest level (level 1), but they are mostly predicted by the reconstructed data points from higher levels: a total of 25% of the data points in 2D case or 12.5% of the data points in 3D case. Therefore, setting a smaller error bound at a higher level may preserve a very good overall prediction accuracy, which improves the compression quality in turn. Another important motivation is that having flexible and online-tuned level-wise error bounds makes the metric-driven optimization of lossy compression possible, with which the compressor can dynamically set error bounds to provide different compression results according to different optimization targets.

In Figure 2.5, the key differences between QoZ and SZ3 are shown using an example (based on a 2D data array). As shown in the figure, there are three key differences: (1) QoZ adopts anchor points that can minimize the error propagation in the interpolation methods from the top level to the lower levels; (2) QoZ dynamically tunes the parameters (i.e., error bounds) at different levels, which can improve compression ratio in turn; (3) QoZ uses a level-adapted interpolation method (highlighted in red font), which can further improve compression ratio in turn. In the example illustrated in the figure, under error bound 0.05 QoZ interpolates along dim 0 $\rightarrow$ dim 1 at level 2 with cubic spline interpolation, while its interpolation dynamically switches to dim 1 $\rightarrow$ dim 0 with linear spline interpolation under error bound 0.1 at level 1. How to select the best-fit prediction method will be discussed later on in Section 2.6 in detail.

In the implementation, there are two critical parameters (α and β) to tune the level-wise error bounds for the interpolation-based predictors. Specifically, given a global error bound e, the error bound for the interpolation level l is:

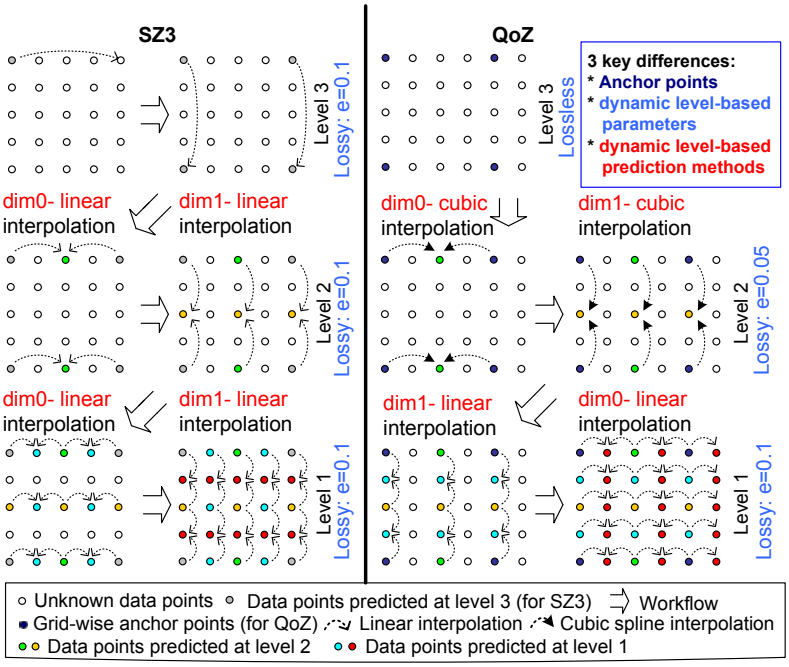


Figure 2.5: Illustrating the key difference between QoZ vs. SZ3 (using 2D dataset as an example).

$$e_l = \frac{e}{min(\alpha^{l-1}, \beta)} \ (\alpha \geq 1 \ and \ \beta \geq 1) \tag{2.1}$$

The availability and effectiveness of e_l is determined by the following policy:

- $e_l \leq e$, $\forall l$. This is to make sure compression errors for all data points must be within the user-set error bound e.

- $e_1 = e$. This means that the compression of the data points at level 1 which involves 75% (87.5%) of data points in the 2D (3D) input uses the maximum acceptable error bound (e) to optimize the compression ratio.

- $e_{l_1} \geq e_{l_2}$ when $l_1 < l_2$. Since every interpolation has to use lossy reconstructed data instead of the original data (to respect error bound strictly during decompression),

the data reconstruction errors would be propagated to all data points at lower levels. Thus, the error bound at higher levels should be smaller than those at lower levels.

It is worth noting that since QoZ is a dynamic quality-metric-driven lossy compressor, there would be multiple choices for α and β based on the same input dataset and user-set error bound, because of various user-specified quality metrics to target. Section 2.6.3 will present how values of α and β are determined during the online compression.

2.5.3 Advancement of QoZ interpolation: QoZ 2.0 data predictor design

With the above interpolation-based data predictor designs, QoZ has already achieved quite satisfactory data prediction accuracy and compression ratio, with which its first major version: QoZ 1.0 is established. In what follows, this section would like to propose several more advanced designs for interpolation-based data prediction, which are integrated into the second major version of QoZ (QoZ 2.0). They are optional modules for data prediction and provide additional trade-offs between compression ratio and speed.

More spline interpolation formulas

Interpolations in QoZ are based on certain spline interpolation formulas, which interpolate each data point with its neighbors along one dimension. Those splines include linear spline and cubic spline. Illustrated in Figure 2.6, the data value d_i on index i is going to be predicted by a prediction p_i with the known data points d_{i-3}, d_{i-1}, d_{i+1}, and d_{i+3} in its neighbours. The linear spline interpolation uses 2 of them with the following formula:

$$p_i = \tfrac{1}{2}d_{i-1} + \tfrac{1}{2}d_{i+1} \tag{2.2}$$

The cubic spline interpolation formulas leverage all the 4 neighbor points, and the formulas are deducted from 3 cubic spline functions ($f_1(x)$, $f_2(x)$, and $f_3(x)$):

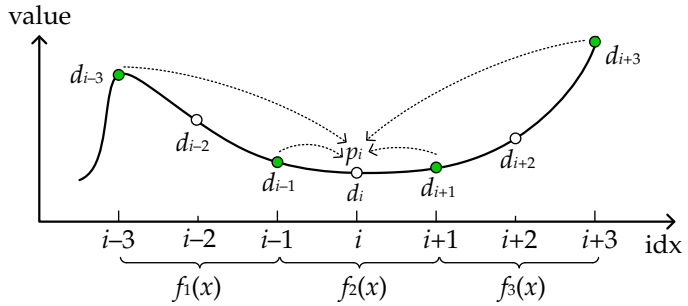


Figure 2.6: Illustration of 1D cubic spline interpolation.

$$
\begin{aligned}
f_1(x) &= a_1(x-(i-3))^3+b_1(x-(i-3))^2+c_1(x-(i-3))+\delta_1 \\
f_2(x) &= a_2(x-(i-1))^3+b_2(x-(i-1))^2+c_2(x-(i-1))+\delta_2 \\
f_3(x) &= a_3(x-(i+1))^3+b_3(x-(i+1))^2+c_3(x-(i+1))+\delta_3
\end{aligned}
\tag{2.3}
$$

The spline functions f_1, f_2, and f_3 have scopes of $[i-3,i-1]$, $[i-1,i+1]$, and $[i+1,i+3]$, respectively. The zero-order, first-order, and second-order interpolation conditions are shown as follows:

$$
\begin{aligned}
& f_1(i-3) = d_{i-3};\ f_1(i-1) = d_{i-1} \\
& f_2(i-1) = d_{i-1};\ f_2(i+1) = d_{i+1} \\
& f_3(i+1) = d_{i+1};\ f_3(i+3) = d_{i+3} \\
& f_1'(i-1) = f_2'(i-1);\ f_2'(i+1) = f_3'(i+1) \\
& f_1''(i-1) = f_2''(i-1);\ f_2''(i+1) = f_3''(i+1)
\end{aligned}
\tag{2.4}
$$

Since f_1, f_2, and f_3 have 12 coefficients in total and Eq. 2.4 only has 10 conditions, two more boundary conditions are needed. The traditional SZ3 and QoZ 1.0 cubic spline interpolation [101, 65] applies the following 'not-a-knot' conditions:

$$f_1'''(i-1) = f_2'''(i-1);\ f_2'''(i+1) = f_3'''(i+1) \tag{2.5}$$

Then with Eq. 2.4 and Eq. 2.5, the prediction value of p_i is:

$$p_i = f_2(i) = -\tfrac{1}{16}d_{i-3} + \tfrac{9}{16}d_{i-1} + \tfrac{9}{16}d_{i+1} - \tfrac{1}{16}d_{i+3} \tag{2.6}$$

However, there are other choices for the 2 boundary conditions, which may lead to different cubic spline interpolation formulas. QoZ 2.0 explores another set of boundary conditions: the natural spline condition, which is:

$$f_1''(i-3) = 0; f_3''(i+3) = 0 \tag{2.7}$$

Combining Eq. 2.4 and Eq. 2.7, the interpolation for predicting p_i is:

$$p_i = f_2(i) = -\tfrac{3}{40}d_{i-3} + \tfrac{23}{40}d_{i-1} + \tfrac{23}{40}d_{i+1} - \tfrac{3}{40}d_{i+3} \tag{2.8}$$

The experiments with multiple datasets under diverse error thresholds showed that Eq. 2.2, Eq. 2.6, and Eq. 2.8 have distinct advantages. In different cases, each of them is able to outperform others. Therefore, QoZ 2.0 employs all 3 of them and dynamically selects from them for each task.

Integrating multi-dimensional spline interpolation

In SZ3 and QoZ 1.0 interpolation-based compressors, for each data point, the interpolation is performed along a single dimension, so they need to switch the interpolation directions during this process and arrange an order for those directions. In the following

text, this interpolation method is named *1D-style interpolation*. As an example, in Figure 2.7 (a), the 1D-style interpolation first proceeds interpolations along Dim0, then performs the rest of the interpolations along Dim1.

Actually, The existing 1D-style interpolation has not fully exploited the multi-dimensional continuity and smoothness of input data arrays, because all the interpolations are constricted in a single-dimensional direction. To address this limitation, a new interpolation paradigm called multi-dimensional spline interpolation is proposed for QoZ 2.0, which can take better advantage of data correlation across multiple dimensions. As shown in Figure 2.7 (b), the multi-dimensional spline interpolation initially performs the 1D interpolations for some data points as there are only 1D neighbors at the moment, then it performs 2D interpolations for the remaining data points that already have neighbors in two dimensions. The multi-dimensional spline interpolation is symmetric across all the dimensions, meaning that it does not need a selection of dimensional order.

With the main concept of the QoZ 2.0 multi-dimensional spline interpolation in mind, two questions remain: how should QoZ 2.0 carry out the multi-dimensional interpolations specifically, and why does it outperform the 1D-style interpolations? To make answers to those questions, the QoZ 2.0 multi-dimensional interpolation is featured as follows. For each data point x, suppose X_i $(1 \leq i \leq n)$ are all the available 1D interpolation results for predicting x (which can either be linear interpolation or cubic interpolation and are along all dimensions), the multi-dimensional interpolation result $X^{'}$ is a linear-combination of X_i:

$$X^{'} = \sum_{i=1}^{n} \alpha_i X_i \quad (\sum_{i=1}^{n} \alpha_i = 1) \tag{2.9}$$

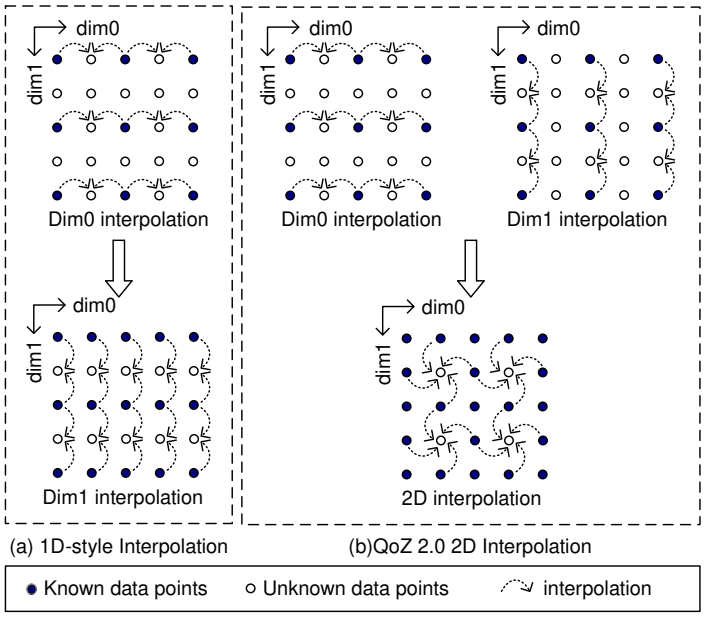


Figure 2.7: Comparison of 1D-style interpolation and QoZ 2.0 multi-dimensional interpolation (an 2D example).

Theorem 1 *With fine-tuned α_i, $X^{'}$ would have a no higher prediction error than that of the 1D-style interpolation X_i.*

Proof. Without loss of generality, we can regard $\{X_i\}$ and $X^{'}$ as random variables, in which $\{X_i\}$ are independent with each other. When dealing with smooth data inputs, the $\{X_i\}$ can be thought of as no-biased estimations of x, i.e. $E(X_i) = x$.

Now consider the $X^{'}$. Since $\sum_{i=1}^{n} \alpha_i = 1$, it is easy to know that $E(X^{'}) = x$, so $X^{'}$ is still a non-biased estimation of x. Because X_i are independent with each other, $(X^{'} - x) = \sum_{i=1}^{n} \alpha_i (X_i - x)$ follows the distribution of $N(0, \sigma^2)$, in which:

$$\sigma^2 = \sum_{i=1}^{n} \alpha_i^2 \sigma_i^2 \tag{2.10}$$

With the Lagrange method, based on the constraint $\sum_{i=1}^{n} \alpha_i = 1$,

$$\min \sigma^2 = \frac{\prod_{i=1}^{n} \sigma_i^2}{\sum_{i=1}^{n} \pi_i} \leq \min\{\sigma_1^2, \sigma_2^2, ...\sigma_n^2\} \ (\pi_i = \frac{\prod_{j=1}^{n} \sigma_j^2}{\sigma_i^2}) \tag{2.11}$$

The minimum is obtained when:

$$\alpha_i^* = \frac{\pi_i}{\sum_{j=1}^{n} \pi_j} \tag{2.12}$$

As such, we have proved that, if the $\{\alpha_i\}$ is selected based on Eq. 2.12, the prediction error variance of the multi-dimensional interpolation $X^{'}$ will be no larger than each of the 1D-style interpolation X_i according to Eq. 2.11. So, the average L-1 prediction error will also be minimized. ■

How to determine α_i^* (i.e. how to estimate σ_i^2) will be detailed in Section 2.6.

Levarging interpolation re-ordering

After the proposal of natural cubic spline and multi-dimensional interpolation, QoZ 2.0 also introduces interpolation re-ordering, which improves both prediction accuracy and speed. It includes two aspects: the fast-varying-first interpolation and same-level cubic interpolation. First, this section discusses the fast-varying-first interpolation. In the existing implementation of 1D interpolations, the interpolations are executed axis by axis on the input dataset, and along each axis, the interpolations are performed 'slice by slice'. The 'slice' here means a slice of the data array along an interpolation axis. Figure 2.8 (a) presents a 2D example for the order of interpolations adopted by QoZ 1.0 (and also SZ3): the interpolations are performed in the sequence of numbers (1, 2, 3, $\cdots$). For the interpolation along Dim0 in QoZ, it follows dim0-major order: the interpolation is executed along Dim0 with a higher preference compared with Dim1. However, when Dim1 is the fastest-varying-dimension , this interpolation order may fall into a bad cache usage because it is successively accessing data points located distantly in the memory. To resolve this issue, QoZ 2.0 re-

arranges the interpolation order, having the interpolations first move along the fast-varying dimension (the Dim1-major style as in Figure 2.8), as demonstrated in Figure 2.8 (b). The interpolation position first traverses through Dim1 and then moves along Dim0. In this way, the data points are accessed sequentially with shorter distances in the memory so that the cache usage can be optimized, greatly saving the memory access cost.

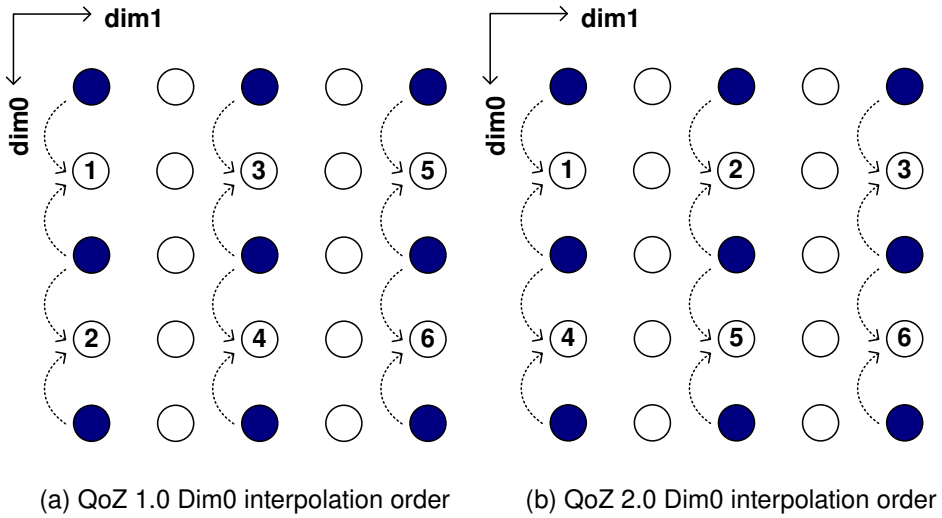


Figure 2.8: Comparison of QoZ 1.0 and QoZ 2.0 interpolation orders (Dim1 is the fastest-varying dimension).

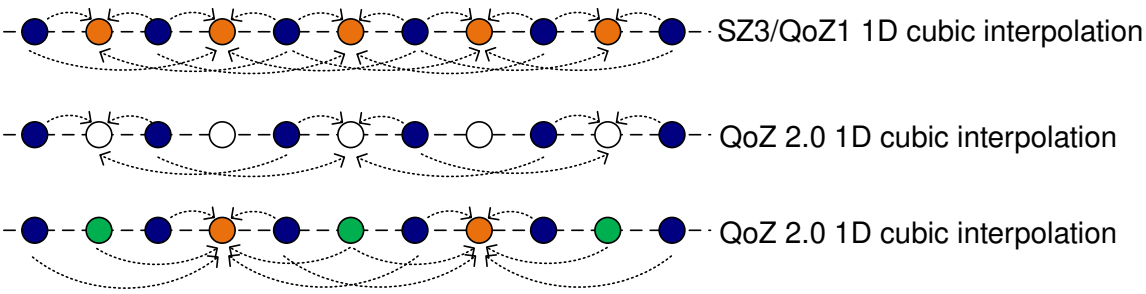


Figure 2.9: Illustration of same-level cubic interpolation.

Next, QoZ 2.0 develops a new same-level cubic interpolation, which can further improve prediction accuracy. In the traditional interpolation design [65, 101], at each interpolation level, the neighbor points of each data point to be interpolated are limited on the higher levels (interpolation levels with larger strides). For the 1D cubic spline interpolation applied on a data point with stride s, 4 neighbor points with distance s and $3s$ are used, which have been predicted on the higher interpolation levels. As shown in Figure 2.9 (note that s is the distance between each closest hollow and solid point), the first row shows this interpolation method, in which all the hollow data points (on the current interpolation level) are predicted by the solid data points (on higher interpolation levels). If it is able to include more neighbors for each point (for example, the 2 white points with a distance of $2s$ to it), the prediction accuracy can be improved. As illustrated in the 2nd and 3rd rows of Figure 2.9, instead of traversing through all the white data points in one step, QoZ 2.0 splits the 1D cubic spline interpolation into 2 steps. In the first step (the second row of Figure 2.9), half of the white points are interpolated by inter-level interpolation (the existing interpolation) with 4 neighbor points. In the second round, the rest half of the white points are interpolated by the same-level interpolation with 6 neighbor points for each, including points interpolated on higher interpolation levels and the current interpolation level. With this new interpolation, half of the data points are predicted with two more neighbor points to achieve better prediction accuracy. Similar to the deductions in Section 2.5.3, for a data point p_i, with its 6 neighbor points d_{i-3}, d_{i-2}, d_{i-1}, d_{i+1}, d_{i+2}, and d_{i+3} the same-level cubic spline interpolation formula would be the following two. Eq. 2.13 is for the not-a-knot cubic spline and Eq. 2.14 is for the natural cubic spline. The same strategy can also be

extended to the multi-dimensional interpolation, splitting it into 2 steps each with roughly halved data points.

$$p_i = -\tfrac{1}{6}d_{i-2} + \tfrac{4}{6}d_{i-1} + \tfrac{4}{6}d_{i+1} - \tfrac{1}{6}d_{i+2} \tag{2.13}$$

$$p_i = \tfrac{3}{62}d_{i-3} - \tfrac{18}{62}d_{i-2} + \tfrac{46}{62}d_{i-1} + \tfrac{46}{62}d_{i+1} - \tfrac{18}{62}d_{i+2} + \tfrac{3}{62}d_{i+3} \tag{2.14}$$

2.6 QoZ Interpolation Auto-Tuning Module

This section describes the auto-tuning and optimization strategies for the proposed QoZ data predictor in section 2.5. Regarding the above-mentioned parameterized level-wise interpolation-based predictor, there are several remaining great challenging issues to resolve. For example, what type of interpolation operation should be used on each interpolation level? How does it set prediction parameters (α and β for computing level-wise error bounds) in order to optimize the user-specified quality metric? To this end, QoZ also involves an efficient online tuning method, which not only can select the best-fit predictor and optimize the parameters at different levels based on diverse quality metrics but also has a very low execution overhead such that the overall compression performance can still be maintained well. The optimization strategies are detailed in the following text.

2.6.1 Auto-tuning preparation: the efficient uniform sampling in QoZ

In order to control the online analysis overhead, QoZ adopts a data sampling method, which plays an important role in reaching a good trade-off between the accuracy of the predictor/parameter selection and the computation overhead of this selection. To this end, the sampled data should be small enough to keep a very low time cost for the analysis

and they should be good representatives for the whole input data. As such, a uniform block-based sampling method is proposed, which can catch not only the pattern of the local area in the data but also the global picture of the data effectively. The sampling method is based on fixed block size and fixed sampling stride, as illustrated in Figure 2.10 using an example based on the CESM-ATM climate simulation dataset. The sampling rate (defined as the percentage of the number of sampled data points over the total number of data points) is determined by both block size and sampling stride. For instance, for a 2D dataset, if the block size is 4×4 and the sampling stride is 10, the sampling rate will be $\frac{4\times4}{10\times10}$=16%. In the parameter tuning over the sampled data, the prediction step is performed separately on each data block while the Huffman and dictionary encoding [22] are applied on the entire aggregated quantization bins for accurate bit rate estimation.

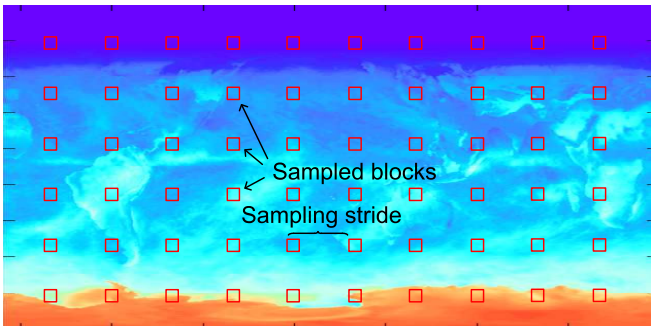


Figure 2.10: Illustrating QoZ data sampling using CESM-ATM dataset (field FSUTOA).

2.6.2 Level-adapted selection of best-fit predictor

Unlike SZ3 [58] which uses a fixed interpolation method at different levels throughout the whole data array, QoZ selects and applies the online-determined best-fit interpolation method on different levels with very limited computational overhead compared with the entire compression time. First, QoZ samples data blocks with the method introduced

in section 2.6.1. Next, on each interpolation level, QoZ runs a few trial compression runs with different interpolation/prediction methods (a.k.a., interpolators) over the sampled data blocks. As mentioned previously, the candidate interpolators involve two different types of interpolation and different dimension orders for a multi-dimensional input. As the total number of dimension orders grows very fast with the dimension number of the data (e.g. 6 for 3D), being consistent with SZ3 [58], QoZ only tests 2 dimension orders: dimension index increasing or decreasing, (e.g. 012 and 210 for 3D) as they cover the best choices in almost all cases. Then, QoZ compares the mean absolute prediction errors (L_1) and selects the one with the lowest mean absolute prediction error as the best-fit interpolator for the corresponding level. The reason it uses absolute error is that it is most closely related to the compression ratio of quantization bins, which was verified in [55]. The selection of interpolators does not need to be specifically tuned according to different quality metrics because its purpose is to minimize prediction errors, which both benefit bit rate and quality metrics. Since the interpolator selection is based on the sampled blocks, a tricky situation is that when the sampled block size is smaller than the anchor point stride, the blocks cannot cover some high interpolation levels. To solve this issue, QoZ applies the best-fit interpolator selected on the highest level of the sampled blocks to all higher levels.

2.6.3 Quality metric oriented parameter auto-tuning

This subsection presents the technical details of the user-specified quality-metric-oriented parameter auto-tuning algorithm in QoZ, which is critical to the user-specified quality-metric-driven lossy compression. It includes constructing parameter candidates, online compression result evaluation, and online parameter auto-tuning.

Preparation: constructing parameter candidates

In the step of quality-metric-oriented parameter auto-tuning, QoZ optimizes the level-wise interpolation error bound. The Formula 2.1 includes two critical parameters α and β to determine the error bound setting for each level. According to masses of experiments with different datasets across various domains, QoZ narrows the best parameter candidates as follows: α ={1, 1.25, 1.5, 1.75, or 2} and β = {1.5, 2, 3, or 4}, because these values cover the optimal or near-optimal settings of α and β in most cases without too many pairs to test with. In the algorithm (to be shown later), the optimal combination of the α and β will be determined online based on a lightweight compression result evaluation, which brings little computational overhead.

