A Lightweight, Effective Compressibility Estimation Method for Error-bounded Lossy Compression

Arkaprabha Ganguli[†], Robert Underwood[†], Julie Bessac[‡§], David Krasowska[‖¶], Jon C. Calhoun[‖], Sheng Di[†], Franck Cappello[†]
[†]Argonne National Laboratory, Lemont, IL, USA [‡]National Renewable Energy Laboratory, USA
[¶]Northwestern University, USA [‖]Clemson University, USA [§]Virginia Polytechnic Institute and State University, USA
aganguli@anl.gov, runderwood@anl.gov, julie.bessac@nrel.gov, krasow@u.northwestern.edu,
jonccal@clemson.edu, sdi1@anl.gov, cappello@mcs.anl.gov

***Abstract*—Error-bounded lossy compression turns more and more important for the data-moving intensive applications to deal with big datasets efficiently in HPC environments, which often requires knowing the compressibility of the datasets before performing the compression. However, the off-the-shelf state-of-the-art lossy compressors are often driven by error bounds, so the compression ratios cannot be forecasted until the completion of the compression operation. In this paper, we propose a lightweight, robust, easy-to-train model that estimates the compressibility of datasets for different lossy compressors accurately. Our approach combines novel predictors that measure various notions of spatial correlation and smoothness exploited by lossy compressors that are implemented efficiently on the GPU in a framework and that uses mixture model regression to improve robustness with conformal prediction to provide bounds on the estimates. We then use these models with a detailed analysis of speedup to understand the tradeoffs between high speed, consistent speed, and accuracy of the methods on real applications. We evaluate our approach in the context of 3 key applications where compression ratio estimation is highly required.**

***Index Terms*—Lossy Compression, Compression Estimation, Rate Distortion, Error Bounded Lossy Compressors**

I. Introduction

Lossy compression is becoming an increasingly common element of strategy for large-scale data-intensive applications running on HPC systems. However, many real-world use cases are substantially faster with the accurate estimation of compression ratio beforehand and without requiring the expensive compression operations. For example (use case A): finding a best-fit configuration based on a compression ratio target for a particular dataset, the common method is performing a series of trials, which runs lossy compressors repeatedly based on different error bounds [1], [2]. (use case B): choosing the best compressor with the highest compression ratio from among a group of candidate compressors at runtime [3]. (use case C): In order to write multiple compressed datasets into one aggregated file [4] such as HDF5, it is necessary to forecast the compression ratios for each dataset because each parallel process needs to know the starting location of the compressed data to be written in the aggregated file.

This work was supported by the National Science Foundation with Grants SHF-1910197, SHF-1943114, OAC-2003709 and OAC-2104023, and the US Department of Energy under Project 17-SC-20-SC and Contract DE-AC02-06CH11357

Accurately estimating the compressibility of a dataset under an error-bounded lossy compressor is non-trivial. Although there have been some studies exploring this issue, they suffer from many limitations. Jin et al. [5], for example, proposed an efficient compression ratio estimator, but it is designed particularly for only prediction-based compression model, and may also be inaccurate in some datasets or error bounds (to be shown in our paper later). Some of the fast methods [6] are exceptionally inaccurate getting median absolute percentage errors over 90%, resulting in time to account for predictions or high space overhead up to 1.5× [4].

In this paper, we propose a novel accurate, robust, bounded, cheaper-to-train, and lightweight, and compressor-agnostic method to estimate compression ratios accurately for error-bounded lossy compressors using novel statistical predictors from datasets, and an analytical framework for analyzing the impacts of the accuracy and speed of the predictors has on the speedup on applications that use compression ratio estimation. **(1)** Our novel compression prediction model using powerful estimation methods: conformal prediction, which provides uncertainty associated with the predictions, with mixture regression to tackle diverse statistical properties of the data (Section IV-B) as the first statistically **bounded** method (Section VI-D) to estimate compression ratios. This method makes our approach far more **accurate** (Section VI-B) and **robust** (Section VI-C) to differences between datasets than prior approaches **(2)** Novel Metrics to better capture spatial structures than prior metrics: We create 3 (Spatial Correlation, Spatial Diversity, Spatial Smoothness) and identify 2 existing (General Distortion, Coding Gain) existing metrics. **(3)** Novel application use-cases: Our high accuracy and lightweight GPU approach (Section IV-C) enables new use-cases that are prohibitive with older, slower/less accurate methods. **(4)** We present the first mathematical model for speedup and accuracy trade-offs with prediction for each use case (Section V).

