Understanding and Estimating Error Propagation in Neural Networks for Scientific Data Analysis

Weiming He
New Jersey Institute of Tech.
wh249@njit.edu

Qi Chen
Temple University
qi.chen0004@temple.edu

Qian Gong
Oak Ridge National Laboratory
gongq@ornl.gov

Jing Li
New Jersey Institute of Tech.
jingli@njit.edu

Qing Liu
New Jersey Institute of Tech.
qliu@njit.edu

Norbert Podhorszki
Oak Ridge National Laboratory
pnorbert@ornl.gov

Scott Klasky
Oak Ridge National Laboratory
klasky@ornl.gov

Kisung Jung
Pukyong National University
kjung@pknu.ac.kr

Cristian Lacey
Sandia National Laboratory
celacey@sandia.gov

Jackie Chen
Sandia National Laboratory
jhchen@sandia.gov

Hongjian Zhu
X-Byte Research
hzhu@xbyteresearch.com

***Abstract*—Neural networks are increasingly integrated into scientific discovery, where input data reduction and model quantization play a key role in accelerating inference. However, understanding and mitigating the impact of these techniques on output error is critical for ensuring reliable results, particularly in tasks demanding high numerical precision. This paper introduces a comprehensive framework for optimizing neural network inference in scientific computing by combining data reduction and model quantization while maintaining error-controlled outcomes. We develop theoretical analyses to bound error propagation under these techniques and propose a framework that balances computational performance with error constraints. Evaluation on real-world learning-based combustion simulations and satellite image classification shows that our derived error bounds accurately predict observed errors while enabling significant computational speedup under our framework. This work highlights the potential for further leveraging advancements in modern lossy compression algorithms and hardware accelerators that support lower-precision formats.**

I. Introduction

Simulation-based scientific discovery has entered a new era where domain scientists are no longer limited to numerically solving the governing equations as the only path forward. Over the past decade, researchers have been increasingly leveraging new capabilities from machine learning and artificial intelligence (AI) to accelerate the path to knowledge. For example, in combustion simulations conducted at Sandia National Laboratory, calculating the chemical reaction rate—a crucial quantity in the Navier-Stokes equations—is expensive and time consuming due to the large number of species involved in a single reaction, each requiring individual solutions for its reaction rate using a resource-intensive in-house Fortran combustion solver at Sandia National Laboratory. By contrast, using a simple pre-trained neural network with two hidden layers with 50 neurons, combustion scientists can now accurately compute reaction rates for the 9-species hydrogen mechanism at significantly reduced costs [1]. This fundamentally shifts the computational research from compute-driven to a mixed paradigm driven by both compute and AI.

As neural networks become increasingly mature and prevalent in scientific discovery, it is crucial to understand the impact of using neural networks on the outcomes of knowledge discovery, with regard to the time to solution as well as the potential deviation from the ground truth. While this paper does not aim to broadly investigate the uncertainty of neural networks itself, our goal in this work is to understand the impact of model and data uncertainty as a result of employing data reduction, which is done in the following two steps during inference: 1) compressing the input data to the neural network to reduce the data retrieval overhead from persistent storage and 2) quantizing the network weights into fewer levels to reduce memory footprint and bandwidth consumption.

In particular, the compression of input data is motivated by the I/O bottleneck on high-performance computing (HPC) systems where the gap between compute and storage continues to widen [2], [3]. Consequently, large-scale simulations often employ some form of data reduction (e.g., spatiotemporal resolution reduction, compression, model reduction, etc. [4], [5]) to lower the amount of data transferred to/from the storage systems, while controlling the error for reduction to ensure acceptable outcomes. As scientific data analytics moves further towards AI, it is critical to understand and control the error propagation in neural networks so that important physics are not discarded during the training and inference process.

However, existing error-bounding methods have significant limitations when applied to the complex pipelines for modern scientific discovery. For instance, traditional error-bounded

This manuscript has been authored in part by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a non-exclusive, paid up, irrevocable, world-wide license to publish or reproduce the published form of the manuscript, or allow others to do so, for U.S. Government purposes. The DOE will provide public access to these results in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

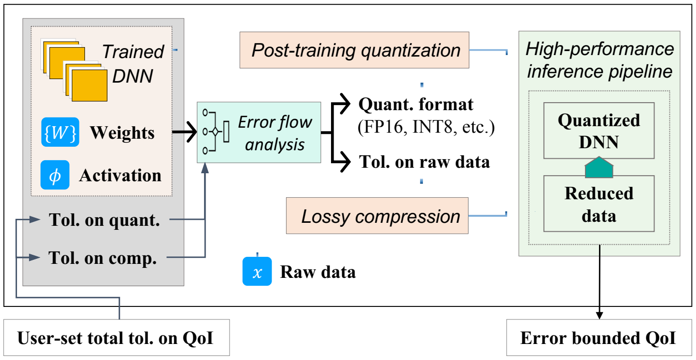

Fig. 1. Proposed framework. Starting with a trained deep neural network and a user-defined error tolerance on the quantity of interest (Tol. on QoI), our approach involves controlling two sub-components of the tolerance: quantization and compression tolerances. These tolerances are then fed into the error flow analysis, which generates an optimal combination of quantization and compression configurations. This enables the construction of an inference pipeline that effectively integrates lossy data reduction and post-training quantization, ensuring efficient and error-bounded inference for input data.

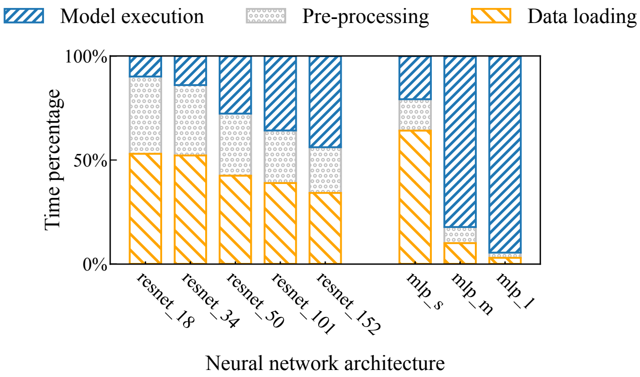

Fig. 2. Percentage of time spent on inference for models including standard ResNet architectures with varying depths adapted for 10-class classification, and MLP models of different sizes. The suffix of ResNet denotes the number of layers in the network. For MLP, *mlp_s* represents a model with a low FLOPS of 0.5M, *mlp_m* represents a model with a medium FLOPS of 4.2M, and *mlp_l* represents a model of high FLOPS of 33.7M.

compressors, such as SZ [6] and ZFP [7], were designed primarily for writing or reading data from disks and do not account for the compounded effect of errors introduced at multiple stages of data processing and inference. As such, they cannot ensure that the accumulated error remains within acceptable bounds at the output of the inference pipeline.

Meanwhile, applying post-training network quantization—a technique that reduces numerical precision (e.g., from FP32 to FP16) for inference computations—is driven by increased computational speed and enhanced memory efficiency. For instance, FP16 operations on NVIDIA GPUs achieve up to 8× arithmetic throughput and a 2× reduction in memory bandwidth compared to FP32 operations, resulting in a combined 16× improvement in peak performance [8]. Additionally, emerging numerical formats, such as TensorFloat-32 (TF32) and Brain Floating Point Format (BF16), offer balances between computational speed and numerical accuracy [9], [10], providing more flexible options for quantization. Advanced quantization strategies that apply block-wise, column-wise, or row-wise quantization to weight matrices can offer tighter quantization and reduced accuracy loss compared to uniform per-layer quantization. By grouping subsets of weights and assigning shared quantization parameters (e.g., scaling factors) within each group, these methods capture the local range of weights more precisely, thereby mitigating quantization error. These advancements provide significant performance improvements compared to using single precision alone.

Alongside compression and optimization efforts, various software tools and frameworks enhance deep learning inference through specialized techniques. The TensorRT SDK accelerates execution by fusing kernels, thereby improving GPU utilization and inference efficiency [11]. PyTorch's `torch.compile` [12] optimizes computation graphs to streamline execution and boost inference speed. The RAPIDS software stack efficiently manages GPU-based data pipelines, enabling rapid preprocessing and transfer. Smol [13] addresses preprocessing bottlenecks to improve end-to-end DNN inference. Although these tools significantly reduce neural network runtime, they lack mechanisms to analytically bound or evaluate the additional error introduced by combining lossy compression and quantization—an essential capability for scientific workflows.

In this paper, we aim to develop theoretical analyses to bound the error propagation of neural networks, when the input data and network weights are reduced. Enabled by these analyses, we propose a framework illustrated in Fig. 1 that provides recommendations on the optimized combination of data reduction and network quantization, given a pre-trained deep neural network and task-specific parameters, such as error tolerance for the quantity of interest (QoI) and the I/O-execution time ratio. Overall, this work provides new insights and tools for leveraging neural networks in scientific applications where efficient and reliable data processing is essential. The contributions of this work are as follows:

- To the best of our knowledge, this work is the first to develop mathematical theories for estimating the output error of neural networks when both the input data and network weights are reduced. By directly regulating the spectral properties of weight matrices, our method provides accurate error bound predictions that encompass the actual error.
- This paper investigates the potential performance benefits of modern compression algorithms and hardware accelerators in scientific tasks, with a particular focus on the synergistic effect of combining compression and quantization—a discovery that emerged from our experiments. We introduce a framework that leverages these advancements while ensuring error-controlled results.
- We evaluated two real-world combustion neural networks—designed to compute reaction rates and dissipation rates, respectively—along with various ResNet models to evaluate the effectiveness the error estimation. Our derived error bound successfully captures the actual achieved error in all cases while enabling considerable speedup.