Operation: online compression quality evaluation

Online optimization of the rate-distortion based on a user-specified quality metric with different parameter settings is non-trivial. The key reason is that rate distortion refers to the relationship between bit-rate and the user-specified quality metric. Thus, accurately identifying the rate-distortion for a specific solution generally needs to collect quite a few compression results based on different compression ratios and various levels of data distortions. Specifically, given two different parameter sets (or solutions) each corresponding to a particular compression result, determining which one is better requires a meticulous analysis as described below. Suppose there are two solutions: setting I (α=1 and β=1.5) and setting II (α=2 and β=3). For a user-given error bound e, the setting I gets the compression result of {bit rate= B_{I}, PSNR=P_{I}}, and the setting II gets the

compression result {bit rate=B_{II}, PSNR=P_{II}}. If $B_{\mathrm{I}} > B_{\mathrm{II}}$ and $P_{\mathrm{I}} < P_{\mathrm{II}}$, it can be easily identified that II is better than I as II has lower bit rate (i.e., higher compression ratio) and higher PSNR (higher reconstructed data quality) meanwhile. However, if $B_{\mathrm{I}} > B_{\mathrm{II}}$ and $P_{\mathrm{I}} > P_{\mathrm{II}}$ or $B_{\mathrm{I}} < B_{\mathrm{II}}$ and $P_{\mathrm{I}} < P_{\mathrm{II}}$ (it is named sophisticated situation in the following text), additional analysis is needed to determine which solution is superior.

QoZ adopts an efficient online comparison method to determine the better choice for the sophisticated situation. Specifically, in this situation, QoZ uses another error bound (denoted as e') which has a small offset to e to perform a sampling-based trial compression for solution B, which can obtain another compression result (i.e., a pair of bit rate and quality metric (such as PSNR)): denoted as B'_{II} and P'_{II}. Then, QoZ can determine the better solution by checking the relationship between the solution I's result $(B_{\mathrm{I}}, P_{\mathrm{I}})$ versus the line constructed by $(B_{\mathrm{II}}, P_{\mathrm{II}})$ and $(B'_{\mathrm{II}}, P'_{\mathrm{II}})$. If the P_{I} is below the constructed line in space, QoZ asserts that the setting II is better, and vice versa. The second error bound e' used for computing B'_{II} and P'_{II} is set to $1.2e$ if $P_{\mathrm{I}} > P_{\mathrm{II}}$ or $0.8e$ if $P_{\mathrm{I}} < P_{\mathrm{II}}$. Such a design can make B_I lie in the range between B_{II} and B'_{II} in most of the cases based on experience, obtaining a very accurate judgment accordingly.

Table 2.1 summarizes all four situations for two comparative solutions I and II based on different compression results. Their compression results are denoted as $\{B_{\mathrm{I}}, M_{\mathrm{I}}\}$ and $\{B_{\mathrm{II}}, M_{\mathrm{II}}\}$, respectively, where M refers to the quality metric (such as PSNR, SSIM, AC). As shown in the table, QoZ can directly identify the better solution for cases 1 and 2. For cases 3 and 4, an additional sampling-based trial compression run would be done for solution II with another error bound. Since all the trial compression runs are on top

of sampled data, the computational overhead is very low, to be verified later. With this well-designed evaluation method, QoZ can traverse all the candidate parameter sets and select the best one for practical compression.

Table 2.1: Comparison cases of two compression results (B_I, M_I) and (B_{II}, M_{II}) for solution I and II under the error bound e

Case#	Case	Comparison
1	$B_{\mathrm{I}} <= B_{\mathrm{II}}$ and $M_{\mathrm{I}} >= M_{\mathrm{II}}$	I is better
2	$B_{\mathrm{I}} >= B_{\mathrm{II}}$ and $M_{\mathrm{I}} <= M_{\mathrm{II}}$	II is better
3	$B_{\mathrm{I}} > B_{\mathrm{II}}$ and $M_{\mathrm{I}} > M_{\mathrm{II}}$	compute $(B'_{\mathrm{II}}, M'_{\mathrm{II}})$ with sol II and $0.8e$ draw a line with the 2 points from sol II check whether $(B_{\mathrm{I}}, M_{\mathrm{I}})$ is above or below
4	$B_{\mathrm{I}} < B_{\mathrm{II}}$ and $M_{\mathrm{I}} < M_{\mathrm{II}}$	compute $(B'_{\mathrm{II}}, M'_{\mathrm{II}})$with sol II and $1.2e$ draw a line with the 2 points from sol II check whether $(B_{\mathrm{I}}, M_{\mathrm{I}})$ is above or below

2.6.4 QoZ 2.0 auto-tuning module

In addition to the fundamental auto-tuning strategies in QoZ 1.0, this dissertation develops several advanced auto-tuning sub-modules in QoZ 2.0, which do quite well in preserving and optimizing the compression quality by making the best use of the abundant interpolation options offered by QoZ 2.0. Figure 2.11 displays all the components and processes of the QoZ 2.0 auto-tuning module. This module inherits the interpolation error-bound tuning process from QoZ 1.0 [65], while substantially upgrading the QoZ 1.0 'global' interpolation tuning. Specifically, QoZ 2.0 exploits several brand-new processes: dynamic dimension freezing tuning, block-wise interpolation tuning, Lorenzo tuning, and a data sampling/analysis process supporting those tuning processes. The remainder of this section will present the detailed design of the auto-tuning-related components in QoZ 2.0.

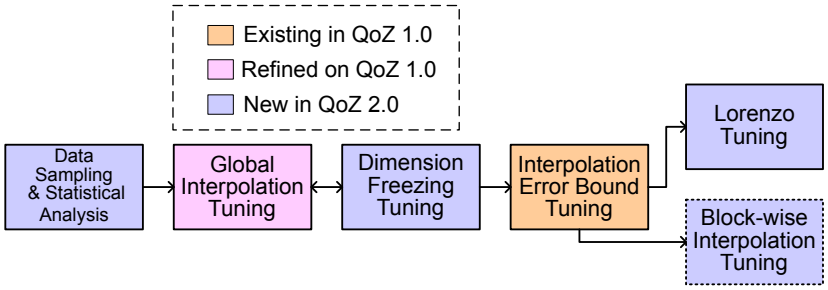


Figure 2.11: QoZ 2.0 auto-tuning module.

Data sampling and statistical analysis

The data sampling and statistical analysis is an auxiliary process of the QoZ 2.0 auto-tuning module. In this process, QoZ 2.0 uniformly samples a small portion from the full data input (based on a hyper-parameter with the default sampling rate of 0.2%), and then it performs the 1D interpolation (both linear and cubic) on those data points with their neighbors along all dimensions. Afterward, the mean square errors (MSE) of the interpolations along different dimensions can serve as the estimations of the interpolation error variances (σ_i^2) described in Section 2.5.3. Thus, it can be used to determine the most non-smooth dimension in the data for dynamic dimension freezing (Section 2.6.4) by selecting the dimension with the largest interpolation MSE.

Global interpolation tuning

The global interpolation tuning process in QoZ 2.0 is derived from the predictor tuning process proposed in QoZ 1.0, which aims to select the best-fit interpolation configuration from different choices Specifically, at each interpolation level, the global interpolation tuning process makes the following selection for the input data:

- **Existing in QoZ 1.0**: The order of interpolation (linear or cubic);
- **Existing in QoZ 1.0**: The dimensional order (only for 1D-style interpolation);
- **New in QoZ 2.0**: The type of cubic spline (not-a-knot or natural, only for cubic);
- **New in QoZ 2.0**: The interpolation paradigm (1D-style or multi-dimensional);
- **New in QoZ 2.0**: Applying inner-level interpolation or not (only for cubic);

Similar to QoZ 1.0, the sampled data are used for performing compression tests with all the available interpolation configurations. Then, QoZ 2.0 selects the interpolation configuration with the lowest average absolute prediction error as the final tuning result.

Dynamic dimension freezing

The dynamic dimension freezing in QoZ 2.0 is designed to avoid inaccurate interpolation predictions along non-smooth dimensions. For a multi-dimensional input data array, it may present fine smoothness along some of its dimensions but present bad smoothness along the other dimensions. In those cases, both the 1D-style and multi-dimensional interpolation will fail in achieving high prediction accuracy as they will involve interpolations along non-smooth directions. The dimension freezing is that, given one dimension, QoZ 2.0 sets anchor points along those dimensions with stride 1 (without intervals) and never performs interpolations along those dimensions. Figure 2.12 uses the interpolation on a 3D data block as an example of dimension freezing. For a clear view, only the 1D interpolations are shown. Figure 2.12 (a) is the normal 1D interpolations without a frozen dimension, and Figure 2.12 (b) is the 1D interpolations with a dimension frozen, in which

no interpolations are made along the frozen dimension. With this dynamic strategy, QoZ 2.0 does not require data smoothness along all dimensions to optimize its compression ratio. According to experimental results, compared to the highly improved prediction accuracy and greatly reduced quantization bin size, the storage overhead for additional anchor points is affordable. To determine whether to freeze a dimension and which dimension should be frozen, the auto-tuning module of QoZ 2.0 first specifies the most non-smooth dimension in the input data array in the statistical analysis (Section 2.6.4), then separately tunes 2 optimized interpolation configurations with/without this dimension frozen. If freezing this dimension presents a better compression ratio, QoZ 2.0 will freeze this dimension. Otherwise, no dimension will be frozen and the normal interpolation paradigm will be executed.

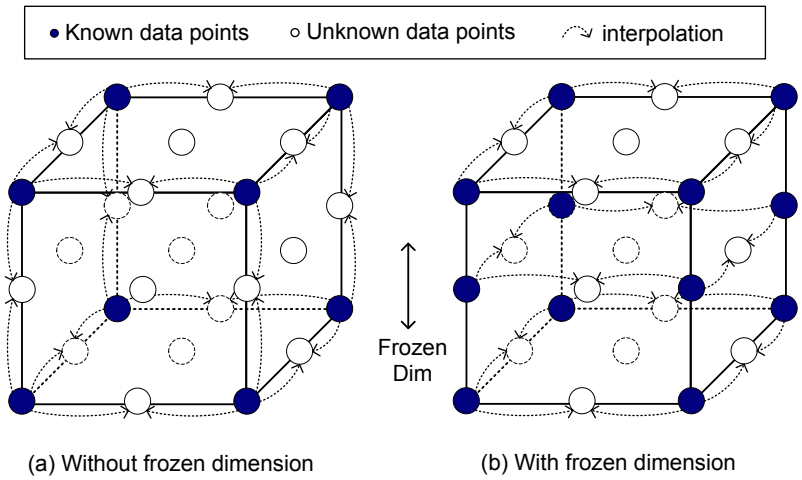


Figure 2.12: Illustration of dimension freezing.

Tuning with Lorenzo predictor

Leveraged in SZ3 but excluded by QoZ 1.0, the dynamic-order Lorenzo predictor designed in [103] is involved in QoZ 2.0, as it is still an essential supplement of interpolation-based predictors for high-accuracy low-compression-ratio cases [101, 58, 66]. In the auto-

tuning compression test process, after the auto-tuning module has acquired the optimized interpolation-based rate-distortion pair and its corresponding configuration, the auto-tuning module runs one more compression test with the Lorenzo predictor, then makes the selection between the interpolation-based predictor and the Lorenzo predictor according to the pre-given optimization target. Following the design in [66], a multiplicative coefficient is applied to adjust the bit rate estimation of the Lorenzo predictor.

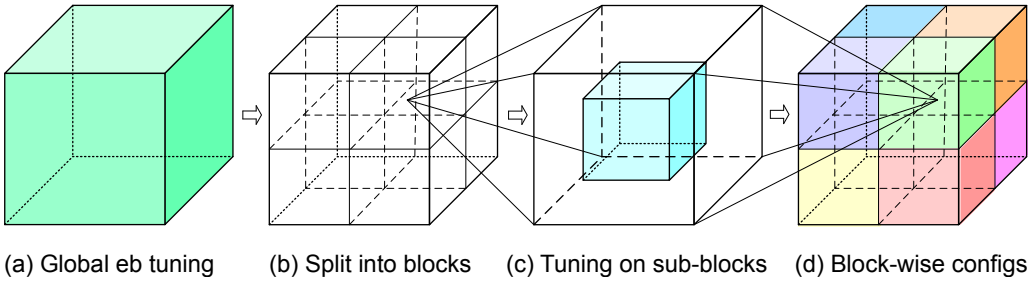


Figure 2.13: Block-wise interpolation tuning.

Block-wise interpolation tuning

If the interpolation predictor is finally selected after the Lorenzo tuning, the block-wise interpolation tuning will fine-tune the interpolation configuration separately on each data block. Various regions of the input data will exhibit different characteristics (such as dimension-wise smoothness), which makes them adapt to different interpolation configurations accordingly. To address this issue, QoZ 2.0 introduces the block-wise interpolation tuning process into its auto-tuning module, dedicated to identifying the best-fit interpolation configurations for diverse segments of the data. Figure 2.13 shows the details of the QoZ 2.0 block-wise interpolation tuning. First, after the auto-tuning has globally deter-

mined the level-wise interpolation error bounds (Figure 2.13 (a)), the input data array is split into blocks (Figure 2.13 (b)) of the same size. On each data block, a sub-block (in default has 4% of the full block size) is sampled out in the center of this block (Figure 2.13 (c)), and then the interpolation configuration for this block (Figure 2.13 (d)) is tuned by the compression tests performed on the sampled sub-block. The block size for block-wise interpolation tuning is a hyper-parameter in QoZ 2.0, and after primary experiments, the default value is set to 32.

2.7 Evaluations of QoZ

To verify the effectiveness and efficiency of QoZ, systematical evaluations of QoZ (including both of its 2 versions) together with five other state-of-the-art error-bounded lossy compressors are presented in this section.

2.7.1 Experimental Setup

Experimental environment and datasets

All the evaluation experiments are conducted on the Purdue Anvil supercomputer and the Argonne Bebop supercomputer. On the Anvil supercomputer, each computing node features two AMD EPYC 7763 CPUs with 64 cores having a 2.45GHz clock rate and 256 GB DDR4-3200 RAM. The computing node used on the Bebop has the Intel Xeon E5-2695v4 CPU with 64 CPU cores and a total of 128GB of DRAM.

In order to evaluate the compressors more comprehensively and systematically, 6 real-world scientific applications from diverse scientific domains that have been frequently

used for the evaluation of scientific data error-bounded lossy compression [102] are involved in the evaluation. The detailed information of the datasets is in Table 2.2. As suggested by domain scientists, some fields of the datasets listed above are transformed to their logarithmic domain before compression for better visualization.

Table 2.2: Information of the datasets in experiments

App.	# files	Dimensions	Total Size	Domain	Type
RTM [43]	37	449×449×235	6.5GB	Seismic Wave	Floating points
SEGSalt [3]	3	1008×1008×352	4.2GB	Geology	Floating points
Miranda [1]	7	256×384×384	1GB	Turbulence	Floating points
SCALE-LetKF [2]	12	98×1200×1200	6.4GB	Climate	Floating points
CESM-ATM [42]	33	26×1800×3600	17GB	Weather	Floating points
JHTDB [52]	10	512×512×512	5GB	Turbulence	Floating points

Comparison of lossy compressors in evaluation

In the experiments, this dissertation compares QoZ with five other error-bounded lossy compressors, which have been verified to have good compression quality and/or performance in prior works [66, 58, 101, 65]. The six compressors can be categorized into **high-performance compressors** and **high-ratio compressors**. The high-performance compressors have relatively fast compression speeds with moderate compression ratios, including SZ3.1 [58] and ZFP 0.5.5 [61]. The high-ratio compressors achieve a high compression ratio/quality with advanced data processing methods, therefore having relatively low compression speeds. They are SPERR 0.6 [49], FAZ [66], and TTHRESH [11]. For QoZ, this dissertation evaluates 2 versions of it: 1.1 (minor updates on 1.0) and 2.0. QoZ should be categorized as a high-performance compressor because it exhibits comparable compression speed with modern high-performance compressors.

Experimental configurations and evaluation metrics

In the compression experiments, the error bound mode adopted is value-range-based error bound (denoted as ϵ) [83], which is essentially equivalent to the absolute error bound (denoted as e), with the relationship of $e = \epsilon \cdot value\_range$. This mode has been broadly used in the lossy compression community [55, 58, 57, 103, 66].

This dissertation performs the evaluation based on the following key metrics:

- Speeds: Check the compression and decompression speeds of compressors.
- Compression ratio (CR) under the same error bound: Compression ratio is the metric mostly cared for by the users. Given the input data X and compressed data Z, the compression ratio CR is: $CR = \frac{|X|}{|Z|}$ ($|$ $|$ is the size operator).
- *Rate-PSNR plots*: Plot curves for compressors with the bit rate of the compressed data and the decompression data PSNR.
- *Rate-SSIM plots*: Another rate distortion evaluation plotting bit rate and SSIM [93].
- Parallel throughput performance with compressors: Simulate and perform parallel data transfer tests on the distributed scientific database on multiple supercomputers.
- Visualization with the same CR: Comparing the visual qualities of the reconstructed data from different compressors based on the same CR.

2.7.2 Experimental Results

Compression speeds

To verify the categorization of compressors and examine the compression efficiency of QoZ, Table 2.3 presents the compression and decompression speeds of comparison compressors and QoZ (under error bound 1e−3, i.e., 10^{-3}) on the Anvil machine. From the table, readers can clearly observe that the high-performance compressors (SZ 3.1, ZFP 0.5.5, and QoZ 1.1) have far better compression speeds than the high-ratio compressors (SPERR, FAZ, and TTHRESH) with the gap of 3×-10×. Moreover, having a speed of around 70% $\sim$ 90% of QoZ 1.1, QoZ 2.0 can still be regarded as a high-performance compressor, achieving 2× $\sim$ 6× performance improvement over SPERR/FAZ, and 4× $\sim$ 17× performance improvement over TTHRESH. This relatively high speed ensures the advantages of QoZ on efficiency-oriented and high-ratio-preferred compression tasks. Figure 2.14 presents the error bound-compression speed curves of each compressor on the 6 tested datasets (the x-axis is the negative log10 of the error bounded and the y-axis is the compression speed). Those plots also prove that QoZ is much more efficient than the high-ratio compressors (SPERR, FAZ, and TTHRESH) and has close performances to SZ3.

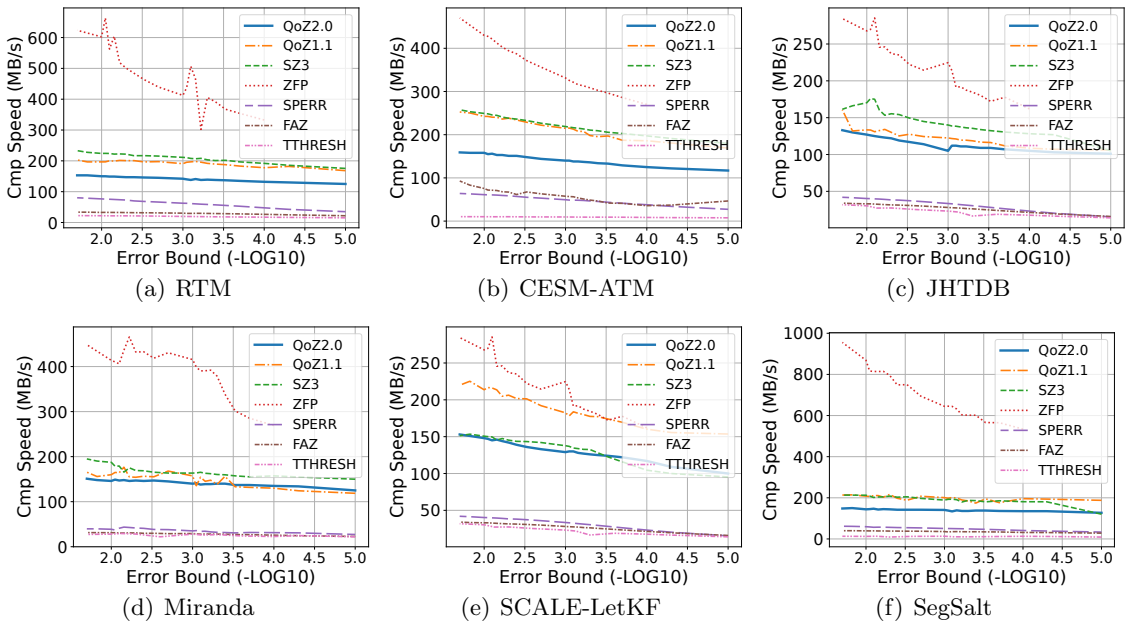
Compression ratios with the same error bounds

Compressing the datasets with the selected compressors under the same error bounds, this dissertation lists all the compression ratios in Table 2.4 and 2.5. Table 2.4 is a comparison among the high-performance compressors. QoZ 2.0 achieves the best com-

Table 2.3: Execution speeds (MB/s per CPU core) with ϵ=1e-3

Type	Dataset	SZ 3.1	ZFP 0.5.5	QoZ 1.1	SPERR 0.6	FAZ	TTHRESH	QoZ 2.0
Compression	CESM	219	331	215	49	58	10	140
	RTM	211	412	191	63	30	18	142
	Miranda	163	416	157	35	29	28	140
	SCALE	188	191	182	32	61	17	129
	JHTDB	140	225	122	33	28	23	105
	SegSalt	189	645	201	51	36	13	141
Decompression	CESM	661	584	689	92	101	53	513
	RTM	786	622	626	124	64	108	510
	Miranda	419	946	351	75	60	111	473
	SCALE	610	553	567	68	140	53	450
	JHTDB	376	425	243	70	59	60	330
	SegSalt	592	1060	629	108	65	97	485

pression ratio in all cases. On the SegSalt dataset, QoZ 2.0 has a 40% $\sim$ 75% compression ratio improvement over the second-best compressor. On the RTM, Miranda, and JHTDB datasets, QoZ 2.0 achieves 20%-45% compression ratio improvements over the second-best. On the CESM-ATM dataset, under the error bound of 1e-3, QoZ 2.0 has a compression ratio of about 2.36$\times$ as high as the second-best (SZ3.1). With these considerable improvements, this dissertation can assert that QoZ 2.0 is the best choice among high-performance compressors regarding optimizing the error-bound-fixed compression ratio. This dissertation also compares the compression ratios of QoZ 2.0 with the ones from the high-ratio compressors in Table 2.5. It shows that QoZ 2.0 can obtain even higher compression ratios than them in certain cases (e.g. on SCALE-LetKF and JHTDB). Note that the speed of QoZ 2.0 is substantially faster than the high-ratio compressors, making it quite competitive over them in speed-concerned use cases.



(a) RTM (b) CESM-ATM (c) JHTDB

(d) Miranda (e) SCALE-LetKF (f) SegSalt

Figure 2.14: Error bound-compression speed plots.

Compression rate-distortion

This section presents the evaluations of the compression rate-distortion with QoZ 2.0 and other high-performance compressors. The high-ratio compressors are capable of achieving excellent compression rate-distortion by spending much more time cost than high-performance compressors, therefore the comparison of rate-distortion would be fair if and only if the high-ratio compressors are excluded, making it within the scope of high-performance compressors to clearly examine how QoZ 2.0 has improved the compression quality meanwhile maximally preserving the compression efficiency. In Figure 2.15, the bit rate-PSNR curves of 4 high-performance compressors on 6 datasets are plotted and displayed (in which the rate-PSNR optimization targets are set for QoZ 1.1, FAZ, and QoZ 2.0). Apparently, QoZ 2.0 has dominated this evaluation term, achieving the best PSNR

Table 2.4: Compression ratios of high-performance compressors (SZ, ZFP, QoZ 1.1 and QoZ 2.0)

Dataset	ϵ	**SZ 3.1**	**ZFP 0.5.5**	**QoZ 1.1**	**QoZ 2.0**	**Improve (%)**
RTM	1E-2	1764	62.9	2156	**2701**	25.3
	1E-3	249	26.2	285	**395**	38.6
	1E-4	55.3	14.3	58	**71.1**	22.6
Miranda	1E-2	574.6	46.6	977	**1320**	35.1
	1E-3	168	25.6	181	**258**	42.5
	1E-4	47.3	14.5	47.7	**63.6**	33.3
SegSalt	1E-2	856	59.1	1005	**1484**	47.7
	1E-3	140.6	24.9	151	**260**	72.2
	1E-4	38.2	14.9	35.9	**61.7**	61.5
SCALE	1E-2	167.3	14.5	160	**186**	11.2
	1E-3	40.4	7.8	41.5	**52.9**	27.5
	1E-4	14.1	4.6	13.4	**15.4**	9.2
JHTDB	1E-2	528.2	22.3	647	**838**	29.5
	1E-3	73.2	9.8	77.8	**101**	29.8
	1E-4	15.8	5	15.9	**20.6**	29.6
CESM-ATM	1E-2	373	18.2	263	**675**	81.0
	1E-3	64.9	9.6	59.4	**153**	135.7
	1E-4	22.9	5.8	21.7	**38.9**	69.9

under all bit rates on each dataset. This implies that, among the high-performance compressors, QoZ 2.0 can always provide the best quality of decompressed data (in terms of PSNR) under the same compression ratio, or can always yield the most compact compressed data for a certain PSNR constraint. On the CESM-ATM dataset, under PSNR of 70, QoZ 2.0 reaches around 360% compression ratio improvement over the second-best QoZ 1.1. On the SegSalt dataset under PSNR of 80, QoZ 2.0 achieves about 100% compression ratio improvement over the second-best QoZ 1.1. There are approximately $20\% \sim 80\%$ same-PSNR compression ratio improvements achieved by QoZ 2.0 on the other 4 datasets.