Compared with the existing error-bounded lossy compressibility estimation methods developed, our solution offers several key advantages: **Accurate** Our approach is substantially more accurate than prior approaches when evaluated the same way in prior work – using the same field for training and evaluation. **Robust** Our approach works also well when we consider using some fields from the same application to predict

other fields. To our knowledge, this is the first training-based approach to evaluate training on one field and prediction others on evaluation. **Bounded** Unlike prior works, our approach provides strong statistical bounds on the error of the compression ratio estimation using a conformal prediction-based approach that provides lower and upper bounds on each estimate – prior approaches provide no statistical guarantees on the error in estimation and provide only point estimates. **Cheaper to train** Prior model training-based approaches used the entirety of a field for training requiring costly exhaustive evaluation of metrics, whereas we provide a methodology to determine an order to consider fields for training that can reduce the time to prepare a model. **Lightweight** Not only is our approach accurate, but suitably lightweight and accelerates each of the uses cases mentioned in the introduction. **Speedup Analysis** Our work provides models of parallel speedup that account for prediction accuracy, speed, and consistency relative to the compressors to enable scientists to evaluate trade-offs between methods for the 3 use cases, and determine how much improvement can be achieved from incremental improvements to each of these approaches.

II. Background

Terms We begin by defining a few key terms for the purposes of this paper. Each **scientific application** may have multiple **runs**. A **dataset** refers to all of the data from a particular run of an application. Datasets from scientific applications can often be partitioned into individual **time-steps** representing stages of the simulation, which in turn has multiple **fields** representing distinct aspects of the simulation (e.g. pressure, temperature). A **buffer** refers to a single multi-dimensional array from an application belonging to a particular field and time-step. We assume that a field and time-step uniquely identify a buffer within a run.

Compressors While there is an accurate method for estimating the lossless compressibility – Shannon entropy [7], there is not a comparable measure of lossy compressibility that works across different applications each project-specific data features and different compressors that may differ significantly in design principles. In what follows, we introduce a few key compressors to be studied in this paper: **ZFP** [8] – ZFP compressor uses a common fixed point notation combined with a "near optimal" block transform – similar to the discrete cosine transform used in JPEG image compression. This approach operates at very high bandwidth, but often does not compress as much as SZ based compressors. **SZ2** [9] and SZ3 [10] and other related compressors use prediction based approaches to decorrelate the data prior to compression. For SZ2, prior work has shown that SZ2 is one of the hardest compressors to accurately estimate compression ratio because it uses multiple predictors (block regression and lorenzo) and special cases to account for [3]. **SZ3**'s compression ratio is easier to predict because it adopts an interpolation based predictor, however, is different in that it does not use a fixed block design like SZ2 does, meaning that the approaches that were designed based on SZ2 (such as Tao [6]) cannot be simply applied on SZ3. SZ3 often pairs high compression ratios with high quality reconstructions. **BitGrooming** [11] leverages aspects of the IEEE floating point specification to improve the compression of floating point data. **DigitRounding** [12] combines rounding with lossless compression techniques for a method that has robust compression performance with uncorrelated data. **MGARD** [13]–[15] uses linear representation theory to achieve high compression ratios and high quality. Additionally, the mathematical theory behind MGARD's design allows it to be used to guarantee the preservation of key quantities of interest that take the form [14] of or can be reduced to [16] a bounded linear functional. **TThresh** [17] – a slow, but highly effective compressor for 3D+ data based on the type of higher-order singular value decomposition. **SPERR** [18] is a relativley slow, but highly effective compressor for 2D and 3D based on wavelet decomposition.

III. Related Work

In this section, we introduce related work on estimating compression ratios of data when using lossy compressors that we compare against in the course of our work and identify key gaps in prior work. There have been a few major types of approaches:

Rate Distortion Theory Rate distortion theory is a well-established branch of mathematics that attempts to bound the compressibility [7], [19]–[21]. The benefit of these approaches is that if the assumptions made by these approaches hold, then it bounds the behavior of any possible future compressor. However, methods taken from rate-distortion theory often make very strict assumptions about the data – such as homoscedasticity and Gaussian distribution of the data source. Real-world datasets routinely violate these assumptions and resulting in the theory making estimates of the compressibility of data that vastly underestimate what lossy compressors that leverage these properties in real-world datasets can achieve. In particular, no existing approach considers correlations existing in scientific datasets.