II. Motivation

This work is motivated by the following observations:

Motivation 1: Scientific data analysis using neural networks spend large portions of time on data loading and model execution, where large amounts of floating-point data need to be ingested and processed. In Fig. 2, we break down the inference time of several common network modules into data loading, pre-processing, and model execution. The absolute processing time depends on the volume of data, which can reach terabyte (TB) or even petabyte (PB) scales in scientific computing. For example, high-fidelity climate simulations can generate up to 85 PB of data per day [14], while a single X-point Gyrokinetics Code (XGC) simulation in fusion research may produce hundreds of petabytes of particle state data [15]. Such enormous data volumes require hours or even days to read [14], let alone process. Our experiments are designed to replicate realistic HPC workloads where in situ analysis must be performed concurrently with simulations. In these scenarios, even state-of-the-art pipelines experience significant delays due to hardware limitations.

Motivation 2: For lossy data reduction, a key challenge is how to control the downstream error in the analysis pipeline, e.g., for visualization, computing new derived quantities. While this has been studied extensively for primary quantities, i.e., those that are reduced, the error propagation in a complex and deep data analysis pipeline has not been well studied. Recent work on MGARD was among the first to study and bound the error for linear quantities of interest (QoI). However, the mathematical operations in neural networks involve non-linear operations and the theories in MGARD cannot be directly applied. This work is motivated by the fact that when reduced data are fed into neural networks with reduced weights, new theories are needed to guarantee the error so that users can make informed decisions with regard to the accuracy of input data and network weights.

III. Method

This work leverages lossy compression and quantization to accelerate neural network inference. To avoid the prediction results to deviate from the ground truth significantly, we enforce an error tolerance that bounds the final QoI error.

A. Overview

In this section, we go through the notation setup and propose the formula for a safe error bound that limits the QoI error when both compression and quantization errors are present simultaneously. Consider an L-layer ResNet building block:

$$y = \mathcal{F}(x, \{W^{(l)}\}) + W_s x \quad \text{for } l = 1, \cdots, L-1 \tag{1}$$

where the function $\mathcal{F}(x, \{W^{(l)}\})$ represents the residual mapping to be learned; x is the input data; $W^{(l)}$ is the weight matrix for layer l; W_s is the shortcut matrix used for linearly projecting the skip connection. Notably, when the dimensions of x and $\mathcal{F}$ are equal, W_s is the identity matrix. Additionally, multi-layer perceptrons (MLPs) can be viewed as a special case of a residual block where W_s is a zero matrix. The recurrence relation inside the L-layer mapping is defined as:

$$\begin{cases} x = h^{(0)} \\ \mathcal{F} = z^{(L)} \end{cases} \text{and} \begin{cases} z^{(l)} = W^{(l)} h^{(l-1)} \\ h^{(l)} = \phi(z^{(l)}) \end{cases}$$

where ϕ is the nonlinear activation; $z^{(l)}$ denotes the pre-activation for layer l, which is the linear combination of inputs and weights before applying the activation function, and $h^{(l)}$ denotes the post-activation for layer l. We use n_l to denote the number of neurons of the l-th layer. Since the subsequent derivation does not depend on bias terms, we have omitted them to streamline the notation.

We aim to bound the error that propagates through each layer of the network, introduced by both lossy reduction and quantization post-training. In this paper, we focus on weight-only quantization using uniform affine transformation with max calibration [8], a.k.a. affine quantization. While we acknowledge that more advanced strategies, such as block-wise, column-wise, or row-wise quantization, can yield tighter memory footprints and potentially higher accuracy, these approaches are more complex to implement and optimize, and therefore do not align with our primary goal of maximizing throughput [16]. We intend to explore these alternative quantization techniques in future work. Let Δx represent the difference between the original data and its reconstruction. Similarly, Δz, Δh and Δy denote the deviations from z, h, and y, respectively. We focus on the discussion of the L2 norm, denoted by $\|\cdot\|_2$. The results can be extended to the L-infinity norm $\|\cdot\|_\infty$, given that:

$$\frac{1}{\sqrt{n}}\|\cdot\|_2 \leq \|\cdot\|_\infty \leq \|\cdot\|_2.$$

Through one linear projection $z^{(l)} = W^{(l)} h^{(l-1)}$, the L2 norm of the input erro r vector $\|\Delta h^{(l-1)}\|_2$ is amplified by a finite factor, bounded by the spectral norm $\sigma_W^{(l)}$ of the weight matrix $W^{(l)}$ by definition, which is defined as:

$$\sigma_W^{(l)} := \sup_{\Delta h \neq 0} \frac{\|W^{(l)} \Delta h^{(l-1)}\|_2}{\|\Delta h^{(l-1)}\|_2}. \tag{2}$$

Note that the value of $\sigma_W^{(l)}$ is the largest singular value of the weight matrix $W^{(l)}$, which remain fixed after training. The power iteration method [17] is commonly used to approximate this value efficiently.

Suppose the activation functions applied after each linear layer have an upper limit on their first derivative, we have:

$$C = \sup_z \frac{d}{dz}\phi(z) \implies \|\Delta h^{(l)}\|_2 \leq C \|\Delta z^{(l)}\|_2.$$

This assumption does not affect the generality. Most commonly used activation functions in modern deep learning, such as ReLU and its variants, GeLU, Tanh, etc. have globally bounded derivatives. In rare cases where the derivative is not globally bounded (e.g., with polynomial activations), we perform a careful local analysis limited to the regions of the activation function where the data encounters. Because standard backpropagation requires bounded derivatives, a local bound is guaranteed in these regions, and establishing it suffices to ensure compatibility with our theoretical framework. Note that our framework does not support discontinuous activation functions, which are rarely used in mainstream deep learning architectures. For common activations including Tanh, ReLU and LeakyReLU, which are covered in this paper, we have $C = 1$. Hence, we omit the constant C in the following discussion for simplicity.

With both lossy reduction and post training affine quantization applied, the final output error $\|\Delta y\|_2$ of the building block in Equation (1) is bounded by the following equation:

$$\begin{aligned} \|\Delta y\|_2 \leq & (\sigma_s + \prod_{l=1}^{L} \sigma_W^{(l)}) \times \|\Delta x\|_2 + \\ & \sum_{l=1}^{L} \Big(\prod_{i=1}^{l-1} (\sigma_W^{(i)} + \frac{q_i \sqrt{\min(n_{i-1}, n_i)}}{\sqrt{3}}) \\ & \times \prod_{j=l+1}^{L} \sigma_W^{(j)} \times \frac{q_l \sqrt{n_0 n_l}}{2\sqrt{3}} \Big) \end{aligned} \tag{3}$$

Here, σ_s is the spectral norm of the shortcut matrix W_s. In the case of a MLP, σ_s is defined to be zero. $q_l = q\left(W^{(l)}\right)$ denotes the quantization step size for each layer. The formulas to calculate the step sizes for several widely used numerical formats are shown in TABLE I.

TABLE I
AVERAGE QUANTIZATION STEP SIZE q FOR COMMON NUMERICAL FORMATS.

FORMAT	STEP SIZE
TF32	$q\left(W\right) = 2^{-10} \times \sqrt{2^{\,2\times\lfloor\log_2 \lvert W_{ij}\rvert\rfloor}}$
FP16	$q\left(W\right) = 2^{-10} \times \sqrt{2^{\,2\times\max(-14,\lfloor\log_2 \lvert W_{ij}\rvert\rfloor)}}$
BF16	$q\left(W\right) = 2^{-7} \times \sqrt{2^{\,2\times\lfloor\log_2 \lvert W_{ij}\rvert\rfloor}}$
INT8	$q\left(W\right) = 2^{-8} \times \Big(\max(W_{ij}) - \min(W_{ij})\Big)$

Inequality (3) provides a practical tool for determining an appropriate combination of compression tolerance and quantization precision in order to meet a user-specified QoI tolerance. Notably, for a given deep network, the upper bound of disturbance in its output depends solely on the compression error $\|\Delta x\|_2$ as well as the numerical format employed in quantization, making it readily applicable.