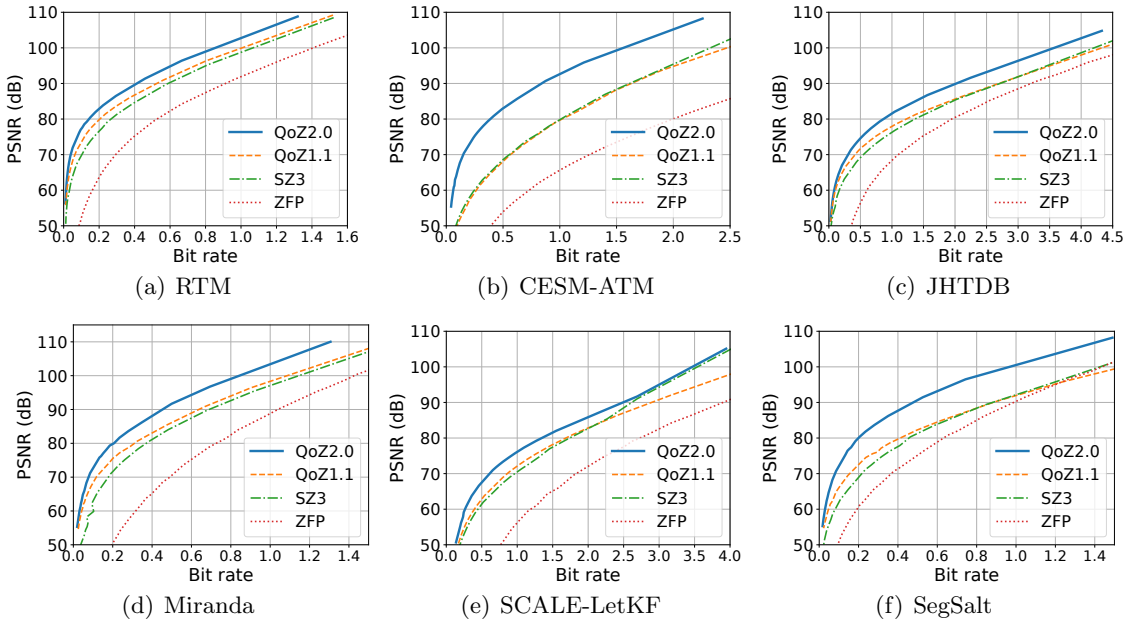
To evaluate the QoZ compression quality with more quality metrics, this dissertation also checked the SSIM of the decompressed results of each high-performance compressor, and those results are illustrated in Figure 2.16. Like the cases for the PSNR, QoZ

Table 2.5: Compression ratios of QoZ 2.0 and high-ratio compressors (SPERR, FAZ, and TTHRESH)

Dataset	ϵ	**SPERR 0.6**	**FAZ**	**TTHRESH**	**QoZ 2.0**
RTM	1E-2	2187	2695	782	**2701**
	1E-3	440	**642**	71.4	395
	1E-4	84.1	**119**	23.7	71.1
Miranda	1E-2	971.4	996.5	447	**1320**
	1E-3	243.9	**263.5**	142	258
	1E-4	74.5	**93.6**	55.1	63.6
SegSalt	1E-2	1219.4	**1639.6**	291	1484
	1E-3	228.9	**388.9**	99.5	260
	1E-4	61.3	**117.3**	28.8	61.7
SCALE	1E-2	103.5	177.9	80.0	**186**
	1E-3	35.5	51.8	18.9	**52.9**
	1E-4	15	**16.8**	8.4	15.4
JHTDB	1E-2	639.8	726	373	**838**
	1E-3	89.3	90.7	65.1	**101**
	1E-4	19.9	20.2	17.1	**20.6**
CESM-ATM	1E-2	**1221**	292	83.5	675
	1E-3	150	77.4	20.4	**153**
	1E-4	35	26.3	8.7	**38.9**

2.0 undoubtedly presented the best SSIM under the same compressed size over all other evaluated high-performance compressors. Under the same compression bit rate, on multiple datasets including RTM, JHTDB, SCALE-LetKF, and SegSalt, there are 20% $\sim$ 40% SSIM improvements from QoZ 2.0 over the second-best QoZ 1.1. The SSIM improvements can be even much larger in the case of the CESM-ATM dataset.

In the analysis, the outstanding compression quality of QoZ 2.0 as a high-performance compressor is attributed to 3 aspects: First, the advanced interpolation techniques described in Section 2.5 have significantly raised the interpolation-based prediction accuracy for smooth datasets such as RTM, Miranda, SegSalt, and JHTDB. Next, avoiding interpolations along non-smooth directions, the compression for datasets with non-smooth di-



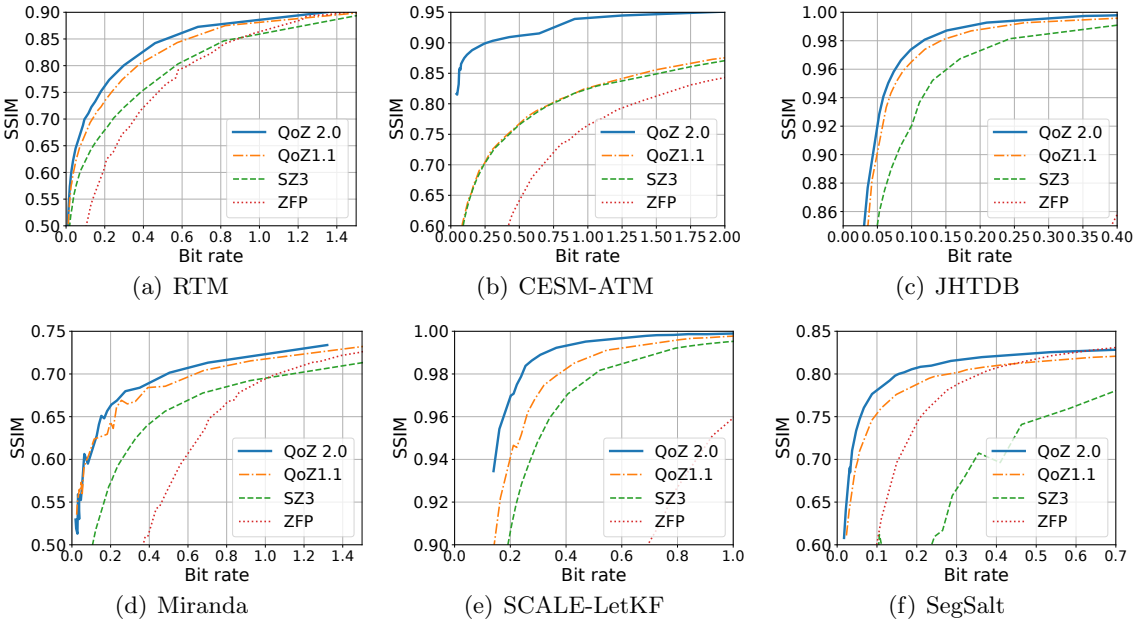
(a) RTM (b) CESM-ATM (c) JHTDB

(d) Miranda (e) SCALE-LetKF (f) SegSalt

Figure 2.15: Rate-distortion (PSNR) plots of high-performance compressors.

mensions (e.g. SCALE-LetKF and CESM-ATM) have been obviously enhanced by the dynamic dimension freezing technique (Section 2.6.4). Third, the block-wise interpolation tuning (Section 2.6.4) fine-tunes the interpolation on each data block, further optimizing the overall compression.

Lastly, this dissertation would like to claim that, in several cases, the compression quality (i.e. rate-distortion) of QoZ can be at least comparable with the high-ratio compressors. In the comparisons between QoZ 2.0 and high-ratio compressors displayed in Figure 2.17, although on the Miranda dataset (Figure 2.17 (b)) QoZ 2.0 has quality gaps to the SPERR and FAZ, Figure 2.17 (a), (c) and (d) indicate that QoZ 2.0 may achieve close or even similar rate-distortion to the high-ratio compressors, with a compression speed far higher than them.



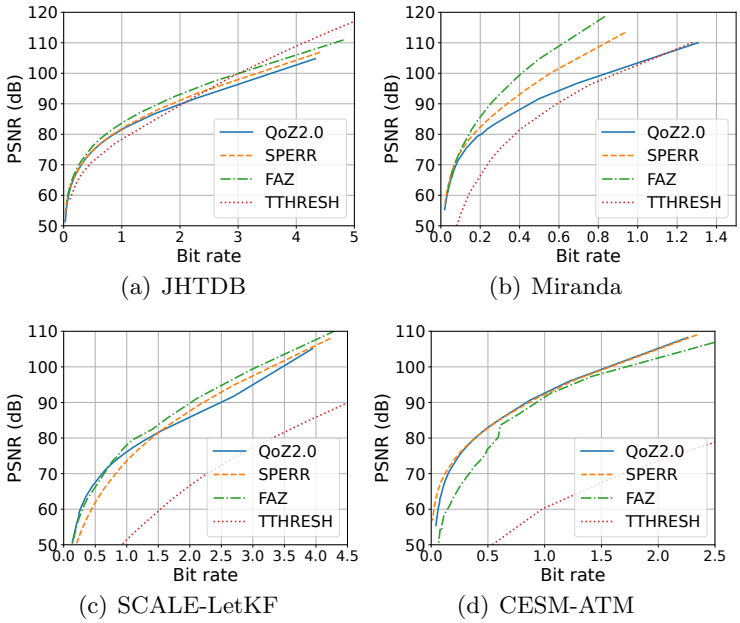
(a) RTM (b) CESM-ATM (c) JHTDB

(d) Miranda (e) SCALE-LetKF (f) SegSalt

Figure 2.16: SSIM of high-performance compressors.

Parallel data transfer test on the distributed database

So far, it has been analyzed how QoZ outperforms other high-performance compressors in terms of compression quality. Furthermore, this section will examine whether it can over-perform existing state-of-the-art error-bounded lossy compressors including high-ratio compressors in real-world use cases in which the compression and decompression time need to be taken into account. To this end, this dissertation has designed a real-world scale parallel data transfer experiment on the distributed scientific database. In this experiment, a distributed scientific database is established on multiple machines, and to accomplish the target of fast data transfer and access between the super-computers, instead of costing unacceptable time transferring the original exascale data, a lossy compressor compresses and decompresses the data in parallel on the source and destination machine, and only



(a) JHTDB (b) Miranda

(c) SCALE-LetKF (d) CESM-ATM

Figure 2.17: Rate-PSNR of QoZ 2.0 and high-ratio compressors (QoZ 2.0's speed is 2x-17x of high-ratio compressors).

the compressed data with a highly-reduced size are transferred between the machines. The total time cost of this task is the accumulation of the local data I/O time, compression time, decompression time, and transfer time of the compressed data.

To convincingly prove the effectiveness of QoZ 2.0 for the parallel data transfer task, this dissertation conducted the corresponding experiments under a certain configuration. For a parallel test with p cores, the experiments augment the datasets by p times then let each core compress and decompress the data in the original size. Using 2048 cores, the experiments leveraged the 7 compressors to compress and transfer the datasets bidirectionally between the Anvil and Bebop supercomputer, constraining the decompressed data following the same distortion (PSNR = 80). The inter-machine data transfer is supported by the Globus Transfer Service [15, 6, 14], which is an efficient and widely adopted data

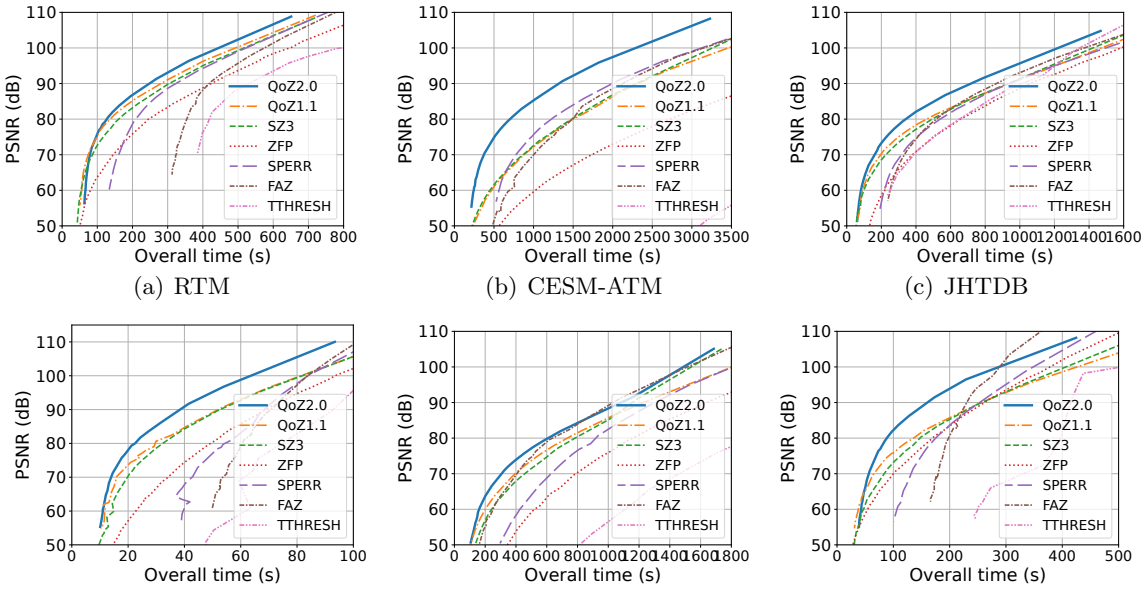
Table 2.6: Compression-based parallel data transfer throughput time (in seconds, 2048 cores, under PSNR=80). Inter-machine speed is the transfer speed of compressed data between 2 machines.

Dataset	Direction	Inter-machine Speed (GB/s)	SZ3	ZFP	QoZ 1.1	SPERR 0.6	FAZ	TTHRESH	QoZ 2.0	Improve (%)
CESM-ATM (41TB)	Anvil to Bebop	0.79 ∼0.91	1934	3221	1812	1560	1586	7752	**1005**	35.6
	Bebop to Anvil	0.95 ∼1.19	1614	2695	1553	1522	1544	8560	**916**	39.8
RTM (14TB)	Anvil to Bebop	0.58 ∼1.19	198	362	**173**	277	494	527	181	-4.8
	Bebop to Anvil	0.47 ∼1.04	189	524	**166**	296	474	560	182	-9.5
Miranda (2TB)	Anvil to Bebop	0.46 ∼1.04	49	84	44	72	87	121	**39**	11.3
	Bebop to Anvil	0.54 ∼0.82	46	117	49	71	86	120	**43**	6.5
SCALE-LetKF (13TB)	Anvil to Bebop	0.88 ∼0.94	873	1354	820	1037	782	2354	**728**	7.0
	Bebop to Anvil	1.05 ∼1.15	745	1181	707	1007	670	2002	**624**	6.8
JHTDB (10TB)	Anvil to Bebop	0.83 ∼1.15	567	826	527	645	583	835	**417**	20.9
	Bebop to Anvil	0.97 ∼1.18	486	707	473	648	574	883	**366**	22.7
SegSalt (8TB)	Anvil to Bebop	0.63 ∼1.18	163	289	174	221	251	393	**137**	15.9
	Bebop to Anvil	0.76 ∼1.06	167	241	153	213	265	300	**132**	14.0

transfer service in scientific research and education fields. Table 2.6 presents data transfer speed and the time cost with each compressor for each dataset. On all of the datasets tested, either QoZ 1.1 or QoZ 2.0 achieves the optimal data transfer time cost, and on most of the datasets, QoZ 2.0 improves the optimal overall transfer time by 5% $\sim$ 40% over QoZ 1.1, and even more when compared to other compressors. Therefore, the optimized balance of compression quality and efficiency of QoZ does contribute to its utility in real-world large-scale parallel data transfer tasks.

Due to the computing resource limitation for executing the multi-core large-scale data transfer tests and repeating them with different datasets, compressors, and configurations, this dissertation has also designed a model for approximating the actual time costs in those tasks. For a specific core number p and a data transfer speed s, the sequential compression/decompression speed of the compression/decompression with the same per-core data is used to estimate the parallel compression/decompression time cost, and the approximated data transfer time is just the compressed data size divides the transfer speed s. With those methods, for each dataset, the time costs are approximated under a variety

of compression error bounds, then the time cost-PSNR curves are plotted and presented in Figure 2.18. The compressor speeds are acquired on the Anvil machine introduced in Section 2.7.1, the core numbers are 2048, and the data transfer speed is set to 1GB/s (according to the experimental results in Table 2.6). From the plots, this dissertation can claim that, for this task, QoZ 2.0 has the potential to keep an advantage over the other existing compressors. On Miranda, CESM-ATM, and JHTDB datasets, with the approximations, QoZ 2.0 exhibits the minimized time cost for each fixed PSNR. On RTM, SCALE-LetKF, and SegSalt datasets, QoZ 2.0 may still always be the top-performing compressor and can have the optimized time cost in wide ranges of PSNR.



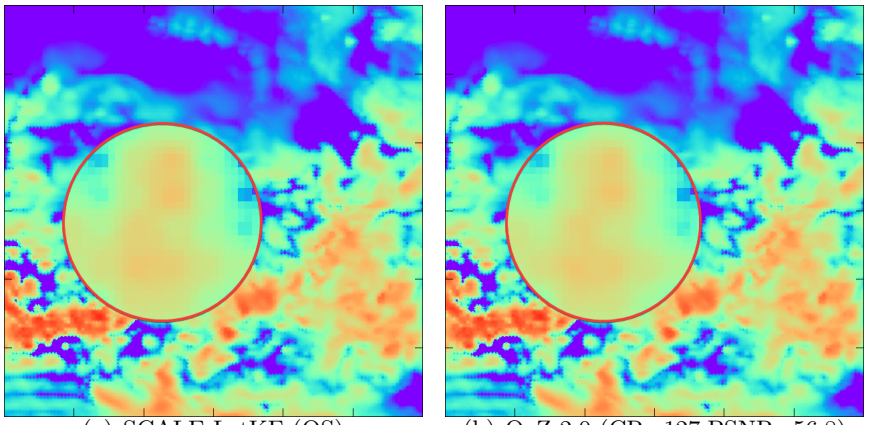
(a) RTM (b) CESM-ATM (c) JHTDB

(d) Miranda (e) SCALE-LetKF (f) SegSalt

Figure 2.18: Parallel data transfer time approximation and decompression PSNR (simulation on the Anvil supercomputer, $p = 2048$, $s = 1GB/s$).

Case study: decompression visualizations

As an example of the effectiveness of QoZ compression, this section proposes a case study of the compression tasks, visualizing the decompression outputs from various high-performance compressors. The example data input is the QS field (getting logarithmized in preprocessing) from the SCALE-LetKF dataset, and it is compressed by QoZ 2.0, QoZ 1.1, and ZFP (SZ3 is omitted in this test because QoZ 1.1 and SZ3 have close speeds and QoZ 1.1 has better compression quality than SZ3) under similar compression ratios. The visualizations of 2-D slices from the original data and decompressed data are presented in Figure 2.19. In this case, among the decompression results with very close compression ratios, the decompression result of QoZ 2.0 (Figure 2.19 (b)) achieves the lowest data distortion with the highest PSNR (56.8). Moreover, regarding the magnified regions in Figure 2.19, compared to the decompression results of QoZ 1.1 (Figure 2.19 (c), PSNR=52.7), QoZ 2.0 has better preserved the local data patterns in the original input (Figure 2.19 (a)). This case is an example to show the strong capability of QoZ 2.0 in providing high-quality compression results with high compression speed.



(a) SCALE-LetKF (QS) (b) QoZ 2.0 (CR=127,PSNR=56.8)

(c) QoZ (CR=126,PSNR=48.4) (d) ZFP (CR=118,PSNR=21.0)

Figure 2.19: Visualization of SCALE-QS field (logarithmized) and the decompressed data.

Chapter 3

FAZ: Pipeline-Auto-Tuning-Based High-Ratio Scientific Error-Bounded Lossy Compression

3.1 Overview

3.1.1 Motivations and challenges: Pursuing extremely high-ratio scientific error-bounded lossy compression

In the last chapter, QoZ has shown how to ensure high-quality scientific error-bounded lossy compression with adequate compression efficiency. In this chapter, this dissertation will switch to another aspect, demonstrating how it can pursue an extremely high-ratio data compression with a hybrid of various data compression strategies when efficiency is not a major concern anymore. This topic and its outcomes will be critical for the

storage cost reduction tasks of scientific databases. Today's high-performance computing (HPC) applications are generating huge amounts of scientific data on exascale systems. For example, the Gyrokinetic Toroidal Code (GTC) [60] which simulates the movement of magnetic particles in a confined fusion plasma produces petabytes of data in a few hours [91]. Such unprecedented data volume and velocity lead to severe problems in data storage and transmission, dictating an urgent need for effective high-ratio data compression.

State-of-the-art error-bounded lossy compressors, such as SZ3 [58, 101], ZFP [61] and MGARD+ [57] have been effective in reducing storage requirement and alleviating I/O pressure in many applications [56]. However, no single lossy compressor can provide dominant compression quality for all the datasets due to their diverse characteristics. This fact makes it a challenge for scientists to adopt error-bounded lossy compressors in their practical use cases as numerous experiments are needed to identify the best-fit candidate, leading to unexpected delays in scientific discoveries. Therefore, this dissertation plans to propose a new work of scientific error-bounded lossy compression that merges heterogeneous data compression techniques and can be highly self-adaptive to diverse compression tasks in order to achieve an optimized compression ratio across a large variety of scientific datasets.

In particular, designing this highly flexible and inclusive compression framework faces several critical challenges. First, it is non-trivial to organize the diverse compression techniques in a single compression framework, as different techniques usually have distinct data processing pipelines. For example, prediction-based compressors such as SZ3 [58] process data in a point-by-point fashion, while transform-based compressors such as ZFP [61] require block-wise processing for efficient decorrelation. Second, it is challenging to design a

universal auto-selection and auto-tuning algorithm orchestrating all the heterogeneous data processing modules. Last, it is always important to balance the effectiveness and efficiency of auto-tuning to make it adaptive toward diverse user requirements.

3.1.2 Contributions of FAZ

Addressing the aforementioned research targets and challenges. this chapter proposes FAZ, a flexible, modular, and auto-tuned compression framework that projects the best adaptability of error-bounded compression to free scientists from trials and errors when selecting lossy compressors. FAZ can achieve significantly better compression quality on various application datasets over other existing works. This is especially attributed to the adaptive design, in which FAZ leverages a fully adaptive compression pipeline whose modules are mostly dynamically selected and auto-tuned during the compression. This substantially differs from existing works which either depend on fixed compression pipelines [54, 84] or suffer from strict auto-tuning and limited adaptability [103, 65]. The contributions of FAZ are summarized as follows:

- This dissertation carries out systematic and thorough compression characterization and analysis on most of the representative scientific datasets to identify the pros and cons of state-of-the-art lossy compression techniques, which is fundamental to the subsequent design and optimization.

- This dissertation proposes a flexible modular error-bounded compression framework FAZ, which integrates heterogeneous compression techniques and features an advanced module for automatically tuning the best compression pipeline.

- In experimental evaluations, FAZ outperforms all existing error-bounded lossy compressors under the tested compression cases in various quality metrics. Compared with the second-best lossy compressors, FAZ achieves compression ratios of up to 120% improvement under the same error bound, up to 190% improvement under the same PSNR, and up to 75% improvement under the same SSIM.

3.2 Related Work

This section discusses the related works in two facets: existing error-bounded lossy compressors and existing techniques for auto-selection/auto-tuning in lossy compression.

Multiple error-bounded lossy compressors have been developed for scientific data reduction toward different use cases [13]. Existing lossy compressors mainly fall into four categories – prediction-based, transform-based, dimension-reduction-based, and neural-network-based. Prediction-based compressors leverage data approximation algorithms (e.g., linear regression [55] and dynamic spline interpolations [101]) to predict each data point and then apply quantization or similar methods to control the data distortion based on user-specified error bounds. Typical examples include FPZIP [62], SZ2 [55] and SZ3 [58, 101]. Transform-based compressors make use of transforms to decorrelate the data so that the transformed coefficients are much easier to compress. One typical example is ZFP [61], which adopts exponent alignment with orthogonal transform and embedded encoding. Several other compressors including SSEM [76], VAPOR [20], and SPERR [49], adopts wavelet transforms. Dimension-reduction-based compressors such as TTHRESH [11] depend on the dimension reduction techniques such as (high-order) singular vector decomposition (SVD).

Neural-network-based compressors [64, 29, 68, 33] adopt neural network models such as autoencoder families [12, 45, 46].

Since many existing studies have shown that error-bounded lossy compressors generally exhibit distinct effects on different datasets, there exist a few studies to combine different compressors. Lu et al. [70] proposed a method to make a better choice between SZ and ZFP by estimating their compression ratios based on the same error bound. Tao et al. [84] proposed a more advanced algorithm that can select the better compressor (either SZ or ZFP) based on the peak signal-to-noise ratio (PSNR), which is a more commonly used data distortion metric in the community. Liang et al. [54] proposed a method to integrate the ZFP compressor into the prediction stage of the SZ model and then select the better predictor between them. These methods, however, are dependent on the specific off-the-shelf compressors each with fixed compression pipelines and parameter settings, therefore still have strict limitations to achieve higher compression quality.

Another well-researched topic is the auto-tuning of specific compressors. Zhao et al. [103] improved the compression quality for SZ by developing an effective method to auto-search the best parameter settings for data predictors. The QoZ [65, 67] proposed in Chapter 2 also projects a special capability of dynamically parameterizing the interpolation-based data predictors and error settings according to flexible compression optimization targets. However, they are all limited to just auto-tuning the parameters of data predictors. In contrast, FAZ establishes a fully dynamic compression framework by including different compression archetypes, which can obtain significantly improved compression quality for most scientific datasets.

3.3 Problem Formulation

This section formulates the research problem for high-ratio pipeline-auto-tuning-based scientific error-bounded lossy compression. Unlike the traditional lossy compression optimization strategies that rely on fixed compression pipelines or just parameter auto-tuning, FAZ projects a flexible compression pipeline (denoted as P in the following text) that can be auto-tuned across different compression archetypes at runtime. Every compression pipeline P is composed of several functional modules (denoted as $\{M_i\}$) each with a set of parameters (denoted as $\{\Theta_i\}$) to optimize. Those modules include (but are not limited to) data transform, data prediction, bit-stream encoding, and lossless post-processing, each playing an important role in the whole framework. For the compression, this section denotes the input data array as X and the compressed data as Z. For decompression, the reconstructed data is denoted as $X^{'}$.

Specifically, the research objective is to instantiate an optimized compression pipeline P by (a) selecting outstanding data processing modules, (b) auto-tuning their parameters based on the input dataset and user requirements, involving both compression ratio and quality metrics, as shown below.

$$
\begin{aligned}
\{M_i\}, \{\Theta_i\} = & \underset{\{M_i\} \subset \mathcal{M},\ \Theta}{\arg \mathrm{OPT}}\ T(X, X^{'}, Z) \\
s.t. \quad & P_c, P_d = \mathcal{P}(\{M_i\}, \{\Theta_i\}) \\
& Z = P_c(X), X^{'} = P_d(Z) \\
& |x_i - x_i^{'}| \leq e, \forall x_i \in X, x_i^{'} \in X^{'}
\end{aligned}
\tag{3.1}
$$

In Formula (3.1), T is an optimization target, which can be maximizing compression ratio or optimizing the rate-distortion. In the rate-distortion, rate means bit-rate (the average

number of bits per data value required in the compression) and distortion means a certain distortion on data such as PSNR [83] and Structural Similarity Index Measure (SSIM) [93]. OPT refers to the optimization operation (e.g., max, min) according to the target. $\mathcal{M}$ is the complete module set, $\mathcal{P}$ is the pipeline constructing method, P_c and P_d are the compressor part and decompressor part of pipeline P, and e is the user-specified or deduced absolute error bound.