Training-Based Statistical and Machine Learning Because theory-based approaches tend to be intractable for real-world use cases, there has been extensive work to consider using a series of predictors to estimate the compressibility of datasets with particular compressors using statistics or machine learning. These methods can be further subdivided into black-box and white-box methods. Black-box methods use predictors that are not derived from the internals of the compressors to be robust to changes in the compressors, whereas white-box methods model key aspects of the compressor to in principle get more accurate estimations or to allow counterfactual estimation.

Lu's approach [22] leverages a Gaussian model to combine several features that are internal to the SZ and ZFP compressors such as the number of nodes in the Huffman tree, and the number of outliers from mispredictions – quantities that require nearly running nearly the entire compressor to get accurate estimates of compressibility. This approach was later refined in [23] leveraging the deep knowledge of the internals

of the compressors to make counterfactual assessments of compressibility based on other encoding stages or prediction approaches.

Tao [6] developed a fast method but less accurate method that leveraged a minimal amount of information to estimate compression ratios for SZ2 and ZFP for online selection between these compressors. This approach samples data, estimates the probability density function of the data in the blocks, and then computes the entropy of the quantized values.

There have been two key approaches that use machine-learning style approaches. In response to the slowness of the applications like [1], Rahman developed a black-box approach [2] leveraging decision trees combined with generally applicable statistical predictors. Another related approach but whitebox comes from Qin [24] who takes a similar approach but uses deep neural networks but leverages internals of the compressor such as the number of nodes in the Huffman tree for SZ, but has slower and less accurate than their earlier work [22]. These methods while achieving higher accuracy than their predecessors lack explainability.

Recently there have been approaches by Jin [4], [5] that aim to get high speed and high accuracy. This white-box approach works for prediction-based compressors like SZ. It functions by first sampling blocks of data to be compressed, using their distribution and empirically observed properties of the predictors to estimate a bit-rate, and then estimating the encoding efficiency of the resulting data with a Huffman encoding using a run-length encoding that is easier to estimate recognizing that the compressor in an ideal case, produces decompressed data with quantization errors near zero.

Lastly, the paper by Underwood [3] focuses on high accuracy by leveraging a black-box statistical approach to estimating the compressibility of scientific data. It uses two predictors: the SVD truncation and the quantized entropy in a linear model to estimate compressibility. This approach is notable in both that it is reasonably accurate and fast when accelerated on the GPU, but it notably has robust performance across compressors and does not depend on any internals of the compressors. However, this method does not perform well in the worst case in out of field prediction (see Section VI-C)

Summary and Comparison to Related Work In this framework, our approach is a black-box statistical family similar to Underwood [3], and differs in its choice of predictors and most importantly in its performance in terms of robustness to dataset and compressor variability, but also in its runtime. We leverage predictors derived from intuitive notions that measure various notions of smoothness combined with more robust statistical approaches. These approaches allow us to make statistical guarantees regarding our accuracy and achieve much higher speed with improved runtime.

IV. Estimation Methodology

Studies like [3] suggest that a dataset's compressibility is reliant on its inherent statistical properties such as its spatial structure. Our work uses these statistical proprerties in a mixture model with conformal prediction to estimate compressability.

A. Predictors based on Notions of Correlation

In this section, we describe how to compute our predictors from a 2D array[1], denoted as $X \in \mathcal{R}^{p \times p}$. We divide it into B spatially connected blocks of dimension $k \times k$, denoted as $X_1, X_2, \ldots, X_B$, by considering row-wise divisions. It is important to note that each block is identified by specific row and column indices. Assuming there are B_c columns and B_r rows of blocks, the total number of blocks is given by $B_c \times B_r = B$, and the array size satisfies $p^2 = Bk^2$. We further represent the vectorized blocks as $X^b = vec(X_b) \in \mathcal{R}^{k^2}$ (row-wise), where $b = 1, 2, \ldots, B$, enabling a comprehensive set of samples for understanding the spatial structure of X. We proceed to discuss the formulation of our metrics.

Spatial Diversity (SD) weights entropy [7] with spatial information. Shannon's entropy [7] does not consider spatial correlation and diversity within the signal. Inspired by [25], we propose a novel method to estimate spatial diversity using spatial entropy. Our approach involves computing a weighted average that considers inter-block and intra-block variability as weights. These weights adhere to two fundamental principles: (1) When distant entities exhibit similarity, spatial variability increases (measured by inter-block variability, w_b^{inter}). (2) When adjacent entities demonstrate diversity, spatial variability increases (measured by intra-block variability, w_b^{intra}).