B. Error Flow Analysis

We have two components of errors: compression and quantization errors. Despite the composed manner these two errors propagate through the network, they contribute to the upper bound of the total error additively. Consider the network configuration space (x, W), representing all possible values of the input features fed into the network and all possible values of the weights in the network. We denote noisy or perturbed versions of variables due to either lossy compression or quantization by the tilde notation $\tilde{\cdot}$. Specifically, $\tilde{x}$ represents the reconstructed version of input x, $\widetilde{W}$ represents the quantized version of weights W, $\tilde{h}$ and $\tilde{z}$ denote the noisy version of hidden variables h and z respectively, affected due to the application of reduction and quantization. We compute the total error Δy on the output via a path integral:

$$\Delta y = \int_{\gamma} \nabla_{\gamma} y(x, W) ds. \tag{4}$$

where $\gamma(t) = (x(t), W(t))$ for $t \in [0, 1]$ is a path within the configuration space and that $\gamma(0) = (x, W)$; $\gamma(1) = \left(\tilde{x}, \widetilde{W}\right)$; $ds = \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dW}{dt}\right)^2}$; dt is the arc length element. Two paths to compute the integral in Equation (4) are:

$$\begin{tikzcd} & (\tilde{x},\ W) \arrow[dr] & \\ (x,\ W) \arrow[ur] \arrow[dr] & & (\tilde{x},\ \widetilde{W}) \\ & (x,\ \widetilde{W}) \arrow[ur] & \end{tikzcd}$$

During fast inference, we typically only have access to the noisy variables $\tilde{x}$ and $\widetilde{W}$; we have no access to the full precision data x, but we can store and access the original weight matrices $\{W^{(l)}\}$. Hence, we follow the path where we first vary x and then W:

$$\Delta y = \int_{x}^{\tilde{x}} \nabla_x\, y(x)|_{W} \cdot dx + \int_{W}^{\widetilde{W}} \nabla_W\, y(W)|_{\tilde{x}} \cdot dW.$$

The first term represents the compression error, introduces by the disturbance of the input x from its original value to the noisy reconstructed version, while keeping the network weights unchanged; the second term represents the quantization error, which occurs when the weights W are transitioned from their original values to the quantized versions, while the inputs are kept the same.

By applying the chain rule and our assumption in Section III-A that the activation functions have their first derivatives bounded by an upper limit of 1, we obtain:

$$\nabla_x\, y(x)|_{W} \leq W_s + W^{(L-1)} \cdot W^{(L-2)} \cdots W^{(1)}.$$

Thus, we establish an upper bound for the compression error:

$$\| \int_{x}^{\tilde{x}} \nabla_x\, y(x)|_{W} \cdot dx \,\|_2 \leq (\sigma_s + \prod_{l=1}^{L} \sigma_W^{(l)}) \times \|\Delta x\|_2. \tag{5}$$

Quantization involves scaling and rounding off weight matrices and activations. The error introduced by activation quantization can be addressed similarly to compression error by applying Equation (5), while excluding all layers preceding the affected activation. Due to affine quantization, the deviations of individual weights $\Delta W^{(l)} = \widetilde{W}^{(l)} - W^{(l)}$ are independently distributed with a uniform distribution centered at zero. Since the components of $\tilde{h}^{(l-1)}$ are finite, the following inner product satisfies Lindeberg's condition [18]. By the Central Limit Theorem, as n_{l-1} goes to infinity, the inner product converges in distribution to a normal random variable:

$$\sum_{j} \Delta W_{ij}^{(l)} \tilde{h}_j^{(l-1)} \xrightarrow{d} \mathcal{N}(0, s_l^2), \quad \forall i,$$

where $s_l^2 = \frac{q_l^2}{12} \cdot \|\tilde{h}^{(l-1)}\|_2^2$, and q_l is the average quantization step size of layer l, see TABLE I. By the Law of Large Numbers and concentration inequalities, when n_l is sufficiently large, which is generally true in scientific neural networks, the norm $\|\Delta W^{(l)} \cdot \tilde{h}^{(l-1)}\|_2$ concentrates around its mean value:

$$\|\Delta W^{(l)} \cdot \tilde{h}^{(l-1)}\|_2 \to s_l \cdot \sqrt{n_l} = \frac{q_l \sqrt{n_l}}{2\sqrt{3}} \cdot \|\tilde{h}^{(l-1)}\|_2.$$

To bound $\|\tilde{h}^{(l-1)}\|_2$ without specific input values, we assume the inputs are normalized within the range $[-1, 1]$ during preprocessing. Thus, we have

$$\|\tilde{h}^{(l-1)}\|_2 \leq \prod_{i=1}^{l-1} \sigma_{\widetilde{W}}^{(i)} \cdot \sqrt{n_0}.$$

Recursively applying the above inequalities through the layers, we obtain:

$$\begin{aligned} \| \int_{W}^{\widetilde{W}} \nabla_W y(W)|_{\tilde{x}} \cdot dW \|_2 \leq \prod_{i=1}^{L-1} \sigma_{\widetilde{W}}^{(i)} \cdot \frac{q_L \sqrt{n_0 n_L}}{2\sqrt{3}} \\ + \sigma_W^{(L)} \Bigg(\sigma_W^{(L-1)} \Big(\sigma_W^{(L-2)} \cdots \big(\sigma_W^{(1)} \|x - \tilde{x}\|_2 \\ + \frac{q_1 \sqrt{n_0 n_1}}{2\sqrt{3}} \big) \cdots + \prod_{i=1}^{L-3} \sigma_{\widetilde{W}}^{(i)} \cdot \frac{q_{L-2} \sqrt{n_0 n_{L-2}}}{2\sqrt{3}} \Big) \\ + \prod_{i=1}^{L-2} \sigma_{\widetilde{W}}^{(i)} \cdot \frac{q_{L-1} \sqrt{n_0 n_{L-1}}}{2\sqrt{3}} \Bigg). \end{aligned}$$

It's also helpful to predict the quantization error bound before we actually quantize the network. Thus, we can eliminate all of $\sigma_{\widetilde{W}}$ in above equation:

$$\sigma_{\widetilde{W}}^{(l)} \leq \sigma_W^{(l)} + \frac{q_l \sqrt{\min(n_{l-1}, n_l)}}{\sqrt{3}} \quad \text{for } l = 1, \cdots, L-1.$$

Combining compression error and quantization error, we arrive at the Inequality (3). This final expression demonstrates how the compression error and quantization error accumulates across layers and provides a clear guideline for selecting quantization parameters to meet the desired QoI tolerance.

C. Parameterized Spectral Normalization

The QoI error depends on the neural network's learned weights, which are shaped by the training data. In H2Combustion, the reaction rates of different species exhibit relatively low sensitivity to individual input variations (i.e., mass fraction changes). In contrast, BorghesiFlame shows high sensitivity to input perturbations. For the EuroSAT classification task, we consider the final feature map as the quantity of interest, as it is essential not only for classification but also for downstream tasks. Its sensitivity to variations in input pixel values remains relatively low. In general, scientists can leverage their empirical knowledge of the data to determine appropriate compression tolerance levels.

$$W_{\text{PSN}} = \frac{W}{\sigma_W} \times \alpha + \beta \tag{6}$$

We introduce a pair of learnable parameters, weight α and bias β, that scale and shift each neuron's normalized value, and the new spectral norm after normalization is $\sigma_{W_{\text{PSN}}} = \alpha$. This parameterized spectral normalization is based on the previous version of spectral normalization used in GAN training [19]. Although the spectral normalization offers significant benefits such as stabilized training and improved generalization, its adoption beyond GANs remains limited due to its higher complexity than the alternative methods like the batch normalization [20]. In our approach, we validate the application of spectral normalization to enhance inference performance while maintaining controlled error rates.

The squared sum of the spectral norm for each layer is added as a penalty to the loss function to constrain the value of the new spectral norm. This modification ensures that the network can represent any underlying function in scientific tasks, where the Lipschitz constants are generally unknown. Lipschitz constant of a differentiable function is the maximum norm of the derivative across the domain, and it is closely related to the spectral norms of each layer in a neural network. If applying the standard spectral normalization, which is designed to constrain the spectral norm of the target layer to be exactly 1, to control the spectral norms of all layers, the resulting network can only fit functions with the Lipschitz constant no more than 1. In comparison, our parameterized spectral normalization allows for the necessary expressive ability of scientific tasks. Furthermore, the additional parameters make it possible to apply the parameterized spectral normalization in the middle way during training or fine-tuning in the end.

IV. Experiment results

This section presents the experimental evaluation.

A. Experimental Setup

We evaluate three scientific data analytics: chemical source term prediction in turbulent hydrogen combustion [1], dissipation rate profiles within the multi-modal combustion configuration called Borghesi flame [21], and land use and land cover classification using EuroSAT satellite imagery [22]. These experiments encompass a wide range of neural network architectures, from small-scale models to larger configurations, including both fully connected networks and ResNets. We incorporate commonly used activation functions, including Tanh, ReLU, and PReLU, and different optimization algorithms, such as Stochastic Gradient Descent (SGD) and Adam. This allows us to demonstrate the applicability of our methods across diverse applications and network configurations.