3.4 Analysis of Existing Compressors

This section presents an in-depth analysis of existing state-of-the-art error-bounded lossy compressors with several important takeaways, which form guiding principles for the FAZ design. Specifically speaking, it answers the following key questions: (1) Which lossy compressors exhibit the best compression ratio and quality on different datasets, and which modules used by them take critical roles in obtaining the high compression rate-distortion? (2) What are the limitations of those compressors and modules?

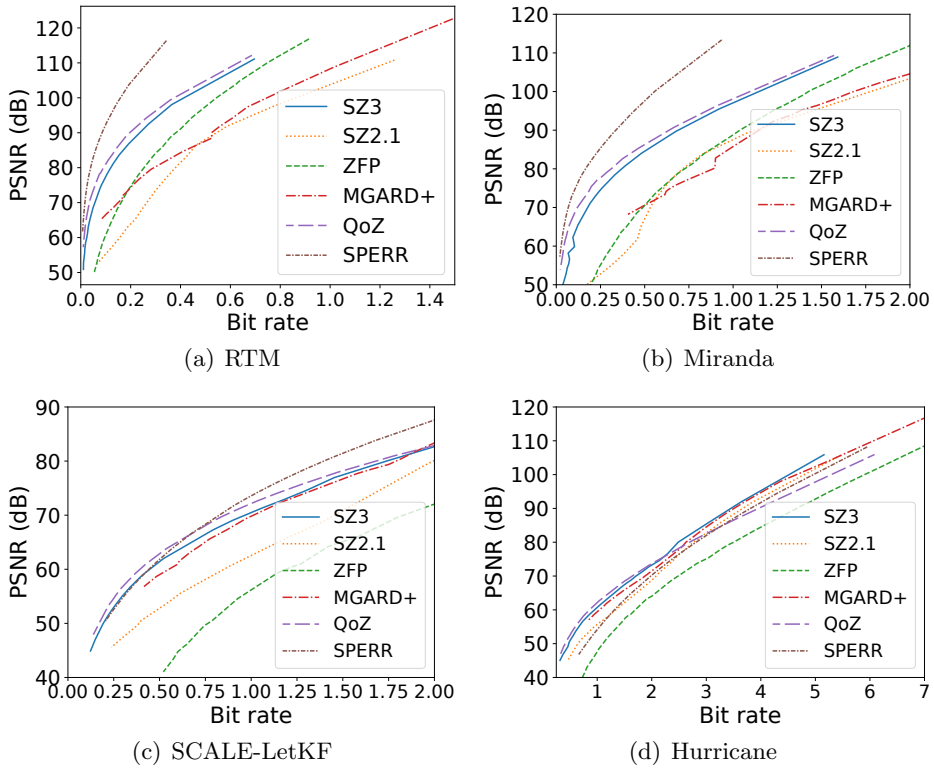
3.4.1 Introduction and investigation of high-ratio error-bounded lossy compression strategies

To analyze and better understand the existing lossy compressors, this dissertation has profiled the rate-distortion (PSNR) of several state-of-the-art error-bounded lossy compressors including SZ3 [58, 101], SZ2.1 [55], ZFP0.5.5 [61], MGARD+ [57], QoZ [65] and SPERR [49] on multiple scientific datasets each with diverse characteristics (details are described in Section 3.7.1). In the results, it is observed that either QoZ or SPERR provides

the best rate-distortion in all datasets, and due to space limitations, this section shows the results in Figure 3.1 based on 4 of the datasets tested with. In the following, this section briefly discusses their designs, as well as key insights into why they can work effectively on lossy compression.

Interpolations

The interpolation-based data predictor used in SZ3 [101] has shown great effectiveness in lossy compression because of the following reasons: First, the interpolation-based data predictor has the ability to recover non-linear data patterns; Second, compared with other data predictors such as Lorenzo [82], the formula of interpolation introduces lower errors when data points are predicted by other predicted and error-bounded values, especially with large error bounds. Second, no coefficients need to be saved for decompression, in contrast to some other data prediction methods such as linear-regression-based predictor [55] that has to store certain amounts of coefficients for the data reconstruction. The predictor of QoZ inherits these advantages from the SZ3's interpolators and also projects new important features, which in many cases improve the compression quality significantly over SZ3 in turn due to two reasons. On the one hand, QoZ includes the anchor point design (losslessly saving a sparse grid of data points), largely improving the data prediction accuracy for the rest of the data points with negligible compression ratio loss. On the other hand, QoZ parameterizes the interpolators to separately auto-tune the interpolator types (linear or cubic), interpolation dimensional orders, and interpolation error bounds on each interpolation level, therefore optimizing the prediction accuracy on those data points.



(a) RTM (b) Miranda

(c) SCALE-LetKF (d) Hurricane

Figure 3.1: Rate-distortion (PSNR) of compressors. In (a) and (b), SPERR is the best. In (c) and (d), QoZ is the best.

Transforms

Another outstanding technique often used in lossy compressors is data transform. A typical example is ZFP [61] which performs a near orthogonal transform at the 4^d block size for d-dimensional data. In comparison, on some datasets, a wavelet transform-based compressor SPERR [49] exhibits much higher compression ratios than other state-of-the-art compressors such as ZFP and SZ with the same data distortion. The key reason is that as a transform-based data compressor, SPERR leverages a global multi-level discrete wavelet transform (DWT) [35], which can decorrelate the input dataset in a global view for a higher compression ratio. SPERR adopts a fixed wavelet transform – CDF9/7 [21], which is popular in many other lossy compressors such as JPEG2000 image compression [85].

3.4.2 Limitations of the existing strategies

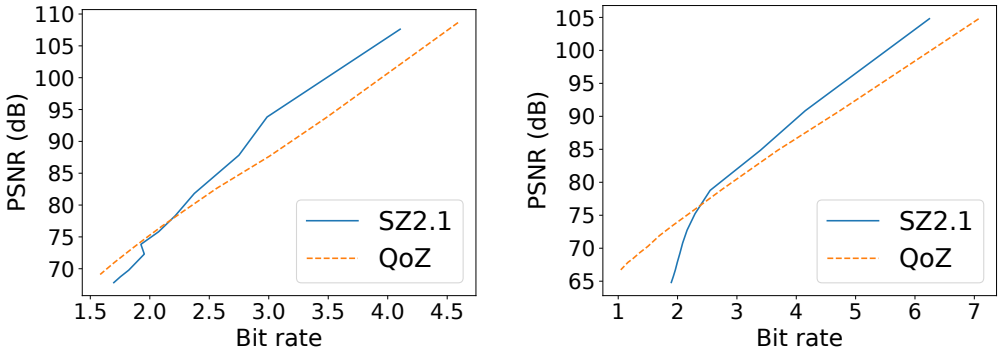
In this subsection, the limitations of the above two strategies are discussed, based on which this dissertation raises new design concepts for the FAZ compression framework.

Interpolation is not the universal solution

As Figure 3.1 (a) and (b) indicate, on some datasets, in terms of rate-distortion the interpolation-based QoZ is not as good as the wavelet-transform-based SPERR which does not integrate any data prediction modules. In the view of this dissertation, the reason is two-fold: On one hand, the data variations in certain datasets are compatible with the wavelet transforms very well, so the transformed coefficient arrays often exhibit a sharp distribution with a high decorrelation nature. On the other hand, the SPECK (Set Partitioning Embedded bloCK) encoder [73] adopted by SPERR is a classic encoding algorithm particularly optimized for wavelet transformed coefficients.

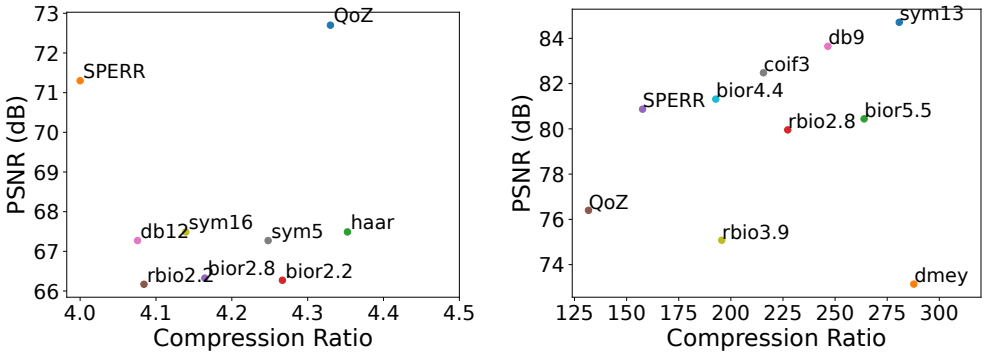
Moreover, even for only prediction-based compressors, the interpolation-based predictor has performed worse than some traditional predictors on certain input data. As analyzed in [101, 58], the interpolation-based predictor used by SZ3 does not perform as well as the Lorenzo data predictor in some high accuracy (low error bound) compression cases. This observation also goes with the improved QoZ interpolation-based predictor in the experiments. Figure 3.2 shows some examples in terms of the rate-distortion (PSNR) curves between QoZ versus SZ2.1. The input data are two snapshots from the Scale-LETKF application and Hurricane application, and relatively small value range-based relative error bounds [83] (lower than 1e-3) are used. In those cases, for the same PSNRs, SZ2.1 (us-

ing the Lorenzo predictor in those cases) exhibits higher compression ratios than the QoZ compressor that adopts interpolation-based predictors. This finding motivates us to merge more types of data predictors other than interpolation into the FAZ.



(a) Scale-LetKF-QC (LOG10) (b) Hurricane-U

Figure 3.2: Interpolations (QoZ) vs Lorenzo (SZ2.1).



(a) NYX-DarkMatterDensity (b) SEGSalt-2000

Figure 3.3: Comparing QoZ (interpolations), SPERR (CDF9/7), and other wavelets (higher PSNR is better).

A dynamic usage of wavelets is needed

There is an important finding about the wavelet transform in the error-bounded lossy compression that the CDF9/7 wavelet used in SPERR is not always effective in de-

correlating the dataset. Figure 3.3 shows several scattered points representing different compression qualities (bit rate and PSNR) when replacing the CDF9/7 wavelet with other wavelets in SPERR for the same compression task with fixed input and error bound, as compared with QoZ and SPERR. Specifically, as the example shown in Figure 3.3 (a), when compressing data of the Dark Matter Density field in the NYX dataset, SPERR suffers from worse rate-distortion results than QoZ which does not use any wavelet modules. According to Figure 3.3 (b), on the #2000 snapshot of the SEGSalt dataset, readers can observe that there are quite a few other wavelets that can provide better compression quality. Those observations motivate us to select the best-fit wavelet and auto-tune related parameters, which may significantly improve the compression quality.

3.5 Overview of FAZ Design

This section presents an overview of the FAZ framework. The framework is based on the SZ3 modular compression framework and perfected with the analysis and findings described in Section 3.4.

3.5.1 Framework

Figure 3.4 presents the flexible modular design of the FAZ compression framework, in which the modules are dynamically selected and combined for different inputs. Such an adaptive design is a development from the SZ3 modular framework [58, 65] and contrasts with the traditional fixed-compression-pipeline design [61, 49], which can lead to significantly improved compression quality. As shown in Figure 3.4, there are 6 stages in

the FAZ compression pipeline: pipeline auto-tuning, data transform, data prediction, error controlling, bit-stream encoding, and lossless post-processing. Moreover, besides importing multiple existing (the yellow ones in Figure 3.4) modules, compared to other existing modular compression frameworks FAZ has involved several new modules (the blue ones in Figure 3.4) for building the compression pipeline. Notice that there could be many ways for combinations of the modules because some stages could be skipped in the pipeline. For instance, the bottom two arrows skip the data transform stage and data prediction stage, respectively, as shown in the figure. In addition to the above modules implemented in FAZ, more new modules can also be included to adapt to new datasets or use cases, based on the composable modular design. What compression modules (techniques) are adaptive to the FAZ framework will be discussed in Section 3.6.

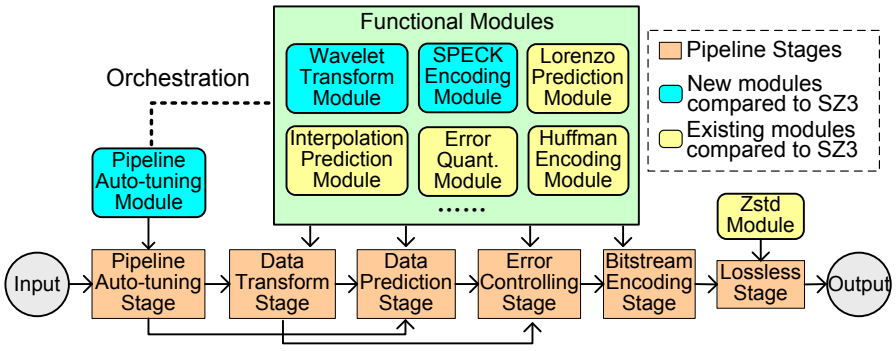


Figure 3.4: FAZ compressor framework.

3.5.2 Modules in FAZ

This subsection describes all the functional compression modules used in FAZ, including both newly involved modules compared to other existing modular compression frameworks such as SZ3 [58] and other existing state-of-the-art modules.

Newly involved modules

Pipeline auto-tuning module: This module is designed for dynamically selecting and tuning other functional modules to build the specific pipeline for each input data, which is the key contribution of FAZ and will be detailed in Section 3.6. The auto-tuned pipeline by this module could be diverse such as an interpolation-predictor-based compression pipeline, a Lorenzo-predictor-based compression pipeline, a compression pipeline integrating the optimized wavelet transform and the SPECK encoder, a hybrid compression pipeline with wavelet transform and interpolation prediction, and so on.

Wavelet transform module: FAZ integrates a dynamic wavelet module. Both forward wavelet transform (for data decorrelation) and inverse wavelet transform (for error bounding) will be leveraged. The module inherits two types of wavelets. The first one is the CDF9/7 wavelet which is used in some existing compressors such as SPERR [49]. The other one is Sym13, which is a member of the symmetric wavelet family [23] and projects the best compression quality in many cases based on a thorough characterization with 200+ different types of wavelets. Which of the two wavelets should be used during the compression is dynamically determined by the auto-tuning module. To preserve the efficiency of auto-tuning, FAZ just includes these two wavelets as they can cover the optimized choices for most of the cases in the comprehensive characterization.

SPECK encoding module: Based on the experiments and analysis, directly using the SPECK encoding for wavelet coefficients generated in the compression pipeline can generally achieve better rate-distortion than compressing the coefficients with data predictors. Therefore, this module is also included in FAZ for optimizing the rate-distortion.

Other existing modules

Here, this subsection briefly introduces the modules that have been well-adopted in existing works and also leveraged by FAZ.

Lorenzo prediction module: The experiment in section 3.4.2 shows the fact that the multi-dimensional multi-layer Lorenzo predictor [82, 103] may outperform the interpolation predictor in some cases, thus it is also included in FAZ for reaching the best adaptability.

Interpolation-based data prediction module: FAZ adopts the anchor-point-based Interpolation predictor, which is similar to the one used in QoZ [65]. Its hyper-parameter tuning is also further optimized (to be detailed in Section 3.6).

Linear-scale error quantization module: Since data prediction and SPECK encoding modules would bring unbounded errors in data reconstruction, FAZ leverages the linear-scale data quantization module for bounding the compression errors. It is a typical technique that has been used in many existing compressors such as SZ2 [55] and SZ3 [58].

Huffman encoding module: Used in FAZ for encoding and shrinking the size of the error quantization bins from the error quantization module.

Zstd lossless module: FAZ applies Zstd [22] at the end of its pipeline to losslessly compress the bit-stream for improving the compression ratio without distortion loss.

3.6 Pipeline Auto-Tuning in FAZ

To substantially improve the compression ratio and quality on the diverse input data characteristics and user requirements brings great challenges to the selection of com-

pression modules and parameters in FAZ. Since no static rules can cover all the cases, an online pipeline auto-tuning module is designed and leveraged for the composition of compression pipelines. Because of the masses of possible combinations of different modules and parameter settings, it is very challenging to design an efficient auto-tuning method that can always construct the best-fit compression pipelines for different datasets/use cases. To address this challenge, FAZ proposes several optimization strategies.

3.6.1 Overall process of pipeline auto-tuning

The overall process of the pipeline auto-tuning module in FAZ is displayed in Figure 3.5, which consists of four components: online statistical analysis for wavelet type selection, variance-based block-wise data sampling, interpolation tuning, and rate-distortion tuning. Briefly speaking, this module first runs a fast statistical analysis on the whole input dataset to determine which wavelet should be used in the auto-tuning. Then, from the input dataset, it samples a small portion of data, based on which the interpolation tuning and rate-distortion tuning are applied to select and determine the parameters of the modules in the pipeline. The interpolation tuning process to determine the level-wise interpolation type and interpolation orders in FAZ is the same as the method used in QoZ [65]. We describe the details as follows.

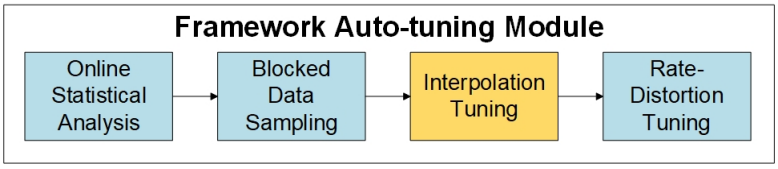


Figure 3.5: FAZ pipeline auto-tuning module. Yellow blocks indicate existing techniques and blue blocks are proposed ones.

3.6.2 Online statistical analysis for wavelets

The wavelet transform may significantly improve the rate-distortion for a variety of datasets. However, its time cost is relatively high. To balance the performance and accuracy of auto-tuning, FAZ endeavors to minimize the computational cost of wavelet transform in the auto-tuning process. Therefore, FAZ applies an efficient statistical analysis to predict which wavelet (CDF9/7 or Sym13) would be better for the given input data and then select it to be used in the rate-distortion tuning, disregarding the other one dynamically.

Our designed analysis method (selection rule) is expressed as follows. FAZ selects CDF 9/7 for rate-distortion tuning if and only if either of the following conditions is met: (1) the smallest size among all dimensions in the input data array is lower than a threshold (denoted by l); (2) The normalized variance of the input data array is larger than a threshold (denoted σ). Otherwise, Sym13 will be selected.

This rule for wavelet selection is drawn from the experiments with masses of real-world scientific datasets, and this dissertation suggests setting l and σ to 128 and 0.003 in practice respectively. As shown in Figure 3.6, the datasets exhibiting better compression quality with Sym13 are marked as red squares, and the ones exhibiting better compression quality with CDF9/7 are marked as blue circles. In the plot, the datasets with each dimension size larger than 128 and normalized variances smaller than 0.003 always exhibit higher compression quality with Sym13 wavelet (red points) than with CDF9/7 and vice versa.

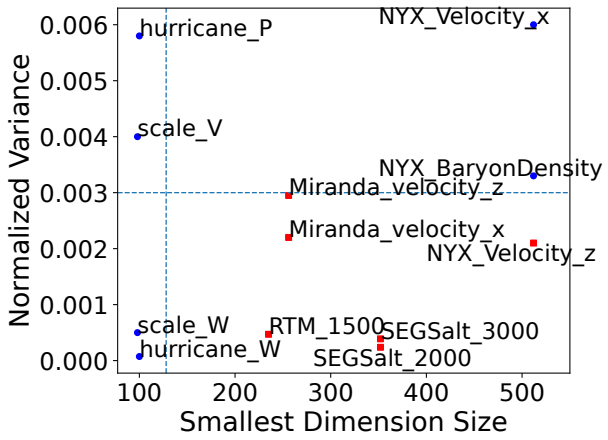


Figure 3.6: Sym13 vs CDF9/7. Blue circles (red squares) are better compressed with CDF9/7 (Sym13).

3.6.3 Variance-based block-wise sampling

FAZ leverages an advanced variance-based data sampling method to reduce the computational cost of auto-tuning. Specifically, it first splits the input data into small fixed-size data blocks, then samples a certain number (based on a hyper-parameter controlling the ratio of sampled data) of the blocks from them with the key feature that it samples data blocks with the largest variances among all of them.

We discuss the design motivation of this design as follows. In fact, some input data arrays have uneven data distributions such as having many zeros in a region. The proposed data sampling method can avoid sampling too many zero data points therefore can obtain higher accuracy in estimating the rate-distortion results compared with the data sampling methods used in SZ3 [101, 58] and QoZ [65].

3.6.4 Rate-distortion tuning

The most critical part of the FAZ pipeline auto-tuning process is the rate-distortion tuning, which is closely relevant to the adaptability of FAZ as each compression module (technique) that can be evaluated within the rate-distortion tuning process has the potential of being leveraged in the FAZ framework. Different from the existing auto-tuning methods such as the one in QoZ [65] which only tunes the interpolation predictors and their parameters or SZ2.1 [55] which just selects data predictor types, the rate-distortion tuning in FAZ is a more generalized quality metric driven auto-tuning method for the overall compression pipeline covering multiple compression stages. Specifically, the rate-distortion tuning in FAZ auto-tunes the data transform type (whether or which wavelet), data predictor type (interpolation or Lorenzo), data encoder type (Huffman or SPECK), and their corresponding parameters (the ones in interpolation and SPECK encoding). Moreover, as its name implies, this tuning is based on a joint optimization of bit rate versus diverse user-specified quality (distortion) metrics such as PSNR, SSIM, autocorrelation, and null (meaning just maximizing compression ratio).

Figure 3.7 shows the entire process of FAZ rate-distortion tuning, including candidate pipeline generation, lightweight compression test, and rate-distortion evaluation.

Step 1: Candidate pipeline generation

In the candidate pipeline generation step, with different compression modules and different parameter settings, FAZ builds up several compression pipelines as candidates. To ensure each candidate pipeline correctly exposes the compression functionality, there are certain rules for the combination of the modules in FAZ due to their characteristics:

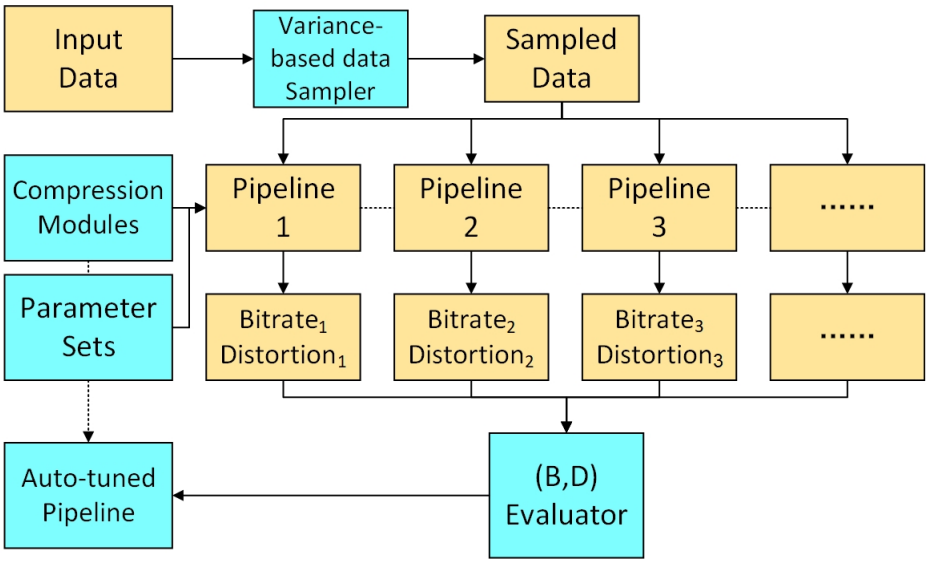


Figure 3.7: The rate-distortion tuning process. Blue blocks are related to newly proposed techniques.

- At most one and at least zero data prediction/transform module can be leveraged.
- SPECK encoding should only be used on the wavelet coefficients.
- Error quantization needs to be applied after data prediction and SPECK encoding.
- Huffman encoding always needs to be applied to the error quantization bins.

Since FAZ also needs to determine several parameters in the rate-distortion tuning, pipelines with the same structure but different parameter settings are regarded as different ones. After careful experiments for balancing the speeds and compression qualities of FAZ, the candidate set for each parameter is specified as follows:

- The α and β in QoZ interpolation: When compressing the non-transformed data, FAZ follows the same setting as in QoZ [65]. When a wavelet module is applied, the α candidates are reduced to $\{1.5,1.75\}$ and β candidates are reduced to $\{2.0,3.0\}$.

- The error threshold of SPECK encoding: Compared with SPERR [49] which fixes it to 1.5e (e is the absolute error bound), FAZ selects it from {0.75e,1.0e,1.25e} when the tuning target is not "maximizing compression ratio" as in initial tests we've found that applying smaller thresholds can lead to quite better rate-distortion. For the "maximizing compression ratio" target, the candidate set is set to {1.5e,1.75e,2.0e}.

Step 2: Light-weight compression test

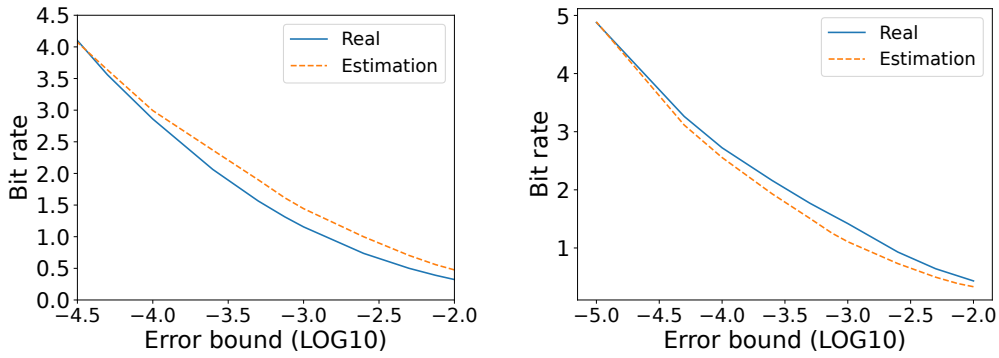
In this step, FAZ performs a lightweight compression test with each of the candidates on the sampled data, obtaining the bit rate and distortion in this compression as the estimation of those metrics for the practical compression. It will not cost a relatively lot of time because the sampled data are very little (typically 0.5% of the whole input data).