Next, we define the weights. The intra-block variability is expressed as the standard deviation of the block, denoted as $w_b^{intra} = sd(X^b)$. The inter-block variability incorporates both the Euclidean distance between the values ($D^e_{b,b'}$) and manhattan distance between the block locations ($D^s_{b,b'}$). Together inter-block variability is : $w_b^{inter} = \frac{\sum_{b' \neq b} D^s_{b,b'} D^e_{b,b'}}{\sum_{b' \neq b} D^s_{b,b'}}$.

Finally, instead of using the generic estimator of entropy, denoted as $H = -\sum_{b=1}^{B} p_b log_2(p_b)$, we define spatially informative entropy as $SD = -\sum_{b=1}^{B^2} w_b^{intra} w_b^{inter} p_b log_2(p_b)$. Here p_b is the probability of block b. A simple way to assign these probabilities is to assume equal likelihood among the blocks, yielding $p_b = \frac{1}{B}$ for $b = 1, 2, \ldots, B$.

Spatial Correlation (SC) is a weighted average of correlation between pairs of blocks weighted by w_b^{intra}. To assess spatial correlation, we compute the correlation matrix $V \in \mathcal{R}^{B \times B}$ with elements $V_{b,b'} = \rho(X^b, X^{b'})$, where ρ represents the Pearson correlation coefficient. Additionally, we define the statistic SC_b to quantify the spatial correlation per block: $SC_b = \frac{\sum_{b' \neq b} D^s_{b,b'} |V_{b,b'}|}{\sum_{b' \neq b} D^s_{b,b'}}$. Here, SC_b measures the strength of long-range correlation in the b^{th} block by assigning weights to pairwise correlations based on spatial distances. Notably, if a highly variable block exhibits a high correlation with other blocks, it contributes more to the overall spatial correlation. Thus, to incorporate this concept, we calculate the weighted average of these spatial correlations: $SC = \frac{\sum_{b=1}^{B} SC_b w_b^{intra}}{\sum_{b=1}^{B} w_b^{intra}}$.

[1] extensions to 3D are possible using approaches similar to [3]

Generic Distortion Measurement The measurement of generic distortion is inspired by the rate-distortion curve [26] used to assess transform coding. Without assuming a Gaussian distribution, for a fixed-rate R , the optimal expected distortion is a function of entropy and quantized entropy: $D \equiv \frac{1}{12}2^{2h(x)}2^{-2R}$. Here, $h(x) = -\int_{\mathcal{R}} f_x(t)log_2 f_x(t)dt$ denotes the differential entropy of the data, obtained by estimating the data distribution f_x, and R represents the average quantized entropy expressed as R=$\frac{1}{k^2}H(\alpha(X^b))$. Consequently, the distortion metric D encompasses various unknown properties of the source, including the data distribution f_x. Therefore, to estimate D, it is necessary to obtain estimates for these quantities. To measure the quantized entropy $H(\alpha(X^b))$, a simple linear scheme $\alpha(x, \epsilon_{abs}) = \lfloor x/\epsilon_{abs} \rfloor * \epsilon_{abs}$ is employed, where $\lfloor x \rfloor$ denotes the floor function, and ϵ_{abs} is the chosen number of subdivisions of the domain. Next, we utilize the empirical data distribution to nonparametrically estimate the probability distribution of the original data and its quantized version. This provides us with the estimated entropy and quantized entropy, denoted by H_b and H_b^q, respectively, to finally compute the estimated generic distortion measure $\hat{D} = \sum_{b=1}^{B} \frac{1}{12}2^{2H_b}2^{-2H_b^q/k^2}$.

Coding Gain In transform coding, the coding gain serves as a crucial metric for assessing the transformation stage's efficacy. Within this context, the Karhunen-Loève Transform (KLT) holds a prominent position as the optimal orthogonal transformation method for minimizing the bit rate (and thus coding gain) for Gaussian designs [26], [27]. The coding gain quantifies the reduction in distortion achieved by applying the KLT for decorrelation, assuming a high rate and optimal bit allocation. We calculate the block covariance matrix as $\Sigma = \frac{1}{B}\sum_{b=1}^{B} X^b(X^b)^T \in \mathcal{R}^{k^2 \times k^2}$, where X^b denotes the data block and ^{T} the matrix-transpose operation. The coding gain is then expressed as a function of the singular values of Σ. Specifically, Coding Gain $= \frac{\left(\prod_{i=1}^{k^2}(\Sigma)ii\right)^{1/k^2}}{\left(\prod_{i=1}^{k^2}\tilde{\sigma}_i^2\right)^{1/k^2}}$, where $\tilde{\sigma}_i$ represents the singular values. The detailed theoretical derivation for the case of a heterogeneous Gaussian source is found in [26]. Although coding gain is only proven to be optimal for Gaussian sources, it remains an indicator compressibility for other data [28].