Prior to training, we apply our proposed parameterized spectral normalization technique to the networks (Section III-C) and subsequently train them in full precision. Upon completion, the model weights are stored for error bound prediction using Equation (3). These experiments were conducted on NVIDIA V100 GPUs on Summit [23] and AMD MI250X GPUs on Frontier [24], both with a Lustre filesystem. The input data are normalized prior to training and inference to ensure consistency, and we use three widely used scientific

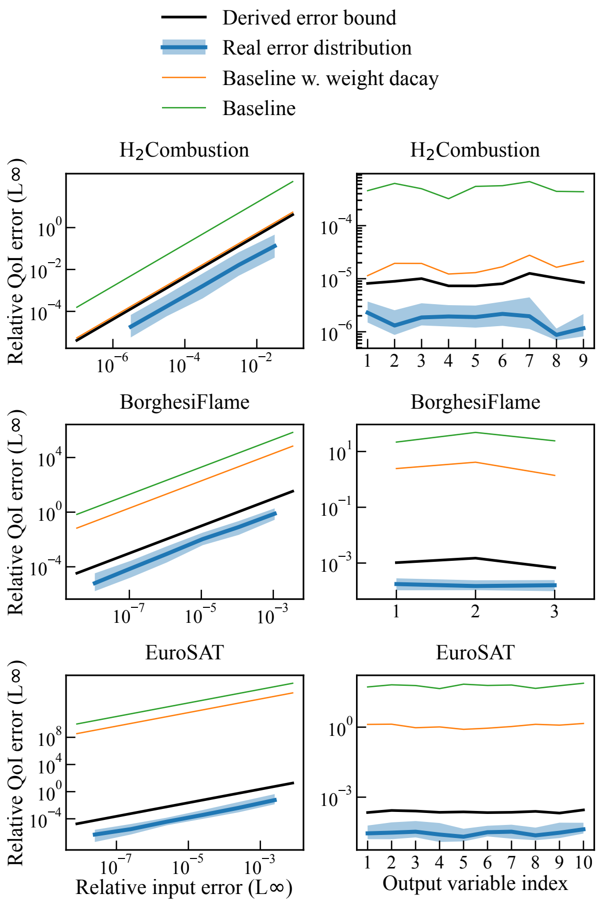

Fig. 3. Error bound prediction vs. the actual error distribution in L-infinity norm. The error distribution is sampled by three compressors and five independent batches of input data. The baseline corresponds to the derived error bound without parameterized spectral normalization, while baseline w. weight decay uses weight decay in place of the proposed parameterized spectral normalization. Left figures show the total error across all output features, while right figures show the per-feature QoI error.

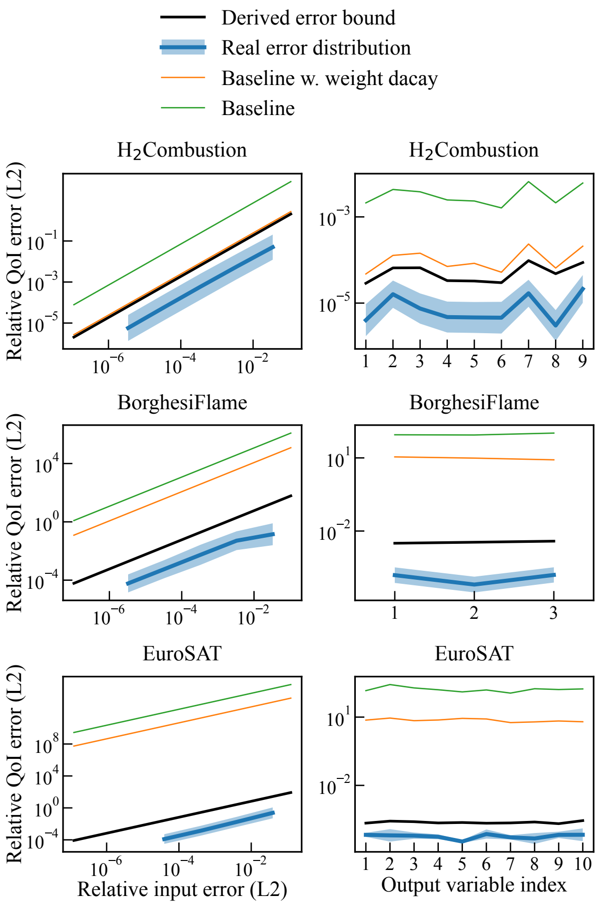

Fig. 4. Error bound prediction vs. the actual error distribution in L2 norm. The error distribution is sampled with three compressors and five independent batches of input data. The baseline corresponds to the derived error bound without parameterized spectral normalization, while baseline w. weight decay uses weight decay in place of the proposed parameterized spectral normalization. Left figures show the total error across all output features, while right figures show the per-feature QoI error.

compressors for input data reduction: ZFP [7], SZ [6], [25], and MGARD [26], [27]. ZFP uses a block-based transform for either fixed-rate or fixed-accuracy compression, making it popular for on-the-fly or in-memory operations. SZ relies on prediction and quantization to achieve high compression ratios, often used in large-scale HPC simulations like climate or fluid dynamics. MGARD employs a multigrid-based approach, making it well-suited for controlling user-defined error tolerances in structured multi-dimensional data. We ran additional experiments on the RTX 3080 Ti GPU, featuring the newer Ampere architecture with the support of TensorFloat32 (TF32) and bfloat16 (BF16). In what follows, we further detail the scientific data analytics we tested:

1) Hydrogen combustion: Turbulent combustion simulation is a critical area within computational fluid dynamics, requiring precise modeling of chemical reactions to accurately predict flame behavior and pollutant formation. We utilize a simplified nine-species mechanism of hydrogen (H_2) combustion [1] to manage computational complexity while capturing essential chemical dynamics. The dataset consists of mass fractions for each species, derived from high-fidelity combustion simulations conducted under various turbulent conditions. The simulations feature a single vortex structure positioned at the center, serving as the source of turbulence. A compact neural network model comprising two hidden layers is designed to receive the mass fractions of the nine species as inputs and produce their corresponding reaction rates as outputs. The model is trained using the standard SGD optimizer, facilitating efficient and stable convergence.

2) Borghesi flame: The Borghesi Flame is a Direct Numerical Simulation (DNS) database that models an auto-igniting turbulent n-dodecane jet flame [21]. It encompasses several stages representative of diesel ignition and combustion processes. We conduct a learning-based dissipation rate profiling on this dataset, which is a critical domain within combustion simulations. The input dataset comprises 13 distinct variables that reflect various aspects of the thermochemical states in combustion simulations. These inputs include mixture fraction gradients, progress variable gradients, and several other derived parameters that are crucial for accurately representing the

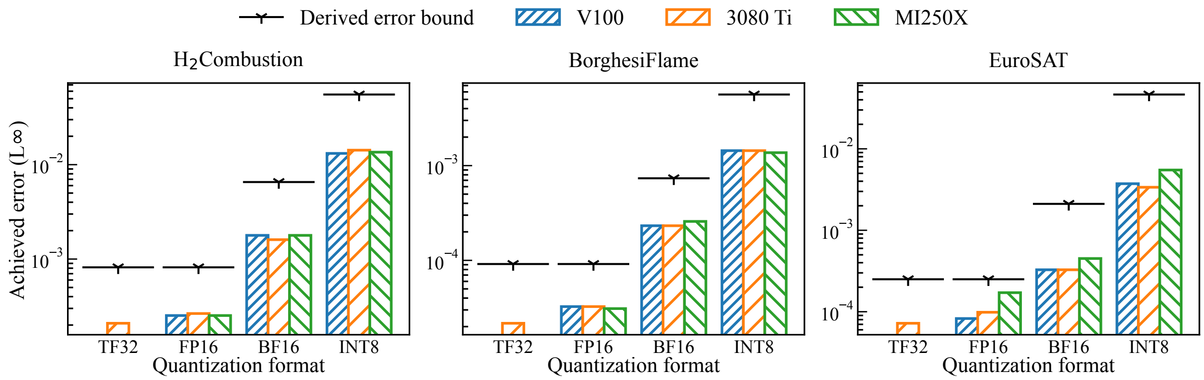

Fig. 5. Comparison of predicted error bounds and achieved relative QoI error in L-infinity Norm across different quantization formats. Results for tf32 and bf16 are available only for the RTX 3080 Ti GPU, as these formats are unsupported on the other GPUs tested. The TF32 format yields a nearly identical error bounds to the FP16 format, which is largely attributed to both formats having the same number of mantissa bits. Similarly, despite that both FP16 and BF16 utilizes a 16-bit size, the BF16 exhibits considerably higher error bounds due to having less mantissa bits.

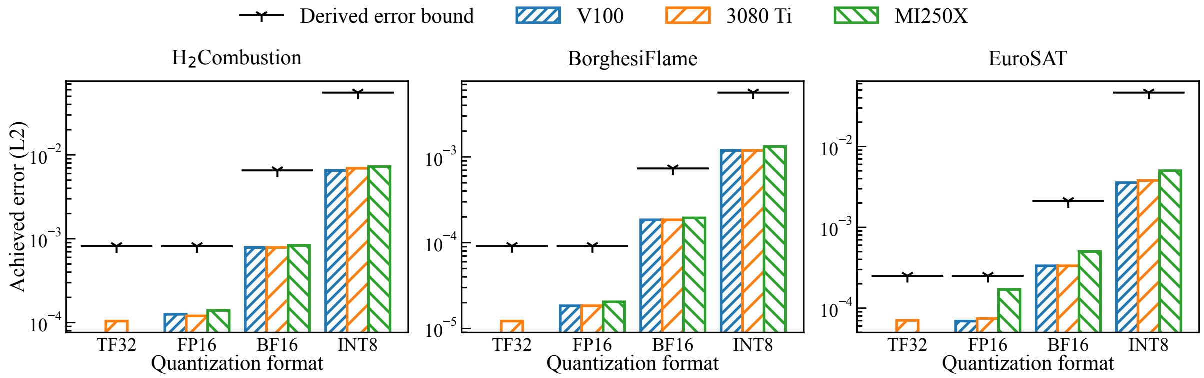

Fig. 6. Comparison of predicted error bounds and achieved relative QoI error in L2 Norm across different quantization formats. Results for tf32 and bf16 are available only for the RTX 3080 Ti GPU, as these formats are unsupported on the other GPUs tested. The TF32 format yields a nearly identical error bounds to the FP16 format, which is largely attributed to both formats having the same number of mantissa bits. Similarly, despite that both FP16 and BF16 utilizes a 16-bit size, the BF16 exhibits considerably higher error bounds due to having less mantissa bits.

state of the turbulent flame in coarse-grained simulations. We use an 8-hidden-layer MLP that outputs three specific filtered dissipation rates: the mixture fraction dissipation rate, the generalized progress variable dissipation rate, and the cross-dissipation rate, encapsulating the dominant scalar transport needed to model the combustion system. The training process utilizes the Adam optimizer, facilitating efficient convergence.