Step 3: Rate-distortion evaluation

Last, an evaluator compares all the (bit-rate, distortion) pairs and eventually selects the compression pipeline with the best (bit-rate, distortion) pair. The method FAZ leverages for comparing the (bit-rate, distortion) pairs is mainly imported from QoZ [65], as its proposed method is adaptive to heterogeneous compression pipelines. The brief introduction for this method is: to pick up the better one from two (bit-rate, distortion) pairs (b_1,d_1) and (b_2,d_2) generated by 2 pipelines is trivial if either $b_1 \leq b_2$ and $d_1 \geq d_2$ (the former is better) or $b_1 \geq b_2$ and $d_1 \leq d_2$ (the latter is better) holds (without loss of generality this dissertation assumes that higher d is better). In other cases, FAZ will run another compression test (on the sampled data) with one of the pipelines under a new error bound, then apply a linear interpolation/extrapolation to determine which of the pipelines can have a better distortion metric under the same bit rate.

Beyond the above comparisons, FAZ has an additional feature for the evaluation: the evaluator dynamically adjusts the bit rates from the pipelines with certain coefficients depending on the pipeline types and error bounds. This design is derived from the finding that the bit rate estimations generated by the compression tests on the sampled data blocks are inaccurate for some types of compression pipelines. Therefore, some active adjustments are needed in the evaluation step. Particularly, the bit-rate estimations for the pipelines including wavelet transforms would be higher than the practical ones whereas the bit-rate estimations for the pipelines including the Lorenzo predictor would be lower than the practical ones. Testing with the snapshot of the QS field in the SCALE-LetKF dataset, Figure 3.8 presents 2 examples of the estimated bit rates from compression tests and the real compression bit rates with 2 compression pipelines: one leverages the CDF9/7 wavelet transform together with the SPECK encoding, the other compresses the data by Lorenzo predictor without a wavelet transform. In the plots of the figure, the x-axis is the log value of the value range-based error bound and the y-axis is the bit rate. This figure exhibits the different characteristics of the bit rate estimations for these 2 kinds of compression pipelines: the estimations for the wavelet-based pipeline are higher than the real results and the estimations for the Lorenzo-based pipeline are lower than the real ones. Moreover, the gap between the estimated and real bit rates is large when the error bound is relatively high but it will shrink when the error bound decreases.

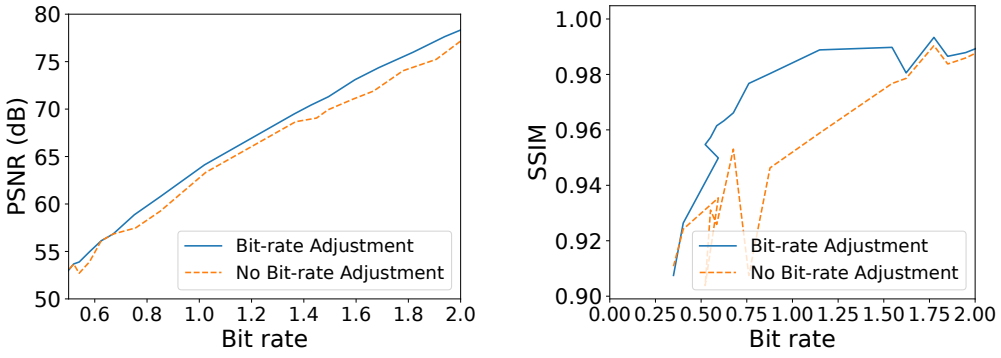
To resolve this issue, based on those findings FAZ introduces 2 coefficients $c_1 < 1$ and $c_2 > 1$ calculated from piecewise linear functions of the value range-based error bound (FAZ uses different functions for different wavelet types). Then for each raw bit



(a) CDF9/7+SPECK Encoding (b) No wavelet+ Lorenzo prediction

Figure 3.8: Comparison of estimated bit rate (in compression test) and real bit rate with different compression pipelines on SCALE-LetKF dataset (QS field).

rate estimation b_1 of the wavelet-based compression pipeline and b_2 of the Lorenzo-based compression pipeline, the corresponding adjusted bit rates $b_1^{'}$ and $b_2^{'}$ for evaluation are: $b_1^{'} = c_1 b_1$ and $b_2^{'} = c_2 b_2$. Figure 3.9 compares the rate-distortion (both PSNR and SSIM) of FAZ leveraging the proposed bit-rate adjustment or not on the Hurricane dataset. The plots verify that leveraging the specific bit-rate adjustment technique largely improves the accuracy and stability of auto-tuning as well as the practical rate-distortion.



(a) Rate-PSNR (b) Rate-SSIM

Figure 3.9: Bit-rate adjustment on Hurricane dataset.

3.7 Performance Evaluation

This section presents the evaluation results of FAZ compared with six other state-of-the-art error-bounded lossy compressors. Those works are either leading related works that are well-adopted and analyzed by many prior studies [55, 58, 61, 57, 103, 70], or the most recent works with outstanding performance [65, 49]. The evaluations are conducted thoroughly in multiple aspects and are based on seven widely used real-world scientific datasets among different domains.

3.7.1 Experimental setup

Execution Environment: The experiments are performed on the Argonne Bebop supercomputer which features over 2,000 nodes, and they use the bdwall nodes of it each having Intel Xeon E5-2695v4 CPU with 64 CPU cores and a total of 128GB of DRAM.

Evaluation Datasets: We evaluate the lossy compressors using seven real-world scientific applications from different domains which have been frequently used for the evaluation of scientific data error bounded lossy compression [102]:

- RTM: Reverse time migration for seismic imaging [43].
- SEGSalt: The 3D SEG/EAGE Salt model [3].
- NYX: A cosmological hydrodynamics simulation based on adaptive mesh [72].
- Hurricane: Simulation of Hurricane Isabel from the National Center for Atmospheric Research [38].

- SCALE-LetKF: Local Ensemble Transform Kalman Filter (LETKF) data assimilation package for the SCALE-RM weather model [2].
- Miranda: Large-eddy simulation of multi-component turbulent flows via a radiation hydrodynamics code [1].

The detailed information of the datasets is in Table 3.1.

Table 3.1: Information of the datasets in experiments

App.	# fields	Dimensions	Total Size	Domain
RTM	11	449×449×235	2.0GB	Seismic Wave
SEGSalt	3	1008×1008×352	4.2GB	Geology
Miranda	7	256×384×384	1GB	Turbulence
SCALE-LetKF	13	98×1200×1200	6.4GB	Climate
NYX	6	512×512×512	3.1GB	Cosmology
Hurricane	13	100×500×500	1.3GB	Weather

Comparison of lossy compressors in evaluation: The experiments compare FAZ with six other lossy compressors, including four well-adopted state-of-the-art works (SZ2.1, SZ3, ZFP0.5.5, MGARD+) [55, 58, 61, 57] and two very recent works: QoZ [65] and SPERR [49] which have been verified to have over-performing compression ratios and/or qualities compared to the others.

Experimental configurations: In the compression experiments, as broadly used in the compression community [55, 58, 57, 103] the error bound mode adopted is value-range-based error bound (denoted as ϵ). It is essentially equivalent to the absolute error bound (denoted as e), with the relationship of $e = \epsilon \cdot value\_range$.

For the hyper-parameter configuration of FAZ, the sample block size is set to 64 and the anchor point stride (for interpolation predictors) is set to 32. For pipeline auto-

tuning, 0.5% of the input data points are sampled. For all the other compressors, their default configurations are applied.

Evaluation metrics: The evaluations for FAZ and comparison compressors are based on several important metrics:

- Compression ratio (CR) under the same error bound: Compression ratio is the metric mostly cared for by the users. Given the input data denoted as X, compressed data denoted as Z, and $|\ |$ is the size operator, the compression ratio CR is: $CR = \frac{|X|}{|Z|}$.
- *Rate-PSNR plots*: Plot the crucial bit rate and PSNR [83] curves for compressors.
- *Rate-SSIM plots*: Another rate distortion evaluation plotting bit rate and SSIM [93].
- Error visualization with the same CR: Comparing the visualizations of the compression errors from different compressors based on the same CR.
- Compression/decompression speed: Check the compression/decompression speeds of compressors including FAZ to verify the computational costs.
- Parallel I/O performance: A parallel data transfer evaluation on a supercomputer.

3.7.2 Experimental results and analysis

Compression ratios with same error bounds

First, we'd like to compare the compression ratios of all lossy compressors under the same certain error bounds. In the experiments, FAZ is auto-tuned with the target of maximizing the compression ratio. Table 3.2 shows the compression ratios of the 7 lossy compressors on the 7 datasets under 3 error bounds (1e-2, 1e-3, 1e-4). The last column

(**OurSol Improve**) presents the improvements in compression ratio by FAZ compared with the second-best compressors. It is observed that FAZ achieves the best compression ratios under all of the cases. As an example, FAZ's compression ratio is 130.6% higher than the second-best compressor SPERR when the error bound is set to 1e-4 on RTM dataset, and its compression ratio is 91.5% higher than the second-best compressor SPERR when the error bound is set to 1e-4 on SEGSalt dataset. On the Miranda dataset, FAZ can have a compression ratio at most 25.6% better than the second-best compressor. On other datasets, it can also achieve 0-25% compression ratio improvements.

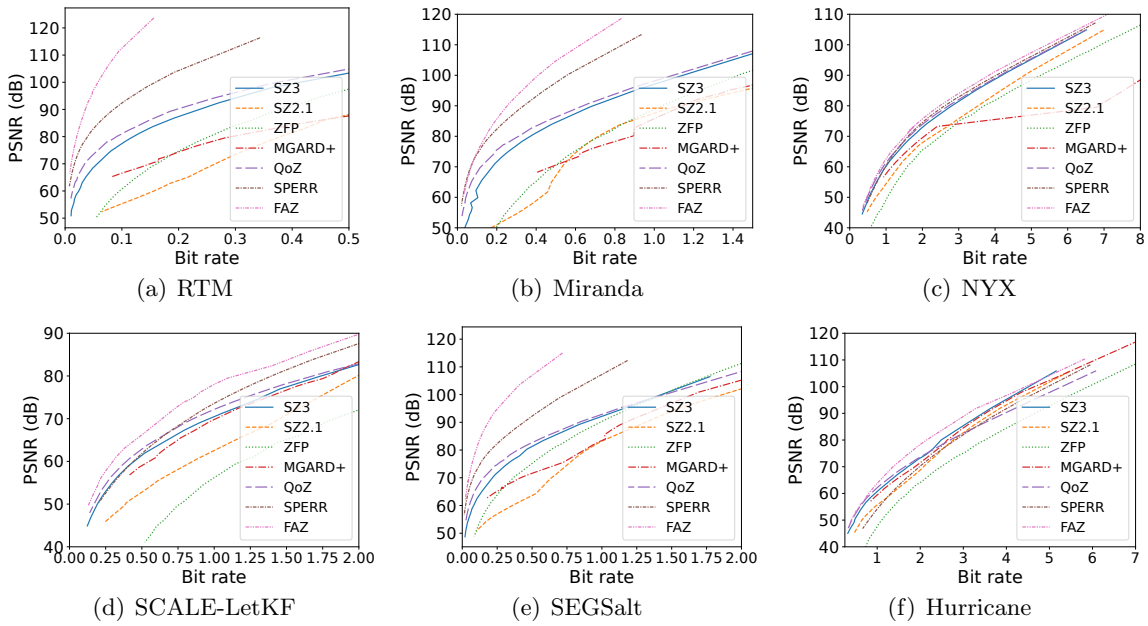
Table 3.2: Compression ratios under the same error bounds

Dataset	ϵ	SZ 2.1	SZ 3	ZFP	MGARD +	QoZ	SPERR	FAZ (Ours)	OurSol Improve
RTM	1E-2	283.3	2041.6	110.9	234.2	2461.2	2687.5	**3759.6**	39.9%
	1E-3	106.8	417.0	59.2	78.5	447.8	720.1	**1245.0**	72.9%
	1E-4	54.4	118.1	35.0	38.3	122.3	223.0	**514.3**	130.6%
Miranda	1E-2	126.3	574.6	46.6	52.1	987.2	971.4	**996.5**	0.9%
	1E-3	59.5	168.0	25.6	26.2	177.1	243.9	**263.5**	8.0%
	1E-4	29.6	47.3	14.5	12.9	48.2	74.5	**93.6**	25.6%
SEGSalt	1E-2	187.8	856	59.1	107.6	907.4	1219.4	**1639.6**	34.5%
	1E-3	50.8	140.6	24.9	35.2	151.9	228.9	**388.9**	69.9%
	1E-4	25.3	38.2	14.9	18.9	38.2	61.3	**117.3**	91.4%
SCALE	1E-2	84.0	167.3	14.5	53.8	163.4	103.5	**177.9**	6.3%
	1E-3	26.5	40.4	7.8	20.3	41.8	35.5	**51.8**	23.9%
	1E-4	13.9	14.1	4.6	10.4	13.4	15.0	**16.8**	12.0%
NYX	1E-2	44.1	61.3	12.0	24.7	62.0	48.0	**63.5**	2.4%
	1E-3	17.1	21.5	6.0	11.2	21.7	20.0	**22.5**	3.7%
	1E-4	7.7	9.1	3.7	5.5	9.2	8.8	**9.3**	1.1%
Hurricane	1E-2	49.8	69.0	11.3	28.4	70.3	35.6	**71.1**	1.1%
	1E-3	17.5	21.8	6.7	12.7	22.2	16.1	**24.6**	10.8%
	1E-4	9.8	**10.5**	4.3	7.4	9.3	8.7	**10.5**	0%

Rate-distortions

In this subsection, the evaluations of the compressors in terms of rate-distortion are presented. For the distortion metrics, this dissertation has tested with both PSNR and SSIM to examine whether FAZ has the adaptability to optimize various quality metrics during compression.

Figure 3.10 is the rate-PSNR plots (x-axis is bit rate and y-axis is PSNR) of the compressors on 6 datasets in which FAZ leverages the tuning target of PSNR. From the plots, readers can easily observe that FAZ achieves the best rate-PSNR curve on all shown datasets. Specifically, on the RTM dataset, FAZ has a 135%/190% compression ratio improvement over the second-best compressor SPERR when the PSNR is about 90/110. On the Miranda dataset, FAZ has a 40% compression ratio improvement over the second-best compressor SPERR when the PSNR is about 85. On the Hurricane dataset, FAZ has a 25% compression ratio improvement over the second-best compressor SZ3 when the PSNR is about 76. In terms of SSIM, Figure 3.11 provides the rate-SSIM plots of each tested lossy compressor. From the plots, it can be concluded that on the experimented datasets FAZ is still mostly the best choice for the SSIM metric. For example, on the SEGSalt dataset, FAZ has a 70% compression ratio improvement over the second-best compressor SPERR when the SSIM is about 0.8. On the SCALE-LetKF dataset, FAZ has a 75% compression ratio improvement over the second-best compressor QoZ when the SSIM is about 0.98. On the NYX dataset, FAZ has a 30% compression ratio improvement over the second-best compressor MGARD+ when the SSIM is about 0.9.



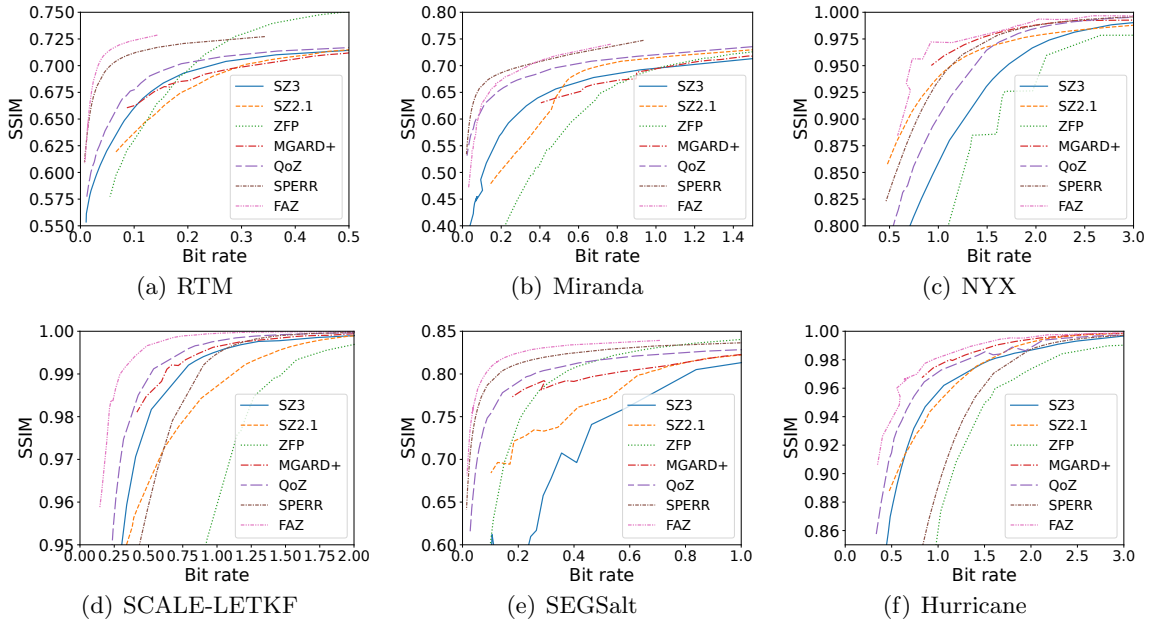
(a) RTM (b) Miranda (c) NYX

(d) SCALE-LetKF (e) SEGSalt (f) Hurricane

Figure 3.10: Rate-distortion (PSNR) of the compressors.

Visualizations

Figure 3.12 demonstrates the low compression errors of FAZ decompressed data. This figure presents the original data visualization of the #3000 snapshot in the SEGSalt dataset and 3 visualizations of the compression errors with 3 different lossy compressors (FAZ, SPERR, and QoZ) under very close compression ratio (about 93-94). The error visualizations are on the same scale (-0.015 to 0.015) and the compression errors of FAZ are quite closer to zero (the green background) than the other 2 compressors, well preserving the data values even under this high compression ratio.



(a) RTM (b) Miranda (c) NYX

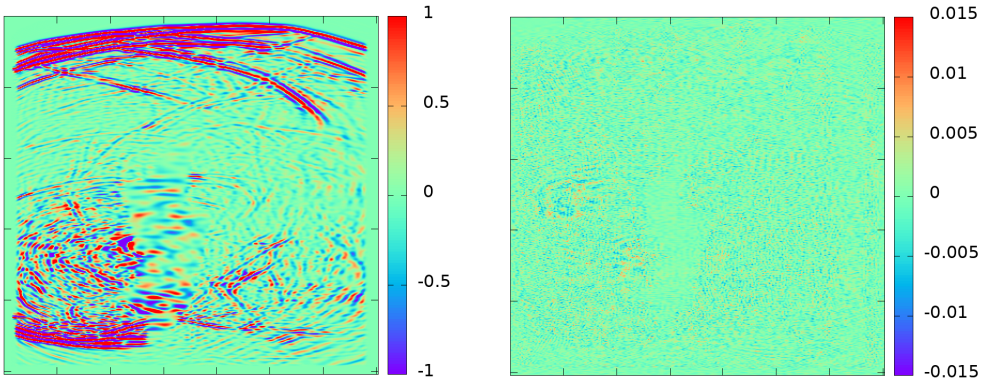
(d) SCALE-LETKF (e) SEGSalt (f) Hurricane

Figure 3.11: Rate-distortion (SSIM) of the compressors.

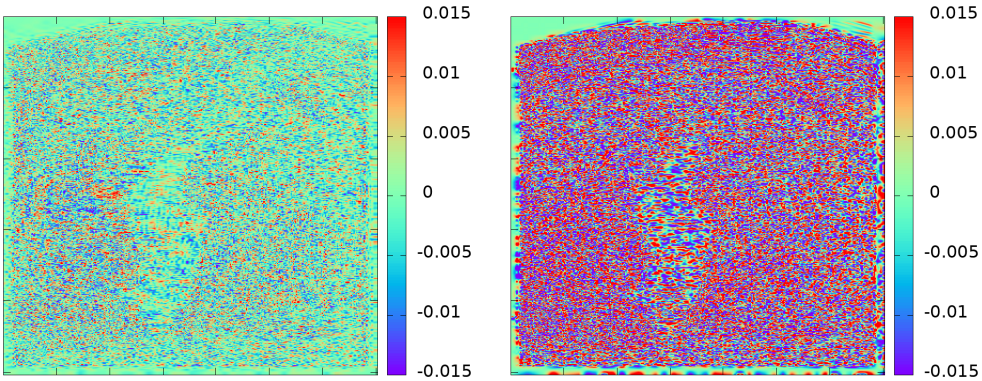
Ablation study of compression modules

The experimental results from different aspects have verified that FAZ has outperformed all the existing state-of-the-art works with respect to both compression ratios and compression qualities. This section will take deep insights into why FAZ has become the best and how the newly integrated modules contribute to the performance of FAZ. With several systematic ablation studies, this dissertation can thoroughly analyze and understand the reasons for the improvements from FAZ over existing works.

The first reason for the good performance of FAZ is the newly integrated wavelet modules. Figure 3.13 (a) shows the rate-PSNR curves on the SEGSalt dataset with QoZ, FAZ, and different configurations between them. From the plot, it can be observed each additional module (Sym13 wavelet, parameter auto-tuning and SPECK encoding) has con-



(a) Original Data (b) FAZ(CR=93.3,PSNR=86.0)



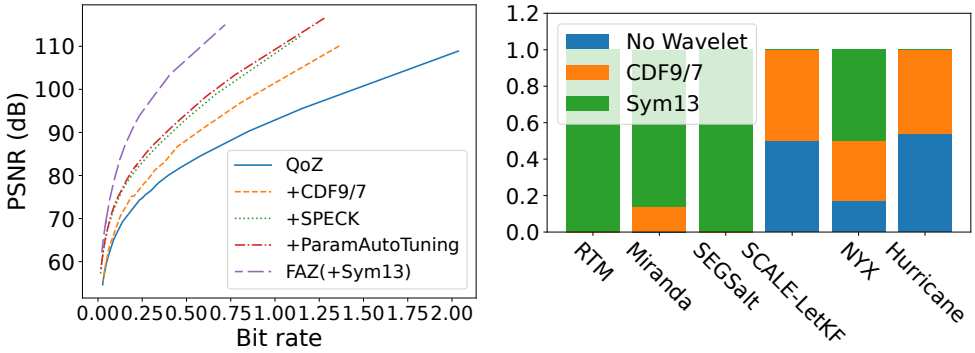
(c) SPERR(CR=94.6,PSNR=75.6) (d) QoZ(CR=93.9,PSNR=68.6)

Figure 3.12: Visualization of data distortion under different compressors (SEGSalt dataset).

tributed to the compression quality of FAZ, among which the Sym13 wavelet has a significant impact on improving the rate-distortion.

Another essential factor in FAZ is that it achieves the optimized compression rate-distortion on various types of inputs by dynamically selecting different wavelet transform types for data preprocessing. Figure 3.13 (b) is a bar chart showing the rates of data snapshots on which FAZ applies a certain type of wavelet transform (CDF9/7, Sym13 or not using wavelet) for each dataset under error bound 1e-3. It verifies that FAZ dynamically selects the best-fit wavelet transform usage for each input data snapshot, and each usage plays an important role in improving the rate-distortion. On RTM, SEGSalt, and Miranda

datasets, Sym13 is the most effective wavelet transform, but on other datasets, CDF9/7 would be a better choice. Moreover, for certain data snapshots in SCALE-LetKF, NYX, and Hurricane datasets, not applying a wavelet transform before the data prediction is the optimized solution for their compression. Benefiting from this dynamic selection of wavelets, the rate-distortions of FAZ outperform any other existing works.



(a) Ablation study on SEGSalt

(b) Wavelet rates (ϵ = 1e-3)

Figure 3.13: Ablation studies.

Compression speeds

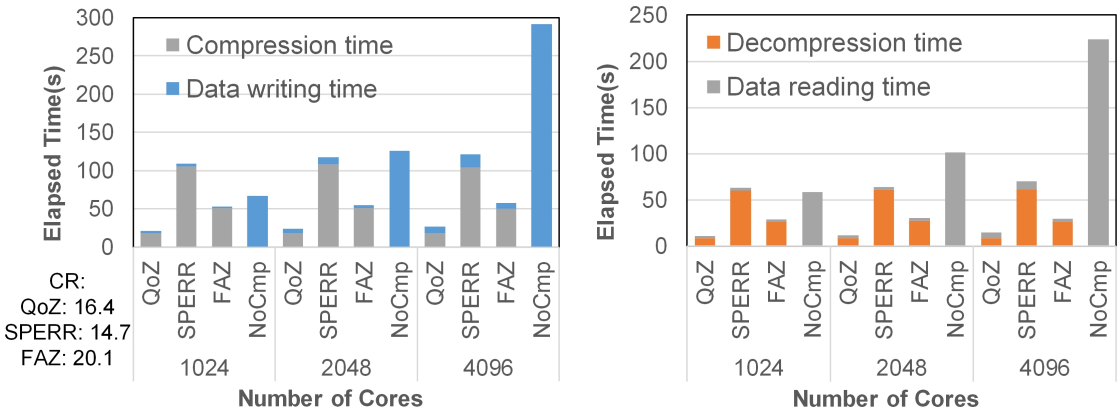
In this part, this dissertation discusses the speed issues of FAZ comparing it with different existing compressors. Firstly, the sequential compression and decompression speed of FAZ and the others (under the error bound of 1e-3, in which QoZ and FAZ apply the PSNR preferred mode) are presented in 3.3. From this table, readers can find a limitation of FAZ and other wavelet-integrated lossy compressors: they will have relatively lower speeds than the state of the arts without the wavelet transform module because the wavelet transform itself has inevitably slow execution speed, especially for the Sym13 wavelet which has a quite large kernel size (26). On the tested datasets, the speed of FAZ is between 0.15x

and 0.45x of SZ3/QoZ, depending on the rate of data snapshots it applies wavelet transform on and the type of wavelet it applies. Therefore, for speed-oriented use cases that may need repetitive compression and decompression processes for the data, FAZ is possibly not a favored compressor choice. Nevertheless, FAZ achieves comparable speed with SPERR, and its speed would be acceptable for use cases in which the data are not compressed too many times but for faster data transmission or better storage space saving the compression ratio or rate distortion is more taken into account. In those cases, FAZ with the optimized compression rate distortion can still be the best choice.