Spatial Smoothness (CovSVD-trunc) The singular value decomposition (SVD) is effective in assessing spatial dependence [29], [3]. This research builds upon the concept of measuring spatial smoothness through the "CovSVD-trunc" metric. The CovSVD-trunc applies the SVD to the block correlation matrix Σ (defined in IV-A). Specifically, it calculates the percentage of singular values necessary to capture 99% of the total variance of the blocks, represented as CovSVD-trunc $= \frac{m}{k^2} \times 100$, where $m = \min_{r=1,2,\ldots,k^2} r \ni \frac{\sum_{i=1}^{r}\tilde{\sigma}_i}{\sum_{i=1}^{k^2}\tilde{\sigma}_i} \geq 0.99$. While the aforementioned SC measure primarily accounts for long-range dependencies, the CovSVD-trunc metric focuses on intra-block correlations.

Ablation Study Figure 1 highlights the difference between the metrics through an ablation study on the hurricane dataset.

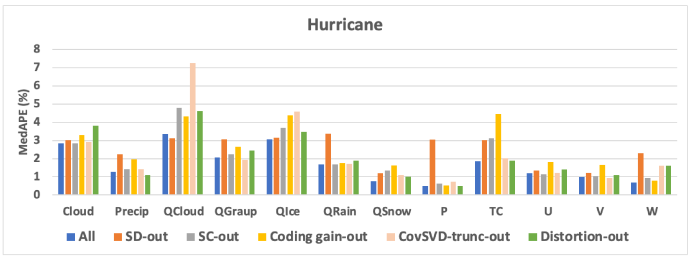


Fig. 1: Ablation Study for statistical predictors

For this study, we construct the model that we detail in Section IV-B using the methodology from Algorithm 2, but exclude/ablate each of the statistical predictors one at a time and compare the median absolute percentage error with the fully specified model. For the study, we train the model on each of the fields individually and then predict on other buffers from the same field (in-sample prediction, see also VI-B) and present results from the SZ3 compressor which traditionally has higher prediction error for prior methods. As can be seen from the figure, each of the different fields is affected the most by the exclusion of different predictors. For example, the qcloud field is sensitive to the exclusion of our predictor CovSVD-trunc whereas the field p is sensitive to the exclusion of the spatial diversity. This suggests that each of the statistical predictors provides a distinct type of information to the model and that they capture distinct aspects of the relationships between data points that influence compressibility.

The aforementioned statistical predictors are closely linked to the compressibility of datasets, as they incorporate various spatial characteristics. Specifically, the spatial diversity (SD), spatial correlation (SC), and CovSVD-trunc collectively capture both long-range and short-range spatial variability. Moreover, the generic distortion measure and coding gain assess the overall compressibility of the data from a compressor-free standpoint. In the next subsection, we demonstrate how these features' predictive capabilities are integrated into a machine learning model, establishing an effective and uncertainty-aware prediction framework for compression ratio (CR).

B. Estimation Method

Our research aims to develop an uncertainty-aware prediction model for compression ratio using the aforementioned predictive features. To predict CRs, we rely on regression models to model the relationship between the CR and its statistical predictors from Sect. IV-A. For each compressor and each dataset field, regression models are fitted between the statistical predictors and associated compression ratios. To mimic most operational conditions, we focus on CRs less than or equal to 100. In practice, few users work with higher compression ratios (except for visualization, which is not a target use cases of this study).

Initial data analysis revealed significant heterogeneity and grouping effects in the relationship between these features and the CR, an empirical validation is presented in Figure 2 using the Hurricane dataset. To capture this group structure,

we employ a mixture of regression models, a well-established framework in statistical analysis [30]. This approach allows for the identification and estimation of distinct patterns or clusters within the data, effectively addressing the presence of grouping effects. Furthermore, to address prediction uncertainty, we incorporated the conformal prediction framework [31], [32]. Conformal prediction provides valid confidence measures or prediction regions for individual predictions, enabling the quantification of uncertainty associated with machine learning models. It aims to produce reliable and calibrated predictions while controlling error rates. In this section, we discuss each of these approaches in detail.