3) EuroSAT: Unlike general image processing applications, scientific tasks such as satellite image classification employ high-precision data and necessitate enhanced numerical accuracy. Specifically, we employ the EuroSAT dataset [22], which comprises multispectral, 16-bit samples derived from satellite observations across various spectral bands. This dataset offers a representation of the Earth's surface using European satellite imagery, designed for LULC classification across 10 distinct output classes. Prior to training, the images are resized to dimensions of 224×224 pixels and subjected to normalization to standardize the input data. We trained a ResNet18 model configured with 10 output features on the EuroSAT dataset, incorporating parameterized spectral normalization. The training process utilizes the standard SGD optimizer.

B. Validation of Error Estimation

In this section, we evaluate the accuracy of the proposed error estimation method and compare that with the achieved error observed in our experiments. To understand the impact on accuracy as a result of both lossy compression and quantization, we first study the compression and quantization error separately. In our study, our estimated error consistently bounds the actual error, with the discrepancy between the two being around one order of magnitude in all three tasks. All errors discussed in this section are relative errors by default.

1) Compression error: Fig. 3 and 4 illustrate the compression error, using the L-infinity and L2 norm, respectively. We calculate both the global QoI error, which represents the total error across all output features, and the per-feature QoI error, which measures the error for a single output feature. The distribution of the achieved QoI errors is compared to the derived error bound. For the global error, results are presented across a range of input error levels, while for the per-feature error, results are shown specifically for a relative input error of 10^{-5}. To assess the achieved QoI error, we apply three compression algorithms—ZFP, SZ, and MGARD—across five independently sampled batches of input data. We plot the

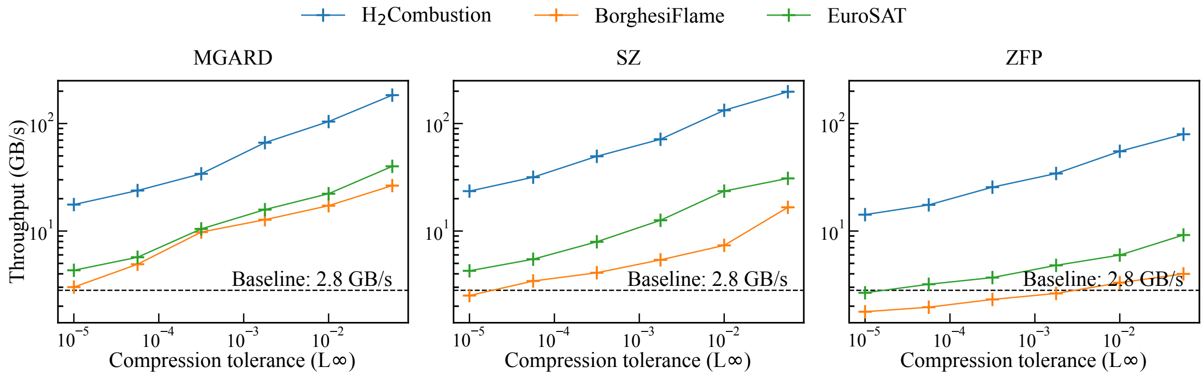

Fig. 7. I/O throughput versus user-specified tolerance on QoI, in L-infinity Norm, using different compression backends. At lower tolerance levels, MGARD and SZ may reduce I/O throughput due to the added decompression time.

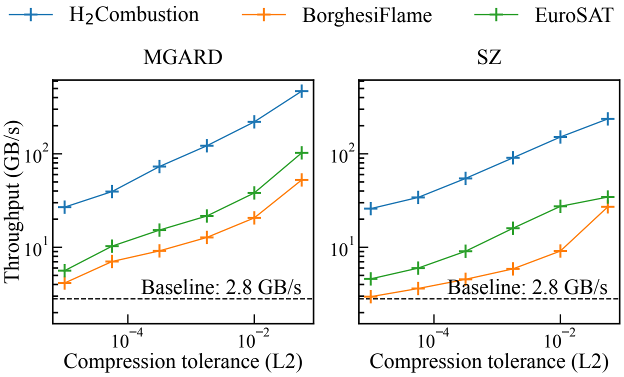

Fig. 8. I/O throughput versus user-specified tolerance on QoI, in L2 Norm, using different compression backends. At lower tolerance levels, MGARD and SZ may reduce I/O throughput due to the added decompression time. ZFP does not support an L2 norm tolerance.

geometric mean and the range of the achieved QoI errors against the actual input errors resulting from the compression backends. It is observed that, even in the worst-case scenarios, the achieved errors consistently remain below the derived error bounds, validating that our error bound is guaranteed.

Furthermore, the gap between the predicted and achieved errors is within one order of magnitude relative to the actual errors. This is significant because when the gap is within one order of magnitude, it indicates that the overall trend of the QoI error is effectively captured without introducing a full order-of-magnitude discrepancy. Achieving this is particularly challenging given the diversity of data and the inherent difficulties of exact error prediction in data reduction [28]. This shows that compression does not significantly distort essential scientific information in deep networks trained with parameterized spectral normalization. This robustness is achieved by directly constraining the spectral norm of each layer, which regulates error propagation across the network and enables precise error bound predictions.

2) Quantization error: Fig. 5 and 6 present the results of quantization error across several widely adopted quantization formats (TF32, FP16, BF16, INT8), with L-infinity and L2 norm, respectively. We quantize our models using the PyTorch Quantization APIs. To validate our proposed error bounds across different computational backends, each with their specific quantization implementations, we conducted experiments on three GPUs: NVIDIA V100, RTX 3080 Ti, and AMD MI250X. Notably, only the RTX 3080 Ti GPU natively supports TF32 and BF16. The NVIDIA V100 and AMD MI250X emulate BF16 computations. Despite the approximations made in deriving the quantization error bounds, the estimated error effectively bounds the observed errors in all cases.

For each QoI metric, the relative error generally increases as quantization precision decreases (e.g., moving from TF32 to INT8), reflecting the trade-off between precision and computational efficiency. For the TF32 format, we obtain error bounds nearly identical to those of the FP16 format, and the achieved errors do not differ significantly either. Such an outcome is largely attributed to both formats having the same number of mantissa bits. Although BF16 also utilizes a 16-bit size, it exhibits considerably higher error bounds and achieved errors compared to FP16 due to having fewer mantissa bits. This highlights the vital role of mantissa bits in maintaining accuracy during the inference phase of scientific models, in contrast to the training phase where the exponent is more critical. Specifically, during the inference of MLPs or CNNs, the dynamic range required for accurate computation is smaller. In training, however, gradients can be extremely small, making them susceptible to underflow. Offering more mantissa bits are essential during inference to ensure high numeric precision in resulting QoIs. INT8 quantization introduces a larger relative error, exceeding 10^{-2} in two tasks. Therefore, the application of this format requires careful consideration. Overall, among the quantization formats investigated, FP16 is the most cost-effective option for scientific inference, as it balances well between total bit size and mantissa bit allocation.

The sensitivity of QoI errors is influenced by the neural network's learned weights, which are shaped by the training data. In H2Combustion, the reaction rates of different species are relatively insensitive to individual input variations (i.e., changes in mass fractions): an input perturbation of 10^{-3} produces a 10^{-3} change in the QoI (measured in L2 norm). In contrast, BorghesiFlame exhibits higher sensitivity, with the

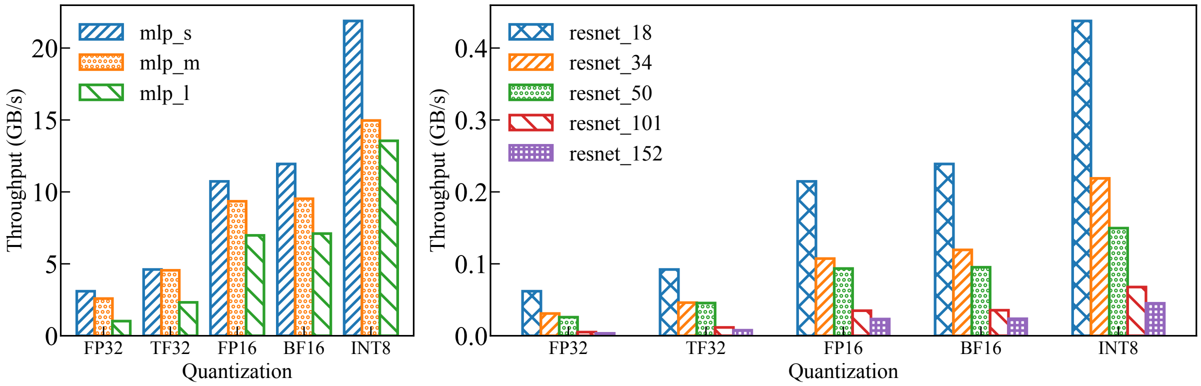

Fig. 9. Data ingestion throughput vs. Quantization formats. The models evaluated include standard ResNet architectures with varying depths, adapted for 10-class classification, and MLP models of different sizes. The suffix of ResNet denotes the number of layers in the network. For MLP, *mlp_s* represents a ResNet with a low FLOPS of 0.5M, *mlp_m* represents a ResNet with a medium FLOPS of 4.2M, and *mlp_l* represents a ResNet of high FLOPS of 33.7M.