Table 3.3: Sequential speeds (MB/s) with ϵ=1e-3

Type	Dataset	SZ 2.1	SZ 3	ZFP	MGARD +	QoZ	SPERR	FAZ (Ours)
Compression	RTM	207	147	556	142	129	8.5	21
	Miranda	201	134	239	149	124	32.4	23
	SEGSalt	187	134	394	138	128	34	19
	SCALE	158	135	131	143	131	26	51
	NYX	181	98	149	131	97	17.5	19
	Hurricane	159	127	137	152	119	21	43
Decompression	RTM	452	410	996	210	388	20	40
	Miranda	404	374	659	212	350	65	43
	SEGSalt	394	363	732	213	328	69	45
	SCALE	305	359	362	202	342	47	105
	NYX	281	172	281	148	169	32	41
	Hurricane	266	279	321	196	278	38	99

Figure 3.14 presents and compares some data dumping and loading performances of FAZ and other compression solutions (QoZ, SPERR, and no compression) in the Hurricane simulation with 1K-4K cores in which the decompression results are in similar PSNR (73) and each core possesses a fixed amount of 1.3GB data. Due to the relatively low se-



(a) Data dumping (b) Data loading

Figure 3.14: Parallel performance evaluation on Hurricane dataset (compression ratios on bottom-left).

quential speeds of FAZ, its parallel performances are not quite competitive with QoZ in this case. However, in large-scale parallel applications, FAZ can show some advantages in terms of performance. We can observe that, with the best compression ratio (20.1) under this PSNR, FAZ spends less time in data dumping and loading than leveraging SPERR or no compressors. Although the parallel performance of FAZ is not as good as QoZ, when the user needs a wavelet transform-related compressor for maximizing the compression ratio, FAZ would still be the best choice in terms of both rate-distortion and parallel speeds.

Chapter 4

SRN-SZ: Scientific Error-Bounded Lossy Compression with Super-Resolution Neural Network

4.1 Overview

4.1.1 Motivations and challenges: Addressing the low-compressibility scientific data compression with new technical routines

In the last 2 chapters, this dissertation details about how to construct effective and efficient scientific data compression frameworks with relatively traditional data prediction techniques. Besides that, This dissertation would also like to explore the possibility of designing scientific lossy compressors with very up-to-date techniques. The fast-developing neural networks have shown their strength in a lot of fields, including the compression of

natural data such as images, videos, and so on. Naturally, it is worth exploring whether they can be effective for scientific error-bounded lossy compression.

This dissertation first clarifies why deep learning techniques for scientific lossy compression will be needed. Over the past dates, the scientific error-bounded lossy compression has been well developed with fast-evolving compression frameworks on diverse technical fundamentals. The new compressors achieved profound compression ratio and quality improvements over old works. However, if we switch our view to some other datasets, especially for the ones with quite lower compressibility, the newly proposed compressors did not achieve as much improvement as users have expected (corresponding experimental results presented in Section 4.3). What is more embarrassing is that the low compressibility data are often the bottleneck of the whole scientific database. Therefore, the challenge of low-compressibility data needs to be seriously addressed, and it motivates this dissertation to exploit new data modeling strategies, for example, deep learning neural networks.

There are several advantages and challenges for deep-learning-based scientific lossy compression. The advantages are that neural networks have a high ability to model complicated data and are adaptive to various data distributions. The challenges are also multi-fold. First, a trained neural network mostly has a fixed structure, which means that it may not be able to fit different compression requirements such as different error bounds. Second, scientific datasets from different domains may show very distinct data patterns. Therefore, the trained network in one scientific domain may be hard to apply to another. This meanwhile sets limits on the amount of training data, since a separate fine-tuning of the network may be needed for every single domain.

4.1.2 Contributions of SRN-SZ

Several attempts have been made to leverage neural networks in error-bounded lossy compression. Autoencoder-based AE-SZ [64] and Coordinate network-based CoordNet [32] are two typical examples. Those deep learning-based compressors may provide well-optimized compression ratios in certain cases, but their limitations are still obvious. The Coordinate network-based compressors [32, 36, 71] suffer from extremely low compression efficiencies as they need to train a new network online for each input. Although autoencoder-based compressors such as [64, 33] can leverage pre-trained networks to avoid per-input training, their compression ratios cannot overperform SZ3 in most cases [64].

In order to address the issues of optimizing the low-compressibility data compression and overcoming the limitations of deep-learning-based error-bounded lossy compression, this chapter proposes SRN-SZ, which is a grand new deep-learning-based error-bounded lossy compression framework. The core innovation of SRN-SZ is that it abstracts the compression and decompression processes of scientific data grids into a hierarchical paradigm of data grid super-resolution, which is the first work of integrating the super-resolution neural network into the error-bounded lossy compressor. Compared with the autoencoders and coordinate networks, the super-resolution networks have two-fold advantages: Unlike coordinate networks, they can be pre-trained before the practical compression tasks. At the same time, they do not generate any latent information that is required to be stored for compression as the autoencoders. Benefiting from those advantages, SRN-SZ achieves acceptable efficiencies and further improved compression ratios over the state-of-the-art error-bounded lossy compressors on multiple low-compressibility datasets.

The contributions of this chapter are detailed as follows:

- This dissertation proposes a new scientific error-bounded lossy compressor SRN-SZ, in which the compression is performed by hierarchical data grid expansion implemented with a hybrid of super-resolution networks and interpolations.
- Leveraging the Hybrid Attention Transformer (HAT) network, this dissertation designs a specialized training pipeline with several adaptive techniques to optimize the super-resolution quality of scientific data.
- This dissertation carries out systematical evaluations with SRN-SZ and 5 other state-of-the-art scientific error-bounded lossy compressors on various scientific datasets from different domains. According to the experimental results, SRN-SZ has achieved up to 75% compression ratio improvements under the same error bound and up to 80% compression ratio improvements under the same PSNR.

4.2 Related Work

This section discusses the related works of SRN-SZ in 3 categories: Traditional scientific error-bounded lossy compression, deep learning-based scientific lossy compression, and super-resolution neural networks.

4.2.1 Traditional scientific error-bounded lossy compression

Traditional scientific error-bounded lossy compressors can be classified into prediction-based, transform-based, and dimension-reduction-based. The prediction-based compressors utilize different data prediction techniques for the compression, such as linear regression

(SZ2 [55]) and interpolations (SZ3 [101] and QoZ [65]). Transform-based compressors decorrelate the input data by data transformation techniques so that the transformed data (a.k.a., coefficients) turn out to be much easier to compress than the original dataset; then it compresses the efficient domain to get a high compression ratio. Typical examples include ZFP [61] leveraging orthogonal discrete transform and SPERR [49] integrating CDF 9/7 wavelet transform. With dimension reduction techniques such as (high-order) singular vector decomposition (SVD), dimension-reduction-based compressors such as TTHRESH [11] can perform the data compression very effectively. Besides the CPU-based compressors, there are also several GPU-specialized error-bounded lossy compressors have also been developed and proposed for better parallelization and throughput. Typical examples are cuSZ [87, 86], cuSZp [37], and FZ-GPU [97].

4.2.2 Deep-learning-based scientific lossy compression

The great success of the recent research of Artificial Intelligence techniques started boosting the development of several other relevant research fields, including the scientific error-bounded lossy compression. Several research works that leverage deep neural networks in error-bounded lossy compression have been proposed [64, 32, 36, 71, 33]. There are mainly 2 archetypes: autoencoder-based compressors which store the autoencoder-encoded latent vectors for compression, and coordinate network-based compressors which train networks online for each input to map the data coordinates to data values. For autoencoder-based compressors, AE-SZ is an example of integrating Slice-Wasserstein autoencoders, and Hayne et al. [33] leverages a double-level autoencoder for compressing 2D data. Examples of coordinate network-based compressors include NeurComp [71], CoordNet [32] and [36].

4.2.3 Super-resolution neural networks

Following the SRCNN [26] which introduced a Convolutional neural network model to the image super-resolution tasks, a large number of convolutional neural network models [17, 80, 4, 7, 59, 99] have been proposed for the super-resolutions. Because of the development of Transformer [90] and its adaption to Computer Vision tasks [27, 69, 94], vision-transformer-based neural networks like [16, 53, 18] have achieved state-of-the-art performance on the image super-resolution task. Among those works, HAT [18] is the most impressive one as it has the widest scope of feature extraction for reconstructing each data point with a hybrid attention model and achieves state-of-the-art performance.

4.3 Problem Formulation and Backgrounds

4.3.1 Research target

The objective of SRN-SZ is to optimize the compression process with regard to a certain optimization target: maximizing compression PSNR under each certain compression ratio. Mathematically speaking, given the input data X, compressed data Z, decompression output X', error bound e, and the target compression ratio T, SRN-SZ will optimize its compressor C and decompressor D via the following optimization problem ($Z = C(X)$ and $X' = D(Z)$):

$$\begin{aligned} & maximize \;\; PSNR(X, X') \\ s.t. \quad & \tfrac{|X|}{|Z|} = T \\ & |x_i - x_i'| \leq e, \; \forall x_i \in X. \end{aligned} \quad , \tag{4.1}$$

4.3.2 Challenge for error-bounded lossy compression: low-compressibility datasets

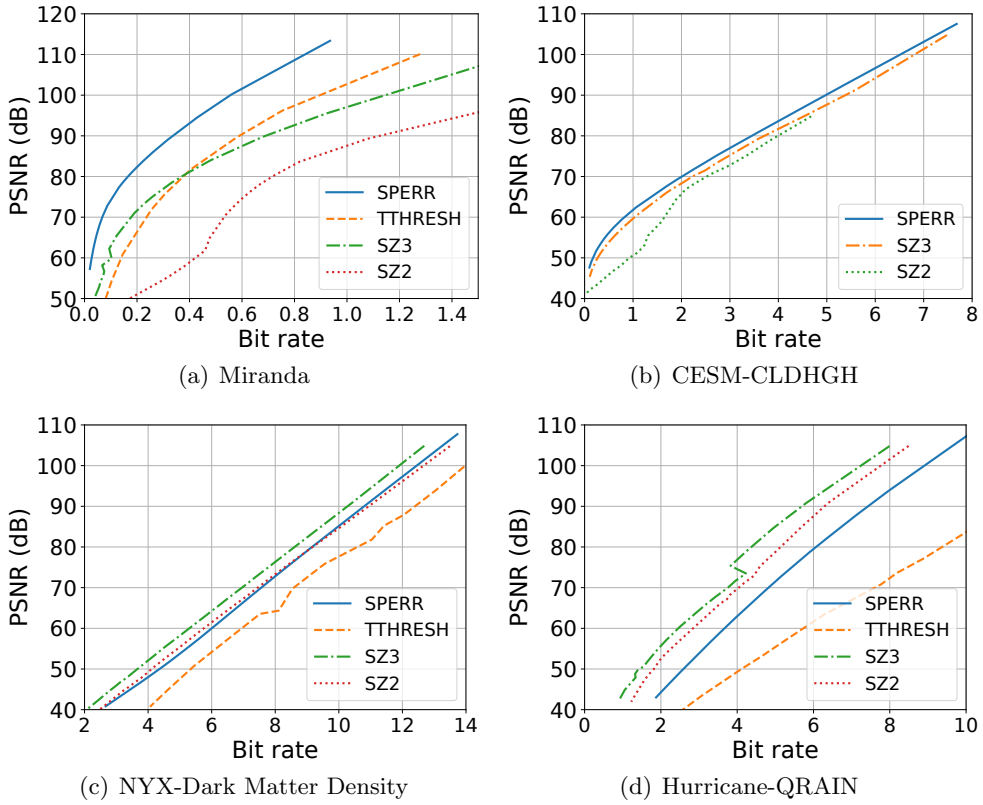
Recently proposed scientific error-bound lossy compressors have succeeded in outperforming the old state-of-the-art compressors dramatically. Compared with the historical SZ 2.1 [55], SZ3 [58] has improved the compressor ratio by up to 460% [101] under the same data distortion. With higher computational costs, wavelet-based compressors such as SPERR [49] may have doubled or even tripled compression ratios compared with SZ3.

However, those exciting improvements in compression ratios are just concentrated on datasets that generally project relatively high compression ratios (e.g. over 100). In other words, the recent proposed works with advanced data compression techniques fail to improve the compression for datasets with relatively low compression ratios to similar extents as they have done in high-ratio cases. Figure 4.1 presents the bit rate-PSNR curves from the compression of 4 scientific datasets with the representative existing error-bounded lossy compressors: prediction-based SZ2 [55] and SZ3 [101, 58], SVD-based TTHRESH [11], and wavelet transform-based SPERR [49] (the compression result of TTHRESH is not shown in Figure 4.1 (b) as TTHRESH does not support 2D data input). For datasets like the Miranda [1] (Figure 4.1 (a)). SZ3 has boosted the compression ratio of SZ2 by over 100%, and SPERR further achieves 2x-3x of the compression ratio over SZ3. However, on other datasets, those 4 compressors have relatively low compression ratios. On certain datasets such as NYX-Dark Matter Density and Hurricane-QRain (Figure 4.1 (c) and (d)), the SPERR and TTHRESH have lower compression ratios than SZ3 does, though they are designed with more complicated data processing techniques and higher computational costs.

It is worth noting that the low-compressibility data snapshots are actually the bottleneck of compression effectiveness because their compressed data size will obviously occupy a very large portion of all data fields (having diverse characteristics) in a single dataset. For example, compressing 100TB of data with a compression ratio of 100 will generate 1TB of compressed data, which means that it can at most save the space of 1TB when optimizing the compression. Nevertheless, if the original data has the same size of 100TB but only has a potential compression ratio of 5 (20TB compressed data), merely improving the compression ratio by 10% will lead to around 1.8TB storage cost reduction. Therefore, overcoming the limitation of existing compressors on low-compressibility data will be significant for optimizing the overall compression process for a large variety of scientific simulation datasets.

4.4 SRN-SZ Design Overview

SRN-SZ is a deep-learning-based error-bounded lossy compressor and is based on a modular compression framework that integrates a hybrid data reconstruction model with both interpolators and super-resolution neural networks. As shown in Figure 4.2, the compression framework of SRN-SZ consists of 4 modules: Data grid sparsification, data grid expansion, Huffman encoding, and Zstd lossless compression. Moreover, the super-resolution neural networks are first pre-trained with a large-size dataset assorted from the scientific database and then fine-tuned with domain-specific datasets before being leveraged in the data grid expansion module of SRN-SZ. In the compression process of SRN-SZ, it first extracts a sparse data grid from the original data input, next, this sparse data grid



(a) Miranda (b) CESM-CLDHGH

(c) NYX-Dark Matter Density (d) Hurricane-QRAIN

Figure 4.1: Rate-distortion (PSNR) of several existing error-bounded compressors.

is expanded step by step with super-resolution networks and interpolators, eventually to a lossy reconstruction of the full-size input grid. Compared to existing deep learning-based compressors which leverage autoencoder-like networks [64, 68] to generate compact representations or coordinate networks [32, 36, 71] mapping data point indices to data values, SRN-SZ has the advantages of both free from the storage cost for the compact representations (required by autoencoders) and per-input network training (required by coordinate networks).

This dissertation demonstrates the detailed compression algorithm of SRN-SZ in Algorithm 1. Lines 1-2 correspond to data grid sparsification, Lines 3-10 correspond to

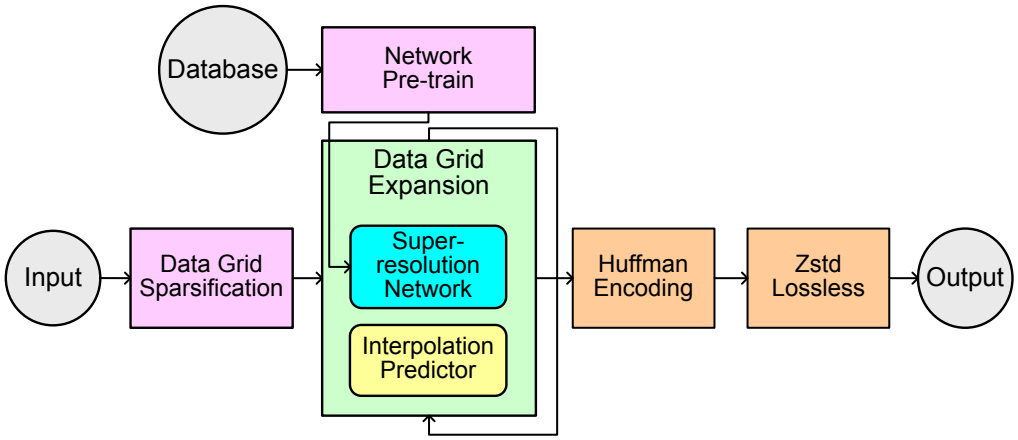


Figure 4.2: SRN-SZ compression framework.

data grid expansion, and Lines 11-12 correspond to Huffman encoding and Zstd lossless compression. To bound the point-wise compression error, the linear quantization is involved in the data grid expansion module, and for clearness of demonstration, it is not displayed in Figure 4.2.

4.5 SRN-SZ Compression Pipeline

This section describes the steps in the SRN-SZ Compression pipeline in detail. Since the encoding and lossless modules of SRN-SZ are the same as the ones in SZ3 and QoZ [101, 58, 65], the following subsections will mainly discuss the data grid sparsification and data grid expansion.

4.5.1 Data grid sparsification

Having shown advantages in MGARD [5, 57], SZ3 [101, 58], and QoZ [65], SRN-SZ adopts a level-wise hierarchical paradigm for its compression process. It starts from a sparse

Algorithm 1 SRN-SZ Compression Algorithm

Input: Input data D, error-bound e, grid sparsification rate r, minimum SRN size s

Output: Compressed data Z

Sparsify D into D_0 with rate r. Save D_0 losslessly /*Data grid sparsification.*/

Set current reconstructed data grid $D^{'} \leftarrow D_0$, Quantized errors $Q \leftarrow \{\}$

while $size(D^{'}) != size(D)$ **do**

    if $size(D^{'}) \leq s$ **then**

        $D^{'}, q = Interp\_and\_Quantize(D, D^{'}, e)$/*Expand $D^{'}$ with interpolation.*/

    else

        $D^{'}, q = HAT\_and\_Quantize(D, D^{'}, e)$ /*Expand $D^{'}$ with HAT network.*/

    end if

    $Q \leftarrow Q \bigcup q$. /*Merge newly acquired quantized errors q.*/

end while

$H \leftarrow$ Huffman_Encode(Q). /*Huffman encoding*/

$Z \leftarrow Zstd(H, D_0)$. /*Zstd compression*/

data grid sampled from the original input dataset. An example of 2D input data is shown in Figure 4.3: certain data points are uniformly sampled from the full data grid with a fixed stride. Those sampled data points in a sparsified data grid will be losslessly saved and the rest data points will be reconstructed in the data grid expansion process. The reason SRN-SZ losslessly saves the sparsified grid instead of directly reconstructing a lossy version of it from scratch as SZ3 does is analyzed below. According to the comparison between evaluations of SZ3 and QoZ [65], for the hierarchical level-wise data reconstruction, an accurate base is essential for preserving the high reconstruction quality of the data points, meanwhile only introducing negligible overhead storage space overhead. To balance the compression ratio loss and data reconstruction accuracy, some tests are conducted and then the dimension-wise rate of data grid sparsification is specified as $\frac{1}{32}$, i.e., reduce the data grid to $\frac{1}{32}$ along each dimension and then save the sparsified grid for the data grid expansion.

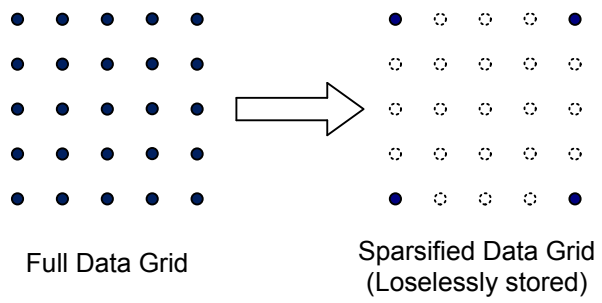


Figure 4.3: Data grid sparsification.

4.5.2 Data grid expansion

Based on the sparsified data grid, the data grid expansion (i.e. reconstruction) process is involved in both the compression and decompression of SRN-SZ. In the compression, the data grid expansion is executed for acquiring the reconstruction errors of data points, and then those errors are quantized and encoded serving as the correction offsets in the decompression. Moreover, During both the compression and decompression process of SRN-SZ, the super-resolution and error-quantization in compression (or error correction in decompression) are executed alternately, which can maximally preserve the accuracy of data grid expansion. As presented in Figure 4.4, the data grid expansion is performed iteratively step by step, until the whole data grid has been reconstructed. In each step, the reconstructed data grid is expanded by 2x along each dimension, therefore its implementation is compatible with both the deep learning-based super-resolution neural networks and the traditional interpolation methods.

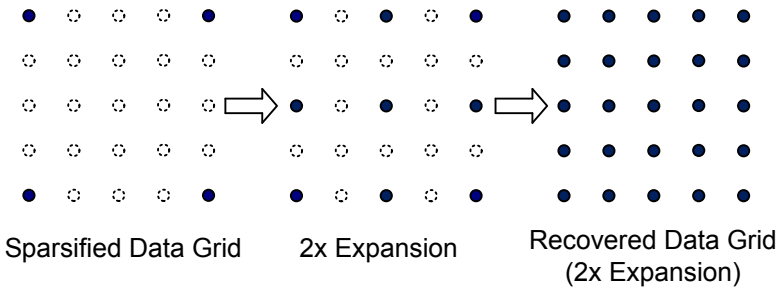


Figure 4.4: Data grid expansion.

HAT super-resolution network

Super-resolution network is the most important data grid expansion technique in SRN-SZ as it is always applied on the last iteration step of data grid expansion, which contains the reconstruction for most of the data points in the input data (about 75% for 2D case and about 87.5% for 3D case). The network SRN-SZ leveraged is the HAT (Hybrid Attention Transformer) network [18], which is a very recent proposed work for image super-resolution and has been proven to be state-of-the-art. The network architecture of HAT is illustrated in Figure 4.5. Developed from [53, 99], HAT is a very-deep residual [34] neural network with transformers [90] as its basic components. HAT has 3 main modules: the initial convolutional layers for shallow feature extraction, the deep feature extraction module integrated with residual hybrid attention groups (RHAG), and a reconstruction module leveraging the Pixel Shuffle technique [77]. The RHAG blocks in the HAT network can be broken down into HAB (hybrid attention block), OCAB (overlapping cross-attention block), and convolutional layers. The main advantage of HAT is that according to the analysis presented in [18], the design of HAT empowers it to make use of a large region of data points for computing each value in its super-resolution output. Therefore, both local and global data patterns can be well utilized in the super-resolution process.

Although HAT was originally designed for the super-resolution of natural images, this dissertation managed to adapt it to the scientific data grid expansion process in SRN-SZ. Feeding an intermediate data grid with size X x Y (or X x Y x Z) into HAT, SRN-SZ uses the super-resolution output of size 2X x 2Y (or 2X x 2Y x 2Z) from HAT as the data grid expansion result in one step. Some key points in bridging the scientific data and the HAT

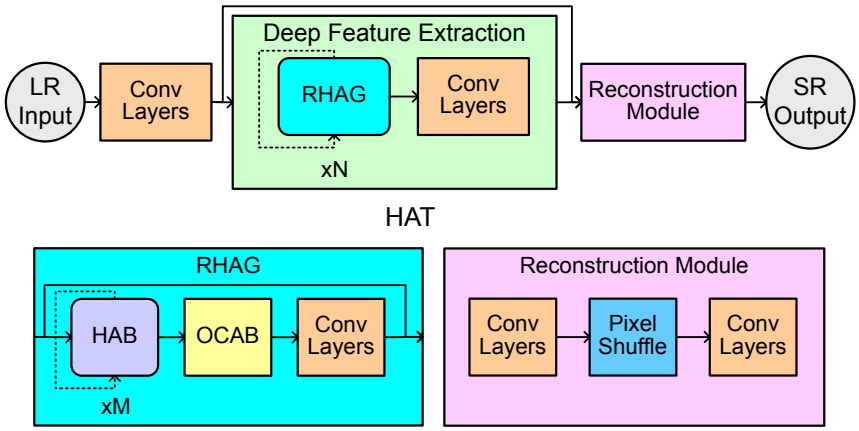


Figure 4.5: HAT network.

network are: First, the input and output channels in HAT have been modified from 3 to 1. Second, the input data grid is normalized to 0-1 before being fed into the network. Last, for 3D data inputs, 2D HAT models can still be used, but the inputs are preprocessed into 2D slices (along all the 3 dimensions) instead of 3D blocks. The reason SRN-SZ applies 2D networks for 3D data is that 3D HAT models suffer from extremely high computational time costs for training and inference, presenting unacceptable flexibility and scalability. Figure 4.6 presents the details of performing 3D super-resolution with those 2D slices. Specifically, with a partially reconstructed 3D data grid (blue points), SRN-SZ performs super-resolution on it with the HAT network in 3 different directions: on top/bottom faces (red points), on left/right faces (green points), and on front/back faces (purple points). The super-resolution results on the edges are the average of 2 directions, and the point on the cubic center is reconstructed by a multi-dimensional spline interpolation, which is introduced in [67] and will be detailed in the next subsection and Figure 4.7 (b).