1) Developing the prediction model: In our regression analysis, we consider a sample comprising n 2D-slices as individual data points, where the outcome variable is $log(CR)$ and the five covariates are: SD, SC, Coding gain, CovSVD-tunc, and generic distortion $\hat{D}$. We represent the predictor vector as $x = (x_1, x_2, \ldots, x_5)$, where x_{ip} denotes the observed value of the p^{th} feature for the i^{th} data point. Assuming there are L latent groups in the data, the contribution of each class to the overall density is estimated by $\pi_1, \pi_2, \ldots, \pi_L$, which represents the probability of being in each class. Using this formulation, the joint distribution of $y|x$ can be expressed as:

$$f(y|\Lambda, x) = \sum_{l=1}^{L} \pi_l f_l(y|\theta_k, x) \tag{1}$$

where $\Lambda = (\Pi, \Theta)$ denotes the vector of all unknown parameters to be estimated, i.e., $\Pi = (\pi_1, \pi_2, \ldots, \pi_{L-1})$ the cluster-allocation probabilities and $\Theta = (\theta_1, \theta_2, \ldots, \theta_L)$ is the set of regression parameters in each of the clusters $\{1, 2, \ldots, L\}$. In the linear regression setting, the cluster-specific regression function $f_l(Y|\theta_l, x)$ can be written as:

$$f_l(Y|\theta_l, x) = \beta_{0l} + \sum_{p=1}^{5} \beta_{pl} x_{ip} + \epsilon_{il}, \quad \epsilon_{il} \sim \mathcal{N}(0, \tilde{\sigma}_l^2). \tag{2}$$

Here for the latent class l, β_{0l} is the vector of class-specific intercepts, $\tilde{\sigma}_l^2$ is the noise variance for class l, and β_{pl} is the vector of regression coefficients for covariates x_p. The class-specific coefficients identify this as a regression mixture model tailored for heterogeneous datasets with complex group structures. In practice, the latent class dimension L is a hyperparameter and we set its value by fitting a clustering method like k-means.

2) Addressing the uncertainty with conformal prediction: Conformal prediction provides a rigorous framework for quantifying and managing prediction uncertainty in a regression framework by offering reliable confidence measures or prediction regions for individual predictions, without relying on any distributional assumptions [33]. Given a training dataset $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, a new predictor vector x_{n+1}, and a prediction scheme such as the mixture of regression Equation IV-B1, our objective is to construct a $(1-\lambda)100\%$ confidence interval $\hat{C}n(x_{n+1})$ for the unobserved target variable y_{n+1}, ensuring that $P(y_{n+1} \in \hat{C}n(x_{n+1})) \geq 1 - \lambda$, with P the considered probability measure. For example, when considering a confidence level of 95%, a well-calibrated conformal predictor in a regression scenario would yield confidence intervals that encompass the true value in at least 95% of instances (Figure 6). Specifically, we adopt the split conformal prediction scheme for its computational flexibility.

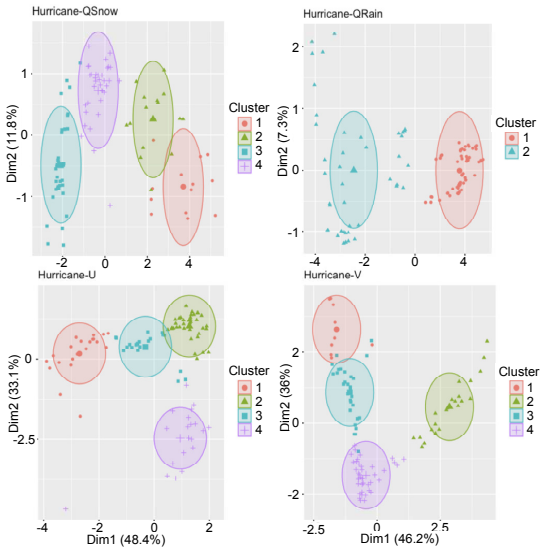


Fig. 2: Visualization of the latent clustering structure in the relationship between compressibility and predictors using four different fields from the Hurricane dataset. To capture the clustering effects in the 6-dimensional multivariate dataset comprising the compression ratio (CR) and the five proposed features, principle component analysis (PCA) was utilized to reduce the dimensionality to two and visualize the clusters on this new basis. The plot displays data points based on the top two principal components, highlighting a clear and noticeable grouping effect. These findings underscore the need for a mixture of regression models to accurately capture the complex associations observed.