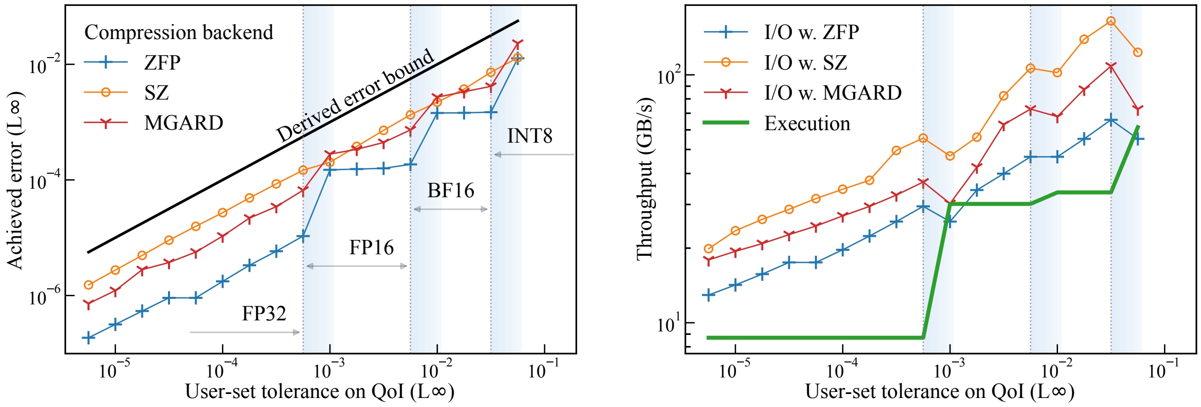

Fig. 10. Coordination of data reduction and quantization to enhance inference speed, prioritizing quantization. Left: Compression exploits the difference between quantization error and the user-specified tolerance. Right: Throughput of the I/O phase and Execution phase plotted against tolerance; the total throughput is determined by the slower of the two phases, with the Execution phase acting as the bottleneck in this example.

same input variation leading to a 10^{-2} change in the QoI. For the EuroSAT classification task, we observe a sensitivity falling between these two cases. Overall, scientists can leverage their empirical understanding of the data to establish appropriate compression tolerance levels.

C. Individual Performance Metrics

We evaluate the I/O performance gains achieved through lossy compression on three datasets, H_2Combustion, BorghesiFlame, and EuroSAT. Depending on the dataset and compression algorithm employed, we achieve up to a tenfold increase in I/O throughput at a QoI tolerance of 10^{-3}. Additionally, we assess the computation throughput gains resulting from quantization on a series of MLPs and ResNets of various sizes using the RTX 3080 Ti GPU, as it is the only GPU currently available to our team that natively supports the TF32 format. We achieve up to a 4.5-fold increase in computation throughput for FP16-quantized models compared to their non-quantized FP32 counterparts. These results demonstrate the effectiveness of our derived error bounds in guiding the optimization of data compression and model quantization, enabling a controlled trade-off between accuracy and performance.

Fig. 7 illustrates I/O throughput in relation to user-specified compression tolerance levels on QoI. The I/O throughput of compressed data is impacted by both the compression ratio achieved, as well as the decompression speed at fetching. At higher tolerance levels, each compression backend generally enhances I/O throughput beyond the baseline of 2.8 GB/s. However, at lower tolerance levels, the throughput for SZ and MGARD decreases, due to the additional decompression time required to meet stringent accuracy requirements. ZFP, in contrast, maintains relatively stable throughput across tolerance levels, in line with the previous report [29] that ZFP has a higher decompression speed compared with SZ and MGARD. This trend suggests that, while compression can significantly boost I/O throughput, the choice of compression backend and tolerance settings must be carefully considered to avoid bottlenecks in scenarios demanding high precision.

The model execution throughput versus computation precisions/quantization formats is shown in Fig. 9. The models evaluated include standard ResNet architectures with varying depths, adapted for 10-class classification, and MLP models of different sizes. In particular, the suffix of ResNet denotes the number of layers in the network. For MLP, *mlp_s* represents

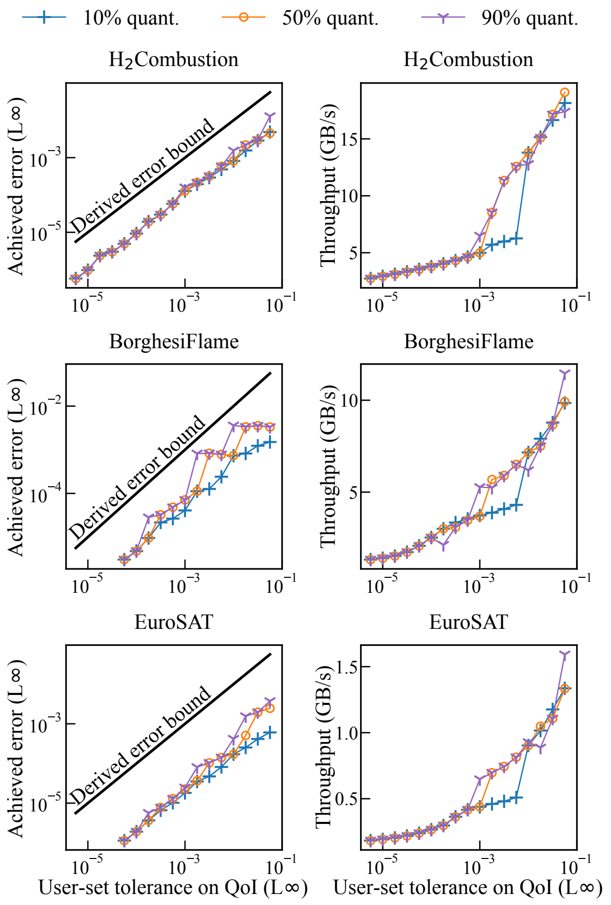

Fig. 11. Left: Predicted error bounds vs. user set tolerance in L-infinity. Right: Throughput vs. user set tolerance in L-infinity. Both results are obtained using MGARD as the compression backend.

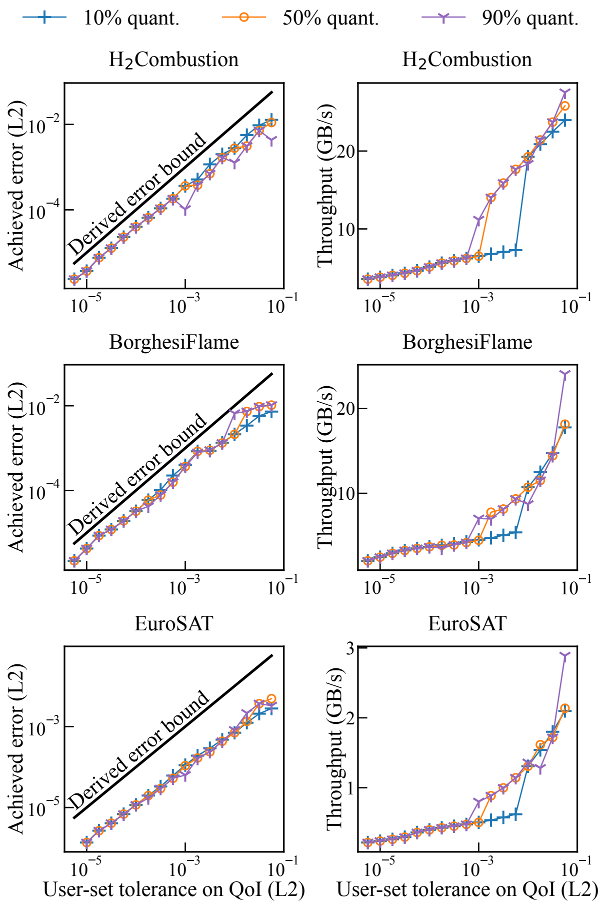

Fig. 12. Left: Predicted error bounds vs. user set tolerance in L-infinity. Right: Throughput vs. user set tolerance in L2. Both results are obtained using MGARD as the compression backend.

a ResNet with a low FLOPS of 0.5M, *mlp_m* represents a ResNet with a medium FLOPS of 4.2M, and *mlp_l* represents a ResNet of high FLOPS of 33.7M. The speedup via quantizing the models is up to 4.5x times of the original throughput, and it is essential in improving the overall performance when model execution is the bottleneck in the inference pipeline.