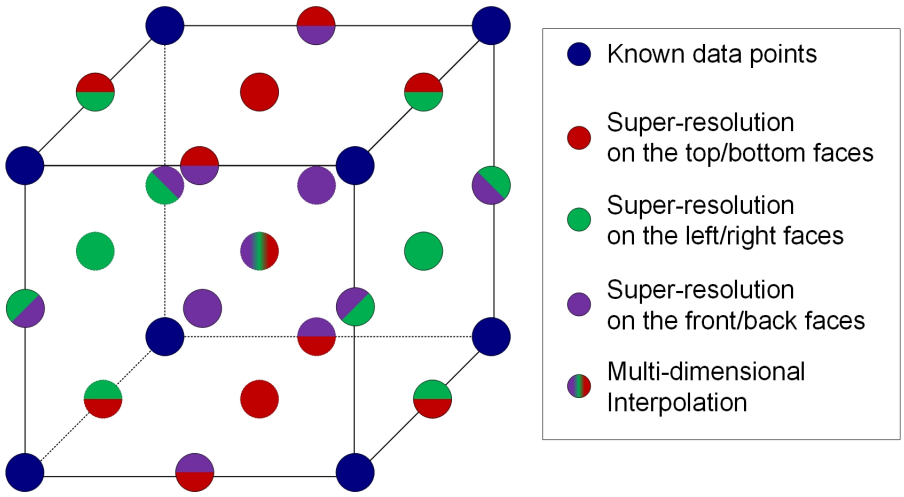


Figure 4.6: 3D super-resolution with 2D slices.

interpolation-based data predictor

It is observed that, when the reconstructing data grid has a small size, the super-resolution network can not work well. Therefore, on some initial steps of data grid expansion in which the current data grid is smaller than a threshold (with a dimension shorter than 64), the traditional QoZ-based interpolation [65] is leveraged for the grid expansion which can auto-tune the best-fit interpolation configurations and error bounds. In addition to the QoZ interpolation, following the design proposed by [67], SRN-SZ also leverages several advanced interpolation designs such as multi-dimensional spline interpolation. Figure 4.7 presents and compares these two interpolation methods, and SRN-SZ will dynamically select the interpolation method for each interpolation level. This adaptive selection design improves both the efficiency of SRN-SZ and the reconstruction quality in the early steps of data grid expansion.

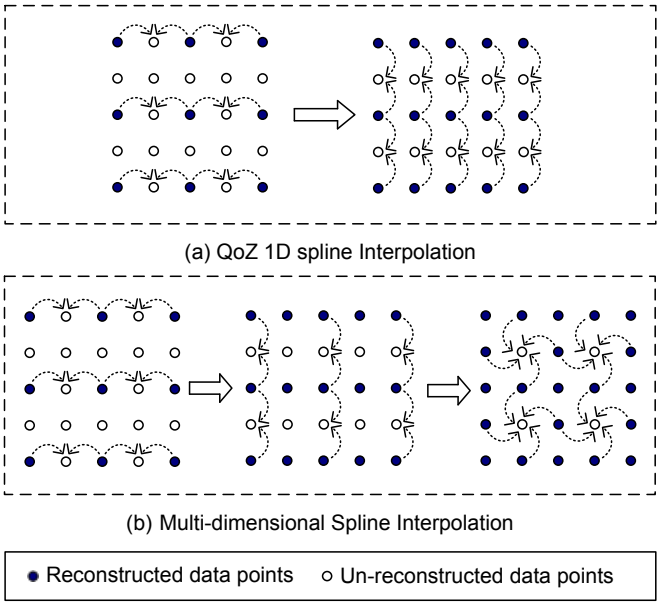


Figure 4.7: Interpolations in SRN-SZ.

4.6 SRN-SZ Network Training

The super-resolution quality of the HAT network plays the most important role in optimizing the compression ratio with controlled data distortion in SRN-SZ, and the core of optimizing the super-resolution quality of the HAT is its training process. The HAT networks in SRN-SZ are pre-trained offline both with an assorted dataset and domain-specific datasets. This design contributes to the flexibility and adaptability of SRN-SZ. Several strategies have been proposed for optimizing the training of the HAT networks in SRN-SZ. Figure 4.8 proposes our designed HAT network training pipeline for SRN-SZ. In the pipeline, each network is trained for two rounds: the general training from scratch and the domain-specific training for fine-tuning. The following subsections describe the key design of this pipeline.

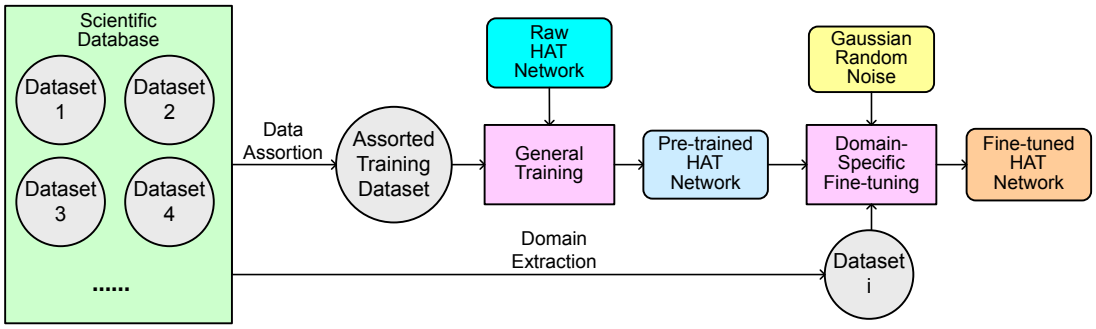


Figure 4.8: SRN-SZ network training pipeline.

4.6.1 Training data collection and preprocessing

This dissertation has collected training data snapshots from a variety of well-known scientific simulations, including CESM-ATM [42], RTM [43], OCEAN, Miranda [1], JHTDB [52], Hurricane-ISABEL [38], SCALE-LetKF [2], NYX [72], and so on. The full list of the scientific simulations used by SRN-SZ for the HAT network training is shown in Table 4.1. With those assorted data snapshots, this dissertation first decomposes 3D data arrays into 2D data slices, next normalize them to [0,1] range, then split all over-sized (over 480x480) slices into smaller slices (480x480) according to the setting in [18]. When yielding the training data batches, the low-resolution, and high-resolution image pairs are randomly cropped from those slices. The widely-used image data augmentation methods like random flip and rotation are excluded from SRN-SZ network training as it is observed that those data augmentation strategies will harm the quality of super-resolution with test data. This assorted and pre-processed dataset will be used for general pre-training of the HAT network.

Table 4.1: Information of the scientific simulations for training data of SRN-SZ

App.	Dimensions	Per snapshot size	Domain
CESM-ATM	2D	1800×3600	Climate
Hurricane	3D	100×500×500	Weather
JHTDB	3D	512×512×512	Turbulence
Miranda	3D	256×384×384	Turbulence
NYX	3D	512×512×512	Cosmology
OCEAN	2D	2400×3600	Oceanology
RTM	3D	449×449×235	Seismic Wave
Scale-LETKF	3D	98×1200×1200	Climate

4.6.2 Domain-specific fine-tuning

Datasets from different scientific domains and simulations would present diverse patterns and characteristics. To make the trained network better adapt to more varied inputs, SRN-SZ needs to fine-tune its super-resolution for certain scientific simulations that are being intensively and consistently used for research and analysis. To address this issue, SRN-SZ developed a domain-specific fine-tuning mechanism. After an initial training phase with the assorted database, SRN-SZ picks up several additional data snapshots generated by those simulations and then fine-tunes the network separately with each simulation data. In this way, SRN-SZ can achieve improved compression ratios on multiple widely used scientific data simulation datasets.

4.6.3 Denoise training with Gaussian random noise

As discussed in Section 4.5.2, the data grid to be expanded in SRN-SZ is a lossy sample from the original data input. At the same time, SRN-SZ will need the super-

resolution of it to fit the original input as much as possible. To simulate this process in the training of the HAT networks in SRN-SZ for better super-resolution results, this dissertation proposes denoise training in SRN-SZ. Specifically, instead of simply using full data grids and the corresponding down-sampled data grids as the training data pairs, SRN-SZ adds Gaussian noise to the down-sampled data grids before feeding them into the network in the training phase. In this way, the trained network will be capable of de-noising the input for more accurate super-resolution outputs. Moreover, it is observed that training networks with intense noises will damage their effectiveness on low error-bound cases, so SRN-SZ separately trains 3 base networks with different intensities of noises: strong noise (with stand derivation of 1% of data range), weak noise (with stand derivation of 0.1% of data range), and no noise. Those networks will correspondingly serve for different compression cases: high error bounds (larger than 1e-2), medium error bounds (1e-4 to 1e-2), and low error bounds (smaller than 1e-4).

4.7 Performance Evaluation

This section describes the setup of experiments, and it presents the experimental results with in-depth analysis. This dissertation evaluates the SRN-SZ and compares it with five other state-of-the-art error-bounded lossy compressors [58, 49, 65, 11, 66].

4.7.1 Experimental setup

Experimental environment and datasets

The experiments are conducted on the Argonne Bebop supercomputer (for CPU-based tests) and the ALCF Theta supercomputer (for GPU-based tests). On the Bebop machine, its nodes of the bdwall partition are used, having an Intel Xeon E5-2695v4 CPU with 64 CPU cores and a total of 128GB of DRAM on each. On the Theta machine, each GPU node of it has 8 NVIDIA TESLA A100 GPUs.

6 data fields from 4 real-world scientific applications in diverse scientific domains are selected for evaluation. Those datasets are frequently used for evaluating scientific error-bounded lossy compression [102]. The information about the datasets and the fields is detailed in Table 4.2. As suggested by domain scientists, some fields of the datasets listed above are transformed to their logarithmic domain for better visualization. For fairness of evaluation, the data snapshots used for the evaluations are never contained in the assorted training dataset and their corresponding fine-tuning datasets. However, for optimizing the compression, some data snapshots in the same data field (but from different runs of the application or from different time steps) are used for training (especially for fine-tuning).

Table 4.2: Information of the datasets in experiments

Name	# fields	Dimensions	Domain
CESM-ATM	CLDHGH, FREQSH	1800×3600	Climate
Ocean	TMXL	2400×3600	Oceanology
NYX	Dark Matter Density	512×512×512	Cosmology
Hurricane	QRain, QGraup	100×500×500	Weather

Comparison of lossy compressors in evaluation

In the experiments, SRN-SZ is evaluated together with five other state-of-the-art lossy compressors. Among those, 4 are the traditional error-bounded lossy compressors: SZ3 [58], QoZ [65], SPERR [49], and FAZ [66]. Another one is the deep learning-based AE-SZ [64], which was verified in [64] to be one of the most effective autoencoder-based error-bounded lossy compressors. This dissertation does not perform comparison experiments with coordinate-network-based compressors due to the reason they suffer from very low compression speed (much slower than SRN-SZ) as they need to perform a network training process for each single compression task [32, 36, 71].

Network training configurations

The training of HAT networks in SRN-SZ applies the network structure and training configurations described in [18]. In each training phase (including general training and domain-specific fine-tuning), the network is trained on 8 GPUs in 200,000 iterations with a mini-batch size of 32. The initial learning rate is 2e-4 and will be halved on step [100K,160K,180K,190K]. For the network training and compression of AE-SZ, this dissertation follows the configurations described in [64].

Evaluation metrics

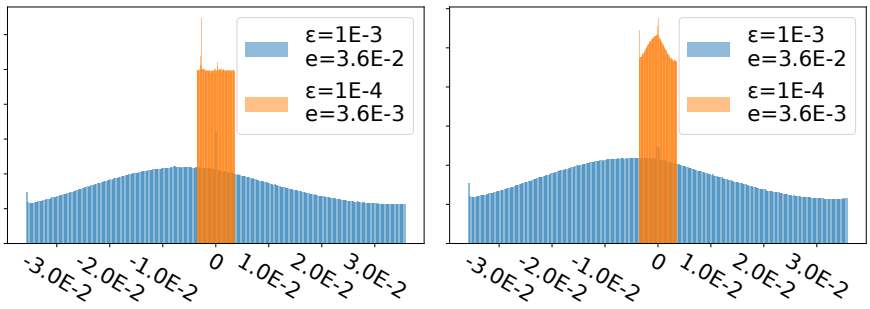
In the compression experiments, the value-range-based error bound mode (denoted as ϵ) is adopted, which is equivalent to the absolute error bound (denoted as e) with the relationship of $e = \epsilon \cdot value\_range$. The evaluations are based on the following key metrics:

- Decompression error verification: Verify that the decompression errors are strictly error-bounded.
- Compression ratio (CR) under the same error bound: Compression ratio is the metric mostly cared for by the users, for fair comparison, the compression ratios under fixed error bounds are presented.
- *Rate-PSNR plots*: Plot curves for compressors with the compression bit rate and decompression PSNR.
- Visualization with the same CR: Comparing the visual qualities of the reconstructed data from different compressors based on the same CR.
- Ablation Study: Verify the effectiveness of each SRN-SZ design component separately.

4.7.2 Evaluation results and analysis

Verification of compression errors versus error bound

First of all, this dissertation verifies that the decompression errors from SRN-SZ have strictly been constrained within the error bounds. To this end, it plots the histograms of decompression errors for each compression task, and two of them (on the QRAIN and QGRAUP fields of the Hurricane-ISABEL dataset) are presented in Figure 4.9. It can be clearly observed that the decompression errors of SRN-SZ always respect the error bound (e) in all cases with no out-of-bound abnormalities of point-wise decompression error. Having examined the error-bounded feature of SRN-SZ, in the following subsections, this dissertation will test, present, and analyze the compression ratios and qualities of SRN-SZ.



(a) Hurricane-QRAIN (b) Hurricane-QGRAUP

Figure 4.9: Histograms of decompression errors from SRN-SZ.

Compression ratio under the same error bounds

The compression ratios of all lossy compressors under the same certain error bounds (1e-3, 1e-4, and 1e-5) are presented in Table 4.3. An interesting fact is that, although proposed later than SZ3, some new compressors (QoZ, SPERR, and FAZ) have not raised the compression ratios well on the tested datasets. In contrast, SRN-SZ has quite improved the compression ratios of error-bounded lossy compressors on almost all of the tested compression cases, over a variety of datasets and error bounds. Particularly, under the error bound of 1e-4 SRN-SZ achieves a 75% compression ratio improvement over the second-best QoZ on the CLDHGH field of the CESM-ATM dataset, and under the error bound of 1e-3 SRN-SZ achieves a 44% compression ratio improvement on the FREQSH field of it. On other datasets, SRN-SZ can also get 3% to 20% compression ratio improvements. Last, compared with other deep learning-based compressors, SRN-SZ has outperformed AE-SZ in an overall assessment.

Table 4.3: Compression ratio comparison based on the same error bounds

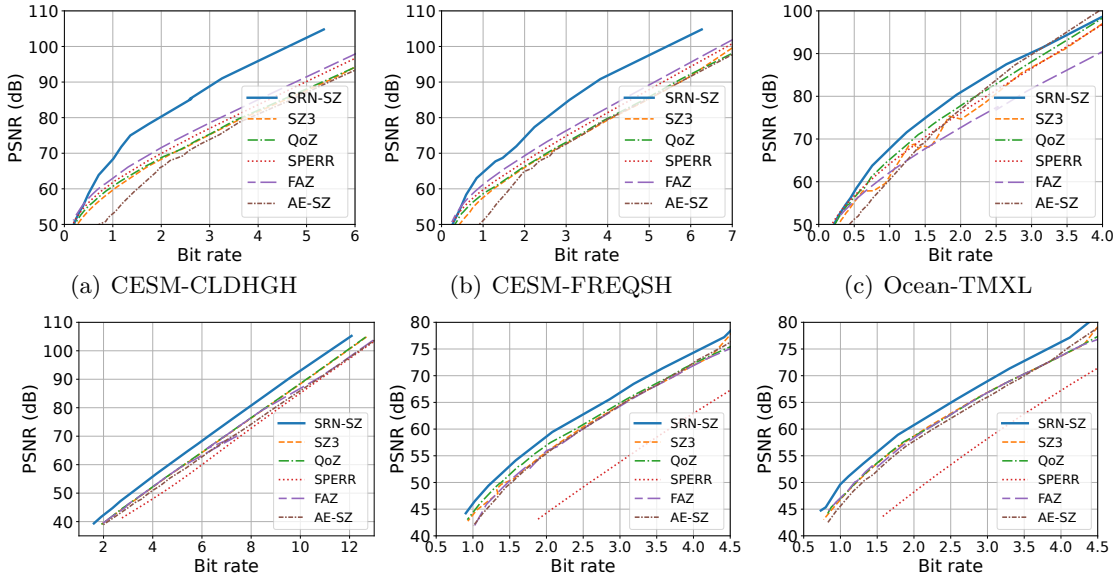
Dataset	ϵ	SZ 3.1	QoZ	SPERR	AE-SZ	FAZ	SRN-SZ	Improve (%)
CESM CLDHGH	1E-3	19.2	18.8	18.9	16.8	16.0	**32.6**	69.8
	1E-4	7.0	7.1	7.0	6.9	6.3	**12.4**	74.6
	1E-5	4.3	4.1	4.2	4.1	3.8	**6.0**	39.5
CESM FREQSH	1E-3	16.4	17.2	16.3	16.3	14.0	**24.7**	43.6
	1E-4	6.4	6.6	6.5	6.5	5.9	**10.4**	57.6
	1E-5	4.2	3.9	4.0	4.0	3.7	**5.1**	21.4
Ocean TMXL	1E-3	25.3	24.9	21.7	23.4	15.5	**29.4**	16.2
	1E-4	11.2	10.6	9.7	11.6	7.3	**12.1**	4.3
	1E-5	6.5	6.6	6.1	7.0	4.7	**7.0**	0.7
NYX DarkMatter Density	1E-3	5.2	5.3	4.5	5.1	4.3	**5.9**	11.3
	1E-4	3.4	3.4	3.1	3.3	3.0	**3.7**	8.8
	1E-5	2.5	2.5	2.3	2.4	2.2	**2.6**	4.0
Hurricane QRAIN	1E-3	10.0	10.3	6.9	10.3	10.1	**11.2**	8.7
	1E-4	**6.5**	5.3	4.5	5.8	4.2	6.4	-1.5
	1E-5	4.0	3.4	3.2	3.8	3.0	**4.1**	2.5
Hurricane QGRAUP	1E-3	11.2	11.2	7.7	11.0	11.2	**12.4**	10.7
	1E-4	**6.6**	5.5	4.8	6.2	4.7	6.0	-9.1
	1E-5	4.0	3.5	3.3	3.9	3.3	**4.3**	7.5

Rate-distortion evaluation

Next, this dissertation presents and analyzes the rate-distortion evaluation of SRN-SZ and other state-of-the-art error-bounded lossy compressors.

Figure 4.10 displays the rate-distortion evaluation results of each lossy compressor on all datasets. In the plots, the x-axis is bit rate and the y-axis is PSNR. SRN-SZ has the best rate-distortion curves on all the datasets. On the CESM-CLDHGH dataset, SRN-SZ achieves 60% to 80% compression ratio improvement than the second-best SPERR in the PSNR range of $70 \sim 80$. On the Ocean-TMXL dataset, SRN-SZ achieves $\sim$20% compression ratio improvement than the second-best QoZ in the PSNR range of $60 \sim 70$. Additionally, SRN-SZ overperforms all other compressors by about 5% to 15% compression ratio improvements on the rest of the datasets.

Those results show that, for certain datasets on which the traditional or autoencoder-based lossy compressors can only present limited compression ratios, SRN-SZ has the potential to optimize the compression of those datasets to a further extent, and the reasons can be attributed to 3 terms. First, those datasets have complex data characteristics and patterns for which traditional data modeling techniques cannot fit well. Second, the newly proposed compression framework of SRN-SZ enables the compressor to directly leverage a super-resolution network for the data prediction via data grid expansion (super-resolution) instead of applying a redundant autoencoder model for which the latent vectors must be stored (such as AE-SZ does). Third, the hybrid usage of interpolations and super-resolution networks makes the interpolation compensate for the limitation of neural networks when dealing with small data grids.



(a) CESM-CLDHGH (b) CESM-FREQSH (c) Ocean-TMXL

(d) NYX-Dark Matter Density (e) Hurricane-QRAIN (f) Hurricane-QGRAUP

Figure 4.10: Rate-distortion evaluation (PSNR).

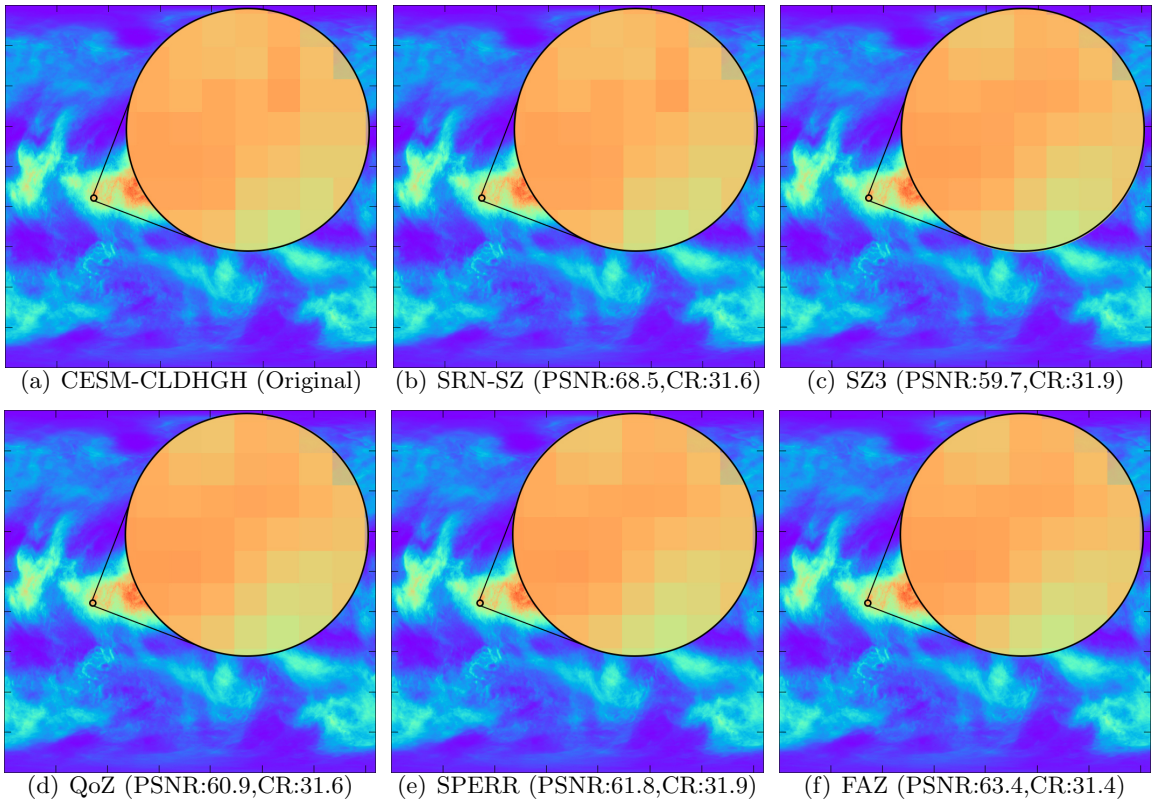
Visualization of decompressed data

As an example of the high compression quality of SRN-SZ, Figure 4.11 presents several visualizations of the decompression results of CESM-CLDHGH data field from multiple compressors, together with the original data as the reference. For a fair comparison, for each compressor, the data are compressed into a fixed compression ratio (around 32) and then decompressed. According to Figure 4.11 (the visualization of AE-SZ is omitted because it has poor quality with PSNR ≈ 53 under the specified compression ratio), in this case, the decompression data of SRN-SZ has the lowest distortion from the original input, with a PSNR of 68.5 which is 5 higher than the second-best FAZ. The zoomed regions also show that SRN-SZ has best preserved the local data patterns as well. The local visualization of SRN-SZ decompressed data is nearly identical to the original data, meanwhile, the ones of other compressors suffer from some quality degradation.

Ablation study of network training strategies

For verifying and understanding how the design details of SRN-SZ contribute to the overall compression quality, especially for the design components in the network pre-training pipelines, this dissertation conducts several ablation studies for the network pre-training, identifying and quantifying the contributions of the corresponding design components.

First, to examine the impact of domain-specific fine-tuning (described in Section 4.6.2) on the training of HAT networks in SRN-SZ. SRN-SZ is tested with networks free of domain-specific fine-tuning and then its compression rate-distortion is compared to the one from ordinary SRN-SZ. This comparison is detailed in Figure 4.12 with 2 examples



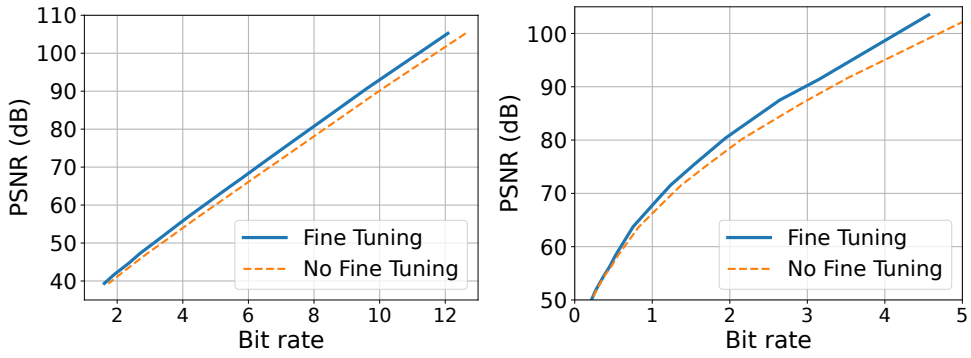
(a) CESM-CLDHGH (Original) (b) SRN-SZ (PSNR:68.5,CR:31.6) (c) SZ3 (PSNR:59.7,CR:31.9)

(d) QoZ (PSNR:60.9,CR:31.6) (e) SPERR (PSNR:61.8,CR:31.9) (f) FAZ (PSNR:63.4,CR:31.4)

Figure 4.11: Visualization of reconstructed data (CESM-CLDHGH).

presented (on Ocean-TMXL and NYX-Dark Matter Density). It is shown that the domain-specific fine-tuning process (the blue curves in Figure 4.12) can consistently improve the compression rate-distortion over the SRN-SZ without a network fine-tuning process (the orange curves in Figure 4.12).