The split conformal prediction algorithm involves several crucial steps. Initially, the training dataset is divided into a proper training set and a calibration set. The proper training set is utilized to train the prediction model, while the calibration set is used to estimate the prediction error. By incorporating the notion of variability into the prediction through a set of residuals, the algorithm achieves robustness. Algorithm 1 provides an overview of the key steps involved. This procedure is efficient – stages 1-5 take $O(N)$ time and can be precomputed before inference, and stage 6 takes $O(1)$ time per inference.

C. Implementation

We implement our predictors as a combined multi-threaded CPU+GPU code in Julia 1.8 using CUDA.jl, TiledIteration, and the Atomix packages. Source code for our implementation can be found on Github[2] Where possible, we pre-allocate memory to be used for the operations to avoid the need for garbage collection and memory allocation during

[2]upon acceptance link https://github.com/robertu94/libpressio-predict/

Algorithm 1 Prediction error evaluation and quantification procedure

Require: Training $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, new feature x_{n+1}, the mixture regression algorithm $\hat{f}$ in Eq. IV-B1, level λ, calibration set size m

1: Split $\{1, 2, \ldots, n\}$ into training set L of size r and calibration set I of size $m = n - r$;
2: Train $\hat{f}_L(x) = \hat{f}(x; (x_l, y_l), l \in L)$;
3: Compute the residuals $\tilde{R}_i = |y_i - \hat{f}_L(x_i)|, i \in I$;
4: Sort the residuals in an increasing order: $\tilde{R}_{(1)}, \tilde{R}_{(2)}, \ldots, \tilde{R}_{(m)}$.
5: Compute the $(1-\lambda)^{th}$ quantile: $\tilde{R}_\lambda = \tilde{R}_{(k)}$ where $k = \lceil (1-\lambda)(m+1) \rceil$;
6: **return** The $(1-\lambda)100\%$ confidence interval

$$\hat{C}_n(x_{n+1}) = \{y \in \mathcal{R} : |y - \hat{f}_L(x_{n+1})| \leq \tilde{R}_\lambda\} = \left[\hat{f}_L(x_{n+1}) - \tilde{R}_\lambda, \hat{f}_L(x_{n+1}) + \tilde{R}_\lambda\right]$$

execution. Our implementation combines the routines of all of the dataset-specific but error-bound agnostic predictors into a single routine to minimize loads. We then execute each pair of blocks in parallel on the CPU. We offload a few key performance-critical operations to the GPU – namely, the eigendecomposition and the outer product used in computing the spatial diversity. Lastly, we remove the need for locking by using atomic instructions to handle the sums that are shared between threads on the CPU to avoid the high-overhead use of a mutex. One exception is that once for each block, we need to add an entire array of values atomically as part of the SVD truncation calculation – we found through profiling that a single mutex was more efficient than an entire sequence of atomic additions. The runtime of the error bound agnostic metrics is $O\left(\frac{p^2}{k*n_c} + \frac{p*k}{n_c\gamma} + \frac{k^6}{\gamma}\right)$ where p number of rows of the matrix, k is the number of rows in each tile, and n_c is the CPU scaling factor, and γ is the GPU core scaling factor. The three terms that bound performance come from the computation of norms of the pair of tiles, the computation of the outer product of each tile, and the SVD in the CovSVD-trunc. Since the tile size k is small and fixed, the $O\left(\frac{p^2}{k*n_c}\right)$ term dominates. For the error bound specific metrics, the bound is $O\left(\frac{k^2 \log k}{n_c}\right)$ and driven by the computation of entropy in the generalized distortion.

V. use cases and Performance Models

In the Introduction, we previewed 3 uses cases for using compression ratio estimation in real applications to achieve speedups: (A) using running lossy compressors to meet a specific compression ratio target [1], (B) choosing amongst a group of compressors which has the greatest compression ratios under a given set of constraints [6], and (C) quickly finding the offsets needed to write into a single HDF5 file in parallel. Additionally, we can also model speedup in training the model. In this section, we model the performance of these both training a model and these use cases to provide insight into the impacts of runtime consistency, runtime latency, and accuracy of predictions on the speedups observed by applications. These results complement the empirical results in Section VI to provide a more comprehensive picture of performance and trade-offs between various approaches. We define common terms in Table I.