As discussed in the previous section, while the FP16 format offers a large speedup due to the smaller total bit size, it maintains a generally acceptable accuracy due to the proper balance of the number of mantissa bits, making it a good choice for quantization. Another format offering a large speedup is INT8, which, however, can introduce error exceeding user tolerances. TF32 and BF16 are not ideal choices in most cases in that they provide little speedup while introducing a large quantization error compared to the upstream formats.

D. Inference Pipeline Performance

The throughput of data I/O and model execution should be balanced to ensure that neither becomes a bottleneck. A balanced pipeline can handle increased loads more gracefully, allowing for better system scaling to meet higher demand. In this section, we underscore the importance of choosing suitable combination of compression and quantization tolerances to achieve the optimum performance improvement.

Given a total QoI tolerance set by a user, individual tolerances for compression and quantization will be allocated according to the discrepancy between the throughput of data I/O and model execution. It is notable that while the compression tolerance takes a continuous input, the quantization tolerance only has a few discrete valid values, attribute to the limited options of quantization format available. In our experiments, we generate a continuous quantization tolerance based on the derived error bound, times the configurable factor to control the proportion of total tolerance allocated to quantization. The best quantization format of which the predicted error bound does not exceed the allocated quantization tolerance is selected. Once quantization is decided, all unutilized tolerance are allocated for data reduction.

Fig. 10 illustrates the interaction between compression and quantization in the experiment of turbulent hydrogen combustion simulation, where quantization is applied whenever the tolerance required is encompassed by the total QoI tolerance. Since our derived error bounds for compression error and quantization error generally deviate from the corresponding achieved errors in different levels, the tightness of the error

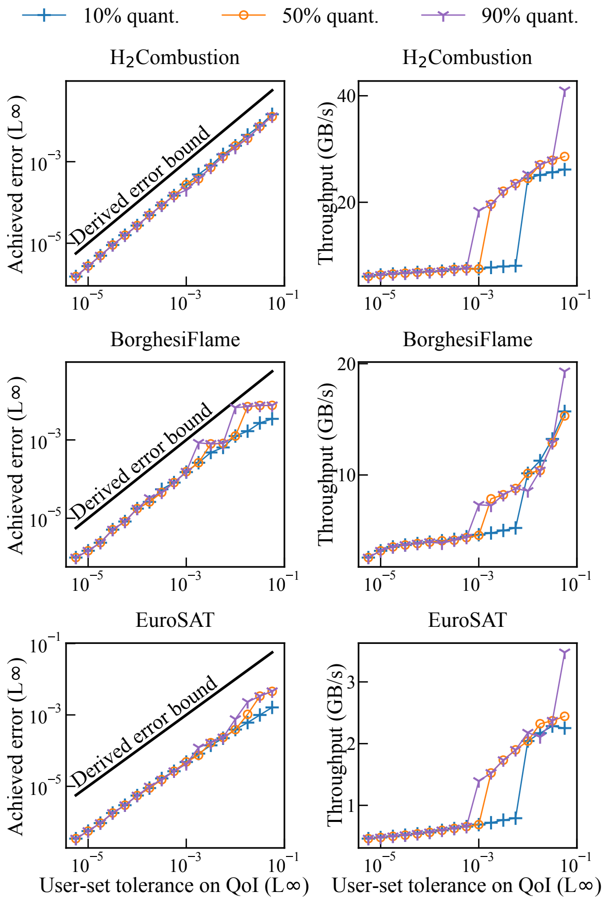

Fig. 13. Left: Predicted error bounds vs. user set tolerance in L-infinity. Right: Throughput vs. user set tolerance in L-infinity. Both results are obtained using SZ as the compression backend.

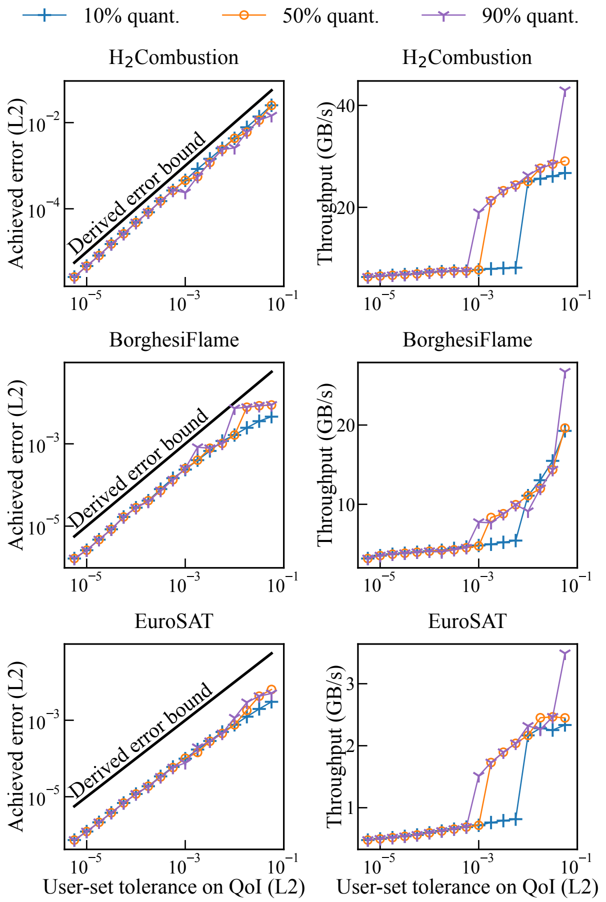

Fig. 14. Left: Predicted error bounds vs. user set tolerance in L-infinity. Right: Throughput vs. user set tolerance in L2. Both results are obtained using SZ as the compression backend.

bound will have a jump around the point where upstream quantization format is applied. In this specific experiment, model execution is a bottleneck, as shown in the right figure that the model execution throughput is consistently smaller than that of the I/O, even at the point where 100% of the total tolerance is allocated to quantization.

Figs. 11 - 15 enumerate the detailed results using MGARD, SZ and ZFP as the compression backend. Results of both L-infinity norm and L2 norm are listed, except for ZFP which does not support an L2 tolerance. We allocate different proportion of the total tolerance to the quantization, from 10% to 90%, to highlight the capability of our approach to balancing the throughput of data I/O and model execution. Lower proportion allocated to quantization effectively shifts the occurrence of quantization rightwards to the higher total tolerance set on QoI. People might notice that in some intervals of the user-set tolerance on QoI, data points overlap across different tolerance allocation strategies. This is because of the limited option of quantization formats, and within certain intervals of the total tolerance, the exact same allocation configuration is selected despite the strategies.

In hydrogen combustion dataset, the turbulence is mainly concentrated around the single vortex at the center. As a result, the input data is easier to compress and it achieves a high compression ratio even for small tolerance levels. However, while the model execution performance is improved by quantization, it remains to be the bottleneck of inference performance. Although we could not fully balance the throughput of both phases in this case, we are able to boost the inference performance by prioritizing quantization in this task.

In experiments involving dissipation rate profiles and satellite image classification, we investigate a more balanced inference pipeline. Although the three datasets differ significantly, users can expect an approximate 5x speedup at a QoI level of 10^{-3}, as shown in Figs. 11 - 15. Notably, throughput improvement accelerates when users can tolerate a larger QoI error, establishing 10^{-3} as a key turning point. This behavior is primarily driven by FP16 quantization, which becomes effective with a tolerance of around 10^{-3}. Beyond enhancing model execution, FP16 quantization also reduces model sensitivity, enabling more aggressive input data compression without significantly compromising QoI accuracy. These findings underscore the critical role of combining compression and quantization to optimize inference performance. Other popular floating-point quantization formats, such as TF32 and BF16, are intentionally designed with fewer mantissa

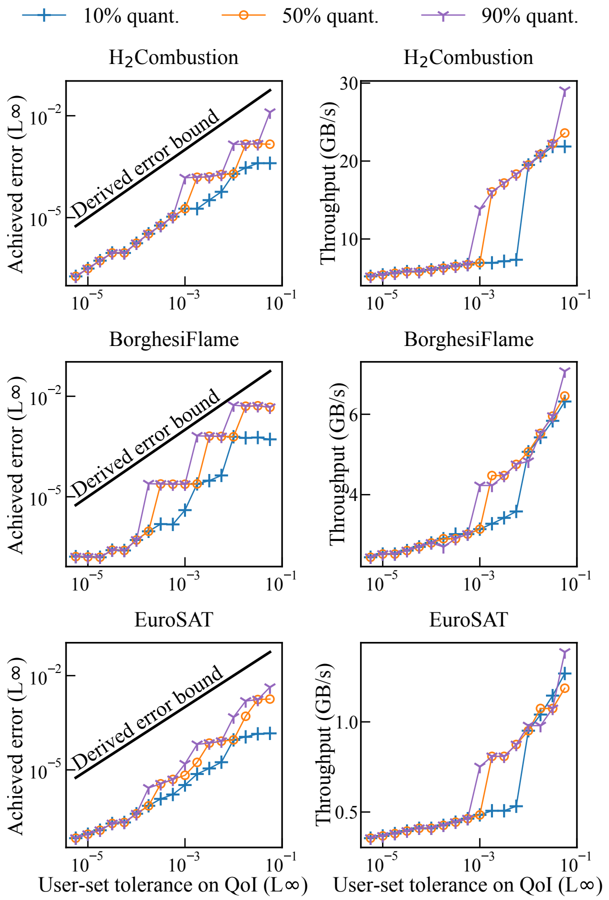

Fig. 15. Left: Predicted error bounds vs. user set tolerance in L-infinity. Right: Throughput vs. user set tolerance in L-infinity. Both results are obtained using ZFP as the compression backend.

bits, resulting in more limited improvements compared to FP16. Our findings suggest the potential benefits of developing lower-precision formats with increased mantissa bits to further enhance performance and compression efficiency for scientific computing workflows.