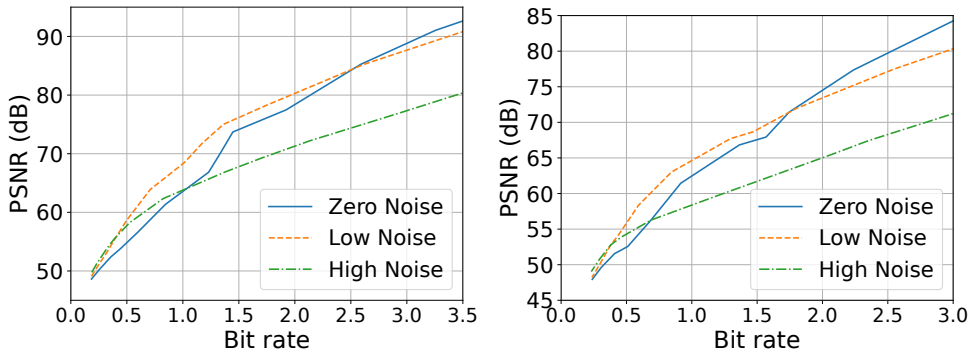
Next, this dissertation addresses the importance of SRN-SZ denoise training by analyzing and comparing the compression rate-distortion of SRN-SZ integrating fixed HAT networks each trained with a certain intensity of noise. In Figure 4.13, the rate-PSNR curves of SRN-SZ with HAT networks trained by 3 different levels of noise intensity (zero noise, low noise of σ=1e-3, and high noise of σ=1e-2) are illustrated. Those compressors exhibit



(a) NYX-Dark Matter Density (b) Ocean-TMXL

Figure 4.12: Ablation study for the domain-specific fine-tuning.

advantages over the others on different bit rate ranges. SRN-SZ with high-noise-trained overperforms the other configurations when the bit rate is smaller than 0.4 (corresponding to error bound > 1e-2). The low-noise-trained HAT network optimizes the SRN-SZ compression under medium bit rates, and when the bit rate is large (error bound < 1e-4), Leveraging networks trained with no noise achieves the best rate-distortion. Those results prove that the error-bound-adaptive dynamic usage of differently-trained HAT networks (with diverse noise intensities) essentially optimizes the compression of SRN-SZ.



(a) CESM-CLDHGH (b) CESM-FREQSH

Figure 4.13: Ablation study for the denoise training.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Scientific error-bounded lossy compression plays an irreplaceable role in ensuring the high-performance management and analysis of exascale scientific data in cutting-edge supercomputing systems. However, existing scientific error-bounded lossy compressors suffer from several critical limitations, including but not limited to under-optimized compression ratios, unstable compression qualities, and fixed outcomes from varied requirements. In order to enhance the utility of scientific error-bounded lossy compression in a larger variety of real-world use cases, this dissertation presents several deep explorations into different aspects of scientific error-bounded lossy compression, proposing diverse designs of task-adaptive scientific error-bounded lossy compressors. Those designs either feature high versatility for general compression requirements or high optimization levels for specialized compression tasks. They are applicable to a diversity of computing platforms and have overcome several critical limitations of existing compressors. Consequently, they push forward

the usability of scientific data compression to a brand new extent. The overall outcomes of this dissertation can be summarized as follows:

- This dissertation develops an efficient error-bounded lossy compression framework that can dynamically optimize different quality metrics in online compression. To this end, this dissertation leverages multiple advanced techniques, including block-wise anchor point structure, multi-level interpolation-based data prediction, level-wise predictor selection, and error-bound auto-tuning. this dissertation also develops a series of optimization strategies including block-wise interpolation tuning, dynamic dimension freezing, and Lorenzo tuning, which can substantially improve the adaptability of the auto-tuning for compression across a broad spectrum of inputs. Experiments show that QoZ outperforms other high-performance error-bounded lossy compressors in compression ratio by up to 140% under the same error bound, and by up to 360% under the same PSNR. In parallel data transfer experiments on the distributed database, QoZ achieves a significant performance gain with up to 40% time cost reduction over the second-best compressor.

- This dissertation proposes FAZ, which has a highly flexible and adaptive design with an effective pipeline auto-tuning method, leveraging effective data compression modules and avoiding their weaknesses. This dissertation evaluates FAZ and the other state-of-the-art error-bounded lossy compressors with 7 real-world datasets on up to 4K cores. Regarding compression ratio and rate-distortion, FAZ outperforms all other state-of-the-art error-bounded lossy compressors, achieving 120%, 190%, and 75% improvements under the same error bound, PSNR, and SSIM. FAZ also has comparable

or better compression/decompression speeds compared with other wavelet-based lossy compressors in sequential and parallel tests.

- This dissertation also introduces SRN-SZ, a deep learning-based error-bounded compressor that leverages one of the most advanced super-resolution neural network archetypes, namely HAT. SRN-SZ abstracts the data prediction process in compression into a hierarchical data grid expansion paradigm, enabling the utility of super-resolution neural networks for lossy compression. To exploit the advantages of different data reconstruction techniques, the data grid expansion in SRN-SZ is performed by a self-adaptive hybrid method of super-resolution HAT networks and interpolations. For the better adaptation of super-resolution networks to scientific data, SRN-SZ integrates a carefully designed network training pipeline for optimizing the network performance. In the evaluations, SRN-SZ outperforms all other state-of-the-art error-bounded lossy compressors in terms of compression ratio and rate-distortion, achieving up to 75% compression ratio improvements under the same error bound and up to 80% compression ratio improvements under the same PSNR.

5.2 Future Work

Scientific lossy compression is a fast-developing research field, therefore quite a few open research problems still lie in this area. After all the demonstrations of research achievements, this dissertation would like to present some discussion about certain unresolved research problems highly relevant to the research topics of this dissertation, which can both be the future work of the author and inspire the readers for their research.

5.2.1 New data prediction scheme for scientific compression

From Lorenzo data predictor, and linear regression, to spline-interpolation-based data predictor, the widely-adopted SZ data compressor has substantially evolved over its generations. The QoZ compressor proposed in Chapter 2 benefits a lot from the excellent spline-interpolation-based data predictor and has updated it with several major developments, but admittedly it has not presented brand new designs of data prediction. Therefore, a naturally raised question is, what would be the next-generation data predictor for scientific error-bounded lossy compression that can outperform spline-interpolation-based data predictor while having a comparable speed to it? The most challenging point is that spline-interpolation costs very limited numbers of numerical operations for the prediction of each data value, which makes a new high-accuracy design quite hard to match its efficiency. To this end, we must make a careful (and even dynamic) selection of the neighbor points for prediction and bring out well-established mathematical formulas. One idea is whether we can better exploit the high-dimensional locality information of the input data, instead of just leveraging 1D splines. Moreover, an adaptive design fitting different characteristics of input data may also be needed.

5.2.2 Efficiency-aware compression auto-tuning

The compressors proposed in this dissertation contain a variety of auto-tuning techniques and targets, but all of them focus on the optimization of compression ratio and quality. In efficiency-sensitive real-world use cases, users may look forward to different trade-offs between the compression quality and the speed. For example, one compressor

may be applied to different input formats (static database or yielding data stream) and different requirements (storage saving or fast processing), which calls for diverged needs of compression speed and quality. If the auto-tuning module of one compressor can involve its efficiency as a factor, it can serve better for a wider range of tasks that show distinct natures. One core challenge is that more flexible designs of compression framework and pipeline are needed to provide an adequate amount of efficiency levels. Another important challenge is that the compressor will be used on diverse platforms, so for the speed-aware compression auto-tuning a reliable strategy of compression speed profiling and/or estimation across different platforms will be needed.

5.2.3 More efficient neural networks for scientific compression

Although neural networks have improved the compression ratio on certain scientific datasets, their usage in practical cases is still limited because of their extremely high computational cost and low speed. To address this issue, future research works will need to deliver more scientific-customized designs of network structure and training/inference schemes. For example, it is worth exploring how to find the best trade-off of effectiveness and efficiency in pruning layers/weights and weight precision. Introducing new network modules (e.g. layers and activation functions) that are better optimized for scientific data can also boost the network's effectiveness without computational cost overheads.

Bibliography

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

[2] Scalable computing for advanced library and environment (scale) – letkf. https://github.com/gylien/scale-letkf.

[3] SEGSalt. https://wiki.seg.org/wiki/SEG/EAGE_Salt_and_Overthrust_Models.

[4] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 252–268, 2018.

[5] Mark Ainsworth, Ozan Tugluk, Ben Whitney, and Scott Klasky. Multilevel techniques for compression and reduction of scientific data—the univariate case. *Computing and Visualization in Science*, 19(5):65–76, 2018.

[6] Rachana Ananthakrishnan, Kyle Chard, Ian Foster, and Steven Tuecke. Globus platform-as-a-service for collaborative science applications. *Concurrency and Computation: Practice and Experience*, 27(2):290–305, 2015.

[7] Saeed Anwar and Nick Barnes. Densely residual laplacian super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1192–1204, 2020.

[8] Allison Baker, D Hammerling, and Terece Turton. Evaluating image quality measures to assess the impact of lossy data compression applied to climate simulation data. volume 38, 06 2019.

[9] Allison H. Baker, Alexander Pinard, and Dorit M. Hammerling. Dssim: a structural similarity index for floating-point data, 2022.

[10] Allison H. Baker, Haiying Xu, Dorit M. Hammerling, Shaomeng Li, and John P. Clyne. Toward a multi-method approach: Lossy data compression for climate simulation data. In *High Performance Computing*, pages 30–42. Springer International Publishing, 2017.

[11] Rafael Ballester-Ripoll, Peter Lindstrom, and Renato Pajarola. TTHRESH: Tensor compression for multidimensional visual data. *IEEE transactions on visualization and computer graphics*, 26(9):2891–2903, 2019.

[12] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *arXiv preprint arXiv:2003.05991*, 2020.

[13] Franck Cappello, Sheng Di, Sihuan Li, Xin Liang, Gok M. Ali, Dingwen Tao, Chun Yoon Hong, Xin-chuan Wu, Yuri Alexeev, and T. Frederic Chong. Use cases of lossy compression for floating-point data in scientific datasets. *International Journal of High Performance Computing Applications (IJHPCA)*, 33(6):1201–1220, 2019.

[14] Kyle Chard, Jim Pruyne, Ben Blaiszik, Rachana Ananthakrishnan, Steven Tuecke, and Ian Foster. Globus data publication as a service: Lowering barriers to reproducible science. In *2015 IEEE 11th International Conference on e-Science*, pages 401–410. IEEE, 2015.

[15] Kyle Chard, Steven Tuecke, and Ian Foster. Globus: Recent enhancements and future plans. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale*, pages 1–8, 2016.

[16] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12299–12310, 2021.

[17] Haoyu Chen, Jinjin Gu, and Zhi Zhang. Attention in attention network for image super-resolution. *arXiv preprint arXiv:2104.09497*, 2021.

[18] Xiangyu Chen, Xintao Wang, Jiantao Zhou, Yu Qiao, and Chao Dong. Activating more pixels in image super-resolution transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22367–22377, 2023.

[19] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *Advances in Neural Information Processing Systems*, 34:9355–9366, 2021.

[20] John Clyne, Pablo Mininni, Alan Norton, and Mark Rast. Interactive desktop analysis of high resolution simulations: application to turbulent plume dynamics and current sheet formation. *New Journal of Physics*, 9(8):301, 2007.

[21] Albert Cohen, Ingrid Daubechies, and J-C Feauveau. Biorthogonal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 45(5):485–560, 1992.

[22] Yann Collet. Zstandard – real-time data compression algorithm. *http://facebook.github.io/zstd/*, 2015.

[23] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 41(7):909–996, 1988.

[24] Sheng Di and Franck Cappello. Fast error-bounded lossy HPC data compression with SZ. In *IEEE International Parallel and Distributed Processing Symposium*, pages 730–739, 2016.

[25] Sheng Di, Jinyang Liu, Kai Zhao, Xin Liang, Robert Underwood, Zhaorui Zhang, Milan Shah, Yafan Huang, Jiajun Huang, Xiaodong Yu, Congrong Ren, Hanqi Guo, Grant Wilkins, Dingwen Tao, Jiannan Tian, Sian Jin, Zizhe Jian, Daoce Wang, MD Hasanur Rahman, Boyuan Zhang, Jon C. Calhoun, Guanpeng Li, Kazutomo Yoshii, Khalid Ayed Alharthi, and Franck Cappello. A survey on error-bounded lossy compression for scientific datasets, 2024.

[26] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part IV 13*, pages 184–199. Springer, 2014.

[27] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[28] Ziquan Fang, Yuntao Du, Lu Chen, Yujia Hu, Yunjun Gao, and Gang Chen. E 2 dtc: An end to end deep trajectory clustering framework via self-training. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 696–707. IEEE, 2021.

[29] Andrew Glaws, Ryan King, and Michael Sprague. Deep learning for in situ data compression of large turbulent flow simulations. *Physical Review Fluids*, 5(11):114602, 2020.

[30] A. M. Gok, S. Di, Y. Alexeev, D. Tao, V. Mironov, X. Liang, and F. Cappello. PaSTRI: Error-bounded lossy compression for two-electron integrals in quantum chemistry. In *2018 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 1–11, Sep. 2018.

[31] Salman Habib, Adrian Pope, Hal Finkel, Nicholas Frontiere, Katrin Heitmann, David Daniel, Patricia Fasel, Vitali Morozov, George Zagaris, Tom Peterka, et al. HACC: Simulating sky surveys on state-of-the-art supercomputing architectures. *New Astronomy*, 42:49–65, 2016.

[32] Jun Han and Chaoli Wang. Coordnet: Data generation and visualization generation for time-varying volumes via a coordinate-based neural network. *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[33] Lucas Hayne, John Clyne, and Shaomeng Li. Using neural networks for two dimensional scientific data compression. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 2956–2965. IEEE, 2021.

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[35] Christopher E Heil and David F Walnut. Continuous and discrete wavelet transforms. *SIAM review*, 31(4):628–666, 1989.

[36] Langwen Huang and Torsten Hoefler. Compressing multidimensional weather and climate data into neural networks. *arXiv preprint arXiv:2210.12538*, 2022.

[37] Yafan Huang, Sheng Di, Xiaodong Yu, Guanpeng Li, and Franck Cappello. cuSZp: An ultra-fast gpu error-bounded lossy compression framework with optimized end-to-end performance. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–13, 2023.

[38] Hurricane ISABEL simulation data. http://vis.computer.org/vis2004contest/data.html, 2004. Online.

[39] Søren Kejser Jensen, Torben Bach Pedersen, and Christian Thomsen. Modelardb: Modular model-based time series management with spark and cassandra. *Proceedings of the VLDB Endowment*, 11(11):1688–1701, 2018.

[40] Zizhe Jian, Sheng Di, Jinyang Liu, Kai Zhao, Xin Liang, Haiying Xu, Robert Underwood, Shixun Wu, Zizhong Chen, and Franck Cappello. CliZ: Optimizing lossy compression for climate datasets with adaptive fine-tuned data prediction. In *2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2024.

[41] Pu Jiao, Sheng Di, Hanqi Guo, Kai Zhao, Jiannan Tian, Dingwen Tao, Xin Liang, and Franck Cappello. Toward quantity-of-interest preserving lossy compression for scientific data. *Proceedings of the VLDB Endowment*, 16(4):697–710, 2022.

[42] J. E. Kay and et al. 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*, 96(8):1333–1349, 2015.

[43] Suha Kayum et al. GeoDRIVE – a high performance computing flexible platform for seismic applications. *First Break*, 38(2):97–100, 2020.

[44] Søren Kejser Jensen, Torben Bach Pedersen, and Christian Thomsen. Scalable model-based management of correlated dimensional time series in modelardb+. *arXiv e-prints*, pages arXiv–1903, 2019.

[45] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[46] Soheil Kolouri, Phillip E Pope, Charles E Martin, and Gustavo K Rohde. Sliced Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

[47] Sriram Lakshminarasimhan, Neil Shah, Stephane Ethier, Scott Klasky, Rob Latham, Rob Ross, and Nagiza F. Samatova. Compressing the incompressible with isabela: In-situ reduction of spatio-temporal data". In Emmanuel Jeannot, Raymond Namyst, and Jean Roman, editors, *Euro-Par 2011 Parallel Processing*, pages 366–379, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[48] Samuel Li, Stanislaw Jaroszynski, Scott Pearse, Leigh Orf, and John Clyne. VAPOR: A visualization package tailored to analyze simulation data in Eart system science. 07 2019.

[49] Shaomeng Li, Peter Lindstrom, and John Clyne. Lossy scientific data compression with sperr. In *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 1007–1017. IEEE, 2023.

[50] Wenbo Li, Xin Lu, Jiangbo Lu, Xiangyu Zhang, and Jiaya Jia. On efficient transformer and image pre-training for low-level vision. *arXiv preprint arXiv:2112.10175*, 3(7):8, 2021.

[51] Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S Jensen, and Wei Wei. Deep representation learning for trajectory similarity computation. In *2018 IEEE 34th international conference on data engineering (ICDE)*, pages 617–628. IEEE, 2018.

[52] Yi Li, Eric Perlman, Minping Wan, Yunke Yang, Charles Meneveau, Randal Burns, Shiyi Chen, Alexander Szalay, and Gregory Eyink. A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence. *Journal of Turbulence*, (9):N31, 2008.

[53] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1833–1844, 2021.

[54] Xin Liang, Sheng Di, Sihuan Li, Dingwen Tao, Bogdan Nicolae, Zizhong Chen, and Franck Cappello. Significantly improving lossy compression quality based on an optimized hybrid prediction model. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–26, 2019.

[55] Xin Liang, Sheng Di, Dingwen Tao, Sihuan Li, Shaomeng Li, Hanqi Guo, Zizhong Chen, and Franck Cappello. Error-controlled lossy compression optimized for high compression ratios of scientific datasets. In *2018 IEEE International Conference on Big Data*. IEEE, 2018.

[56] Xin Liang, Sheng Di, Dingwen Tao, Sihuan Li, Bogdan Nicolae, Zizhong Chen, and Franck Cappello. Improving performance of data dumping with lossy compression for scientific simulation. In *2019 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 1–11. IEEE, 2019.

[57] Xin Liang, Ben Whitney, Jieyang Chen, Lipeng Wan, Qing Liu, Dingwen Tao, James Kress, David R Pugmire, Matthew Wolf, Norbert Podhorszki, et al. Mgard+: Optimizing multilevel methods for error-bounded scientific data reduction. *IEEE Transactions on Computers*, 2021.

[58] Xin Liang, Kai Zhao, Sheng Di, Sihuan Li, Robert Underwood, Ali M Gok, Jiannan Tian, Junjing Deng, Jon C Calhoun, Dingwen Tao, et al. SZ3: A modular framework for composing prediction-based error-bounded lossy compressors. *IEEE Transactions on Big Data*, 2022.

[59] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[60] Z Lin, Hhm TS, WW Lee, WM Tang, and RB White. Turbulent transport reduction by zonal flows: massively parallel simulations. *Science*, 281(5384):1835, 1998.

[61] Peter Lindstrom. Fixed-rate compressed floating-point arrays. *IEEE transactions on visualization and computer graphics*, 20(12):2674–2683, 2014.

[62] Peter G Lindstrom et al. Fpzip. Technical report, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2017.

[63] Jinyang Liu, Sheng Di, Sian Jin, Kai Zhao, Xin Liang, Zizhong Chen, and Franck Cappello. Scientific error-bounded lossy compression with super-resolution neural networks. In *2023 IEEE International Conference on Big Data (BigData)*, pages 229–236. IEEE, 2023.

[64] Jinyang Liu, Sheng Di, Kai Zhao, Sian Jin, Dingwen Tao, Xin Liang, Zizhong Chen, and Franck Cappello. Exploring autoencoder-based error-bounded compression for scientific data. In *2021 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 294–306. IEEE, 2021.

[65] Jinyang Liu, Sheng Di, Kai Zhao, Xin Liang, Zizhong Chen, and Franck 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)*, pages 892–906. IEEE Computer Society, 2022.

[66] Jinyang Liu, Sheng Di, Kai Zhao, Xin Liang, Zizhong Chen, and Franck Cappello. Faz: A flexible auto-tuned modular error-bounded compression framework for scientific data. In *Proceedings of the 37th International Conference on Supercomputing*, pages 1–13, 2023.

[67] Jinyang Liu, Sheng Di, Kai Zhao, Xin Liang, Sian Jin, Zizhe Jian, Jiajun Huang, Shixun Wu, Zizhong Chen, and Franck Cappello. High-performance effective scientific error-bounded lossy compression with auto-tuned multi-component interpolation. *Proceedings of the ACM on Management of Data*, 2(1):1–27, 2024.

[68] Tong Liu, Jinzhen Wang, Qing Liu, Shakeel Alibhai, Tao Lu, and Xubin He. High-ratio lossy compression: Exploring the autoencoder to compress scientific data. *IEEE Transactions on Big Data*, 2021.

[69] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[70] Tao Lu, Qing Liu, Xubin He, Huizhang Luo, Eric Suchyta, Jong Choi, Norbert Podhorszki, Scott Klasky, Mathew Wolf, Tong Liu, et al. Understanding and modeling lossy compression schemes on hpc scientific data. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 348–357. IEEE, 2018.

[71] Yuzhe Lu, Kairong Jiang, Joshua A Levine, and Matthew Berger. Compressive neural representations of volumetric scalar fields. In *Computer Graphics Forum*, volume 40, pages 135–146. Wiley Online Library, 2021.

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

[73] William A Pearlman, Asad Islam, Nithin Nagaraj, and Amir Said. Efficient, low-complexity image coding with a set-partitioning embedded block coder. *IEEE transactions on circuits and systems for video technology*, 14(11):1219–1235, 2004.

[74] Tuomas Pelkonen et al. Gorilla: A fast, scalable, in-memory time series database. *Proc. VLDB Endow.*, 8(12):1816–1827, August 2015.

[75] Sanjith S., R. Ganesan, and Rimal Isaac. Experimental analysis of compacted satellite image quality using different compression methods. *Advanced Science*, 7, 03 2015.

[76] Naoto Sasaki, Kento Sato, Toshio Endo, and Satoshi Matsuoka. Exploration of lossy compression for application-level checkpoint/restart. In *IPDPS 2015*, pages 914–922, 2015.

[77] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[78] Tjerk P Straatsma, Katerina B Antypas, and Timothy J Williams. *Exascale scientific applications: Scalability and performance portability*. CRC Press, 2017.

[79] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012.

[80] Wanjie Sun and Zhenzhong Chen. Learned image downscaling for upscaling using content adaptive resampler. *IEEE Transactions on Image Processing*, 29:4027–4040, 2020.

[81] D. Tao, S. Di, X. Liang, Z. Chen, and F. Cappello. Fixed-psnr lossy compression for scientific data. In *2018 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 314–318, 2018.

[82] Dingwen Tao, Sheng Di, Zizhong Chen, and Franck 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*, pages 1129–1139. IEEE, 2017.

[83] Dingwen Tao, Sheng Di, Hanqi Guo, Zizhong Chen, and Franck Cappello. Z-checker: A framework for assessing lossy compression of scientific data. *The International Journal of High Performance Computing Applications*, 33(2):285–303, 2019.

[84] Dingwen Tao, Sheng Di, Xin Liang, Zizhong Chen, and Franck Cappello. Optimizing lossy compression rate-distortion from automatic online selection between sz and zfp. *IEEE Transactions on Parallel and Distributed Systems*, 30(8):1857–1871, 2019.

[85] David S Taubman, Michael W Marcellin, and Majid Rabbani. Jpeg2000: Image compression fundamentals, standards and practice. *Journal of Electronic Imaging*, 11(2):286–287, 2002.

[86] Jiannan Tian, Sheng Di, Xiaodong Yu, Cody Rivera, Kai Zhao, Sian Jin, Yunhe Feng, Xin Liang, Dingwen Tao, and Franck Cappello. cusz (x): Optimizing error-bounded lossy compression for scientific data on gpus. *CoRR*, 2021.

[87] Jiannan Tian et al. 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*, PACT '20, page 3–15, 2020.

[88] Robert Underwood, Jon C Calhoun, Sheng Di, Amy Apon, and Franck Cappello. Optzconfig: Efficient parallel optimization of lossy compression configuration. *IEEE Transactions on Parallel and Distributed Systems*, 2022.

[89] Robert Underwood, Sheng Di, Jon C Calhoun, and Franck Cappello. Fraz: A generic high-fidelity fixed-ratio lossy compression framework for scientific floating-point data. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 567–577. IEEE, 2020.

[90] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[91] Lipeng Wan, Kshitij V. Mehta, Scott A. Klasky, Matthew D. Wolf, H Y. Wang, W H. Wang, J C. Li, and Zhihong Lin. Data management challenges of exascale scientific simulations: A case study with the gyrokinetic toroidal code and adios. 7 2019.

[92] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for

dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 568–578, 2021.

[93] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[94] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22–31, 2021.

[95] Xiaodong Yu, Sheng Di, Kai Zhao, jiannan Tian, Dingwen Tao, Xin Liang, and Franck Cappello. SZx: an ultra-fast error-bounded lossy compressor for scientific datasets. *arXiv preprint arXiv:2201.13020*, 2022.

[96] Xinyang Yu et al. Two-level data compression using machine learning in time series database. In *36th IEEE International Conference on Data Engineering*, pages 1333–1344, 2020.

[97] Boyuan Zhang, Jiannan Tian, Sheng Di, Xiaodong Yu, Yunhe Feng, Xin Liang, Dingwen Tao, and Franck Cappello. Fz-gpu: A fast and high-ratio lossy compressor for scientific computing applications on gpus. *arXiv preprint arXiv:2304.12557*, 2023.

[98] Dongxiang Zhang, Mengting Ding, Dingyu Yang, Yi Liu, Ju Fan, and Heng Tao Shen. Trajectory simplification: an experimental study and quality analysis. *Proceedings of the VLDB Endowment*, 11(9):934–946, 2018.

[99] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 286–301, 2018.

[100] Kai Zhao, Sheng Di, Perez Danny, Zizhong Chen, and Franck Cappello. MDZ: An efficient error-bounded lossy compressor for molecular dynamics simulations. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, 2022.

[101] Kai Zhao, Sheng Di, Maxim Dmitriev, Thierry-Laurent D. Tonellot, Zizhong Chen, and Franck Cappello. Optimizing error-bounded lossy compression for scientific data by dynamic spline interpolation. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 1643–1654, 2021.

[102] Kai Zhao, Sheng Di, Xin Lian, Sihuan Li, Dingwen Tao, Julie Bessac, Zizhong Chen, and Franck Cappello. SDRBench: Scientific data reduction benchmark for lossy compressors. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 2716–2724, 2020.

[103] Kai Zhao, Sheng Di, Xin Liang, Sihuan Li, Dingwen Tao, Zizhong Chen, and Franck 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, HPDC '20, page 89–100, New York, NY, USA, 2020. Association for Computing Machinery.