In the dissipation rate profiles and satellite image classification experiments, we observe that allocating a fixed proportion of the total tolerance to quantization does not consistently yield an optimal strategy across all tolerance values. Prioritizing either aggressive quantization or compression leads to suboptimal performance across different tolerance intervals. This highlights the need for an optimization algorithm to automate the determination of the optimal strategy, which represents a promising direction for future work. Additionally, the granularity of quantization can be improved by enabling per-layer quantization with different formats, thereby introducing a significantly larger optimization space.

V. Conclusion

In this work, we perform a comprehensive error flow analysis that encompasses errors introduced by both data compression and model quantization. We introduce a parameterized spectral normalization technique to enhance the stability and robustness of neural networks under various error scenarios. Our method effectively constrains the sensitivity of network outputs by directly regulating the spectral properties of weight matrices, thereby enabling the accurate and efficient prediction of error bounds on the final QoI prior to inference. Through extensive experiments, we demonstrate the accuracy and reliability of our proposed error bounds for composed compression and quantization errors. The bounds closely align with empirical error distributions, providing a theoretical foundation for predicting network behavior under perturbations.

This study underscores the importance of incorporating theoretical guarantees into deep learning models to improve their resilience and inference performance. By combining lossy compression and quantization, we observe a 5x speedup at a QoI level of approximately 10^{-3} in three distinct scientific workflows. Our findings also highlight the potential benefits of developing lower-precision formats with additional mantissa bits to further enhance performance and compression efficiency in scientific computing workflows. Notably, among current floating-point formats, FP16 offers a superior balance of cost and precision compared to BF16 and TF32, largely due to its higher ratio of mantissa bits.

VI. Future work

This study demonstrates the importance of combining compression and quantization to optimize inference performance in scientific workflows. While we provide a comprehensive discussion of compression techniques—covering the three most commonly used algorithms in scientific computing—further investigation into quantization remains open. In particular, advanced quantization methods beyond affine quantization, such as block-wise, column-wise, or row-wise schemes, are worth exploring for their potential to balance precision and overhead. Moreover, investigation on the new lower-precision floating-point formats could further enhance performance of scientific workflows. Our findings specifically suggest that formats with increased mantissa bits can offer improved efficiency while minimizing accuracy loss for scientific applications.

Looking ahead, we aim to extend this framework to a wider range of model architectures. Scientific community is increasingly deploying more complex surrogate models, such as U-Nets and transformers, rather than only MLPs or ResNets. This brings new challenges of higher computing demand. Adapting our approach to these architectures requires deriving the corresponding error-flow equations for their unique components, such as nested residual connections and attention mechanisms. We intend to deepen our theoretical foundations in subsequent research, with a special focus on applying these methods to transformer-based weather prediction tasks.

Acknowledgment

This research was supported by DOE ASCR SIRIUS-2 project and DE-SC0024424, NSF OAC-2311757, OAC-2144403, CCF-2134202, CNS-2340171, the Scientific Discovery through Advanced Computing (SciDAC) program for the RAPIDS-2 SciDAC institute.

References

[1] K. S. Jung, B. S. Soriano, J. H. Chen, and M. Khalil, "A hessian-based transfer learning approach for artificial neural networks based chemical kinetics with a sparse dataset," *Proceedings of the Combustion Institute*, vol. 40, no. 1-4, p. 105390, 2024.

[2] Z. Qiao, Q. Liu, N. Podhorszki, S. Klasky, and J. Chen, "Taming i/o variation on qos-less hpc storage: What can applications do?" in *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020, pp. 1–13.

[3] Z. Qiao, Q. Tian, Z. Qin, J. Wang, Q. Liu, N. Podhorszki, S. Klasky, and H. Zhu, "Tango: A cross-layer approach to managing i/o interference over local ephemeral storage," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis*, ser. SC '24. IEEE Press, 2024. [Online]. Available: https://doi.org/10.1109/SC41406.2024.00020

[4] A. Datta, K. F. Ng, D. Balakrishnan, M. Ding, S. W. Chee, Y. Ban, J. Shi, and N. D. Loh, "A data reduction and compression description for high throughput time-resolved electron microscopy," *Nature communications*, vol. 12, no. 1, p. 664, 2021.

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

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

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

[8] H. Wu, P. Judd, X. Zhang, M. Isaev, and P. Micikevicius, "Integer quantization for deep learning inference: Principles and empirical evaluation," *arXiv preprint arXiv:2004.09602*, 2020.

[9] H. Ootomo and R. Yokota, "Recovering single precision accuracy from tensor cores while surpassing the fp32 theoretical peak performance," *The International Journal of High Performance Computing Applications*, vol. 36, no. 4, pp. 475–491, 2022.

[10] D. Kalamkar, D. Mudigere, N. Mellempudi, D. Das, K. Banerjee, S. Avancha, D. T. Vooturi, N. Jammalamadaka, J. Huang, H. Yuen *et al.*, "A study of bfloat16 for deep learning training," *arXiv preprint arXiv:1905.12322*, 2019.

[11] Y. Zhou and K. Yang, "Exploring tensorrt to improve real-time inference for deep learning," in *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*. IEEE, 2022, pp. 2011–2018.

[12] J. Ansel, E. Yang, H. He, N. Gimelshein, A. Jain, M. Voznesensky, B. Bao, P. Bell, D. Berard, E. Burovski *et al.*, "Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, 2024, pp. 929–947.

[13] D. Kang, A. Mathur, T. Veeramacheneni, P. Bailis, and M. Zaharia, "Jointly optimizing preprocessing and inference for dnn-based visual analytics," *arXiv preprint arXiv:2007.13005*, 2020.

[14] I. Foster, M. Ainsworth, B. Allen, J. Bessac, F. Cappello, J. Y. Choi, E. Constantinescu, P. E. Davis, S. Di, W. Di *et al.*, "Computing just what you need: Online data analysis and reduction at extreme scales," in *Euro-Par 2017: Parallel Processing: 23rd International Conference on Parallel and Distributed Computing, Santiago de Compostela, Spain, August 28–September 1, 2017, Proceedings 23*. Springer, 2017, pp. 3–19.

[15] T. Banerjee, J. Choi, J. Lee, Q. Gong, R. Wang, S. Klasky, A. Rangarajan, and S. Ranka, "An algorithmic and software pipeline for very large scale scientific data compression with error guarantees," in *2022 IEEE 29th International Conference on High Performance Computing, Data, and Analytics (HiPC)*. IEEE, 2022, pp. 226–235.

[16] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale," *Advances in Neural Information Processing Systems*, vol. 35, pp. 30 318–30 332, 2022.

[17] R. Mises and H. Pollaczek-Geiringer, "Praktische verfahren der gleichungsauflösung." *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 9, no. 1, pp. 58–77, 1929.

[18] J. W. Lindeberg, "Eine neue herleitung des exponentialgesetzes in der wahrscheinlichkeitsrechnung," *Mathematische Zeitschrift*, vol. 15, no. 1, pp. 211–225, 1922.

[19] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[20] S. Ioffe, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[21] G. Borghesi, A. Krisman, T. Lu, and J. H. Chen, "Direct numerical simulation of a temporally evolving air/n-dodecane jet at low-temperature diesel-relevant conditions," *Combustion and Flame*, vol. 195, pp. 183–202, 2018.

[22] P. Helber, B. Bischke, A. Dengel, and D. Borth, "Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 204–207.

[23] "Summit system at oak ridge leadership computing facility," https://www.olcf.ornl.gov/summit/.

[24] "Frontier system at oak ridge leadership computing facility," https://www.olcf.ornl.gov/frontier/.

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

[26] M. Ainsworth, O. Tugluk, B. Whitney, and S. Klasky, "Multilevel techniques for compression and reduction of scientific data—the multivariate case," *SIAM Journal on Scientific Computing*, vol. 41, no. 2, pp. A1278–A1303, 2019.

[27] X. Liang, B. Whitney, J. Chen, L. Wan, Q. Liu, D. Tao, J. Kress, D. Pugmire, M. Wolf, N. Podhorszki *et al.*, "Mgard+: Optimizing multilevel methods for error-bounded scientific data reduction," *IEEE Transactions on Computers*, vol. 71, no. 7, pp. 1522–1536, 2021.

[28] J. Wang, T. Liu, Q. Liu, X. He, H. Luo, and W. He, "Compression ratio modeling and estimation across error bounds for lossy compression," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1621–1635, 2019.

[29] L. Lawrence Livermore National Security, "Final zfp r&d 100 award," Lawrence Livermore National Laboratory, Tech. Rep., 2023.