Notation	Meaning
$\mathcal{N}(\mu, \sigma)$	normal distribution with mean μ, standard deviation σ
Φ	cumulative density function for $\mathcal{N}(0, 1)$
$e \sim \mathcal{N}(\mu_e, \sigma_e)$	time of predictors dependent on data and error bound
$d \sim \mathcal{N}(\mu_d, \sigma_d)$	time of predictors dependent on data
$y \sim \mathcal{N}(\mu_y, \sigma_y)$	time of computing an estimate
$c_i \sim \mathcal{N}(\mu_{c_i}, \sigma_{c_i})$	time of running the i^{th} compressor
$n_s \in N$	number of searches performed
$n_c \in N$	number of compressors to consider
$n_p \in N$	number of processors to use
$n_b \in N$	number of buffers to compress
$n_m \in N$	number of compressed buffers that fit on a processor

TABLE I: Notation

A. Assumptions

We make a few key modeling assumptions: First, we assume a memory-constrained environment. Specifically that we can fit the original dataset in its entirety and no more than n_m compressed buffers (and associated scratch space) into memory per processor. This assumption represents the real-world use case where compression is running in-situ with an application that heavily uses memory and only a limited amount is available for compression. This assumption can be relaxed by setting $n_m = n_b$. Second, we assume that the average runtime of the compressors and estimation methods have a Gaussian distribution – this appears to be validated by our preliminary testing.

B. Supporting Theorems

A foundational result in statistics show given two normal distributions $A = \mathcal{N}(\mu_a, \sigma_a), B = \mathcal{N}(\mu_b, \sigma_b)$, $\mu_{a+b} = \mu_a + \mu_b$ and $\sigma^2_{a+b} = \sigma^2_a + \sigma^2_b$ From this we derive that the sum of some positive integer k such distributions with equal mean and variance is $k\mu$ and variance $k\sigma^2$.

Further, the paper by Elfving [34] shows that the expected maximum of a group of n samples from a normal distribution is asymptotic $\mu + \sigma\Phi^{-1}\left(\frac{n-\pi/8}{n-\pi/4+1}\right)$ where μ is the mean, σ is the standard deviation, and Φ^{-1} is the inverse cumulative density function of the standard normal distribution. We define $\mathcal{W}(\mu, \sigma, n_t, n_p) = \lceil \frac{n_t}{n_p} \rceil \left(\mu + \sigma\Phi^{-1}\left(\frac{n_p-\pi/8}{n_p-\pi/4+1}\right)\right)$ to be the expected time to run these n_t tasks on n_p processors.

C. Use case A: Searching for a target CR

The expected parallel speedup for use case A using estimates vs not using estimates is

$$\frac{\mathcal{W}(\mu_c, \sigma_c, n_s, n_p)}{\mu_d + \mu_c + \mathcal{W}(\mu_e + \mu_y, \sqrt{\sigma^2_e + \sigma^2_y}, n_s, n_p)}.$$

In the no-estimation case, we need to run the compressor on each buffer for the number of search iterations to use. In the estimation case, we run the dataset-specific predictors once, then the error-bound specific predictors for the number of search iterations, followed by only the compressor only once.

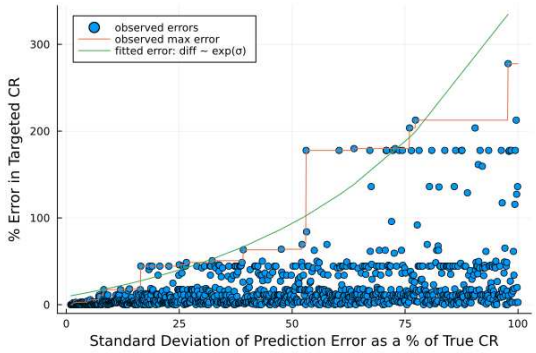


Fig. 3: Use case A: Inaccuracy in the estimates leads to an exponential degradation in the quality of the estimates. Thus, high accuracy methods are preferable for this use case

When running in parallel, we expect $\lceil \frac{n_s}{n_p} \rceil$ executions. In both the estimation and original case, when running in parallel we have to wait for the last process to complete, requiring us to use Elfving's formula.

For use case A, the accuracy needs to be very high to unobtrusively replace an estimate with running the compressor. While very preliminary, we estimated the effects of degradation in the quality of estimates by increasing the levels of prediction error by modeling prediction errors as Gaussian errors and injecting increasing prediction errors, and measuring the difference from the un-perturbed solution. As shown in Figure 3, we found that the error as measured as a percent of the true compression ratio appears to degrade with an exponential of the variance of the prediction error as measured by the percentage of true compression ratio indicating that estimates with error over a few percentage points over the desired target are likely useless for this use case. In Figure 3, .5%, 1%, 2%, 4%, 8% errors respectively led to 9.9%, 10.3%, 11.2%, and 17.4% respectively, but we expect the exact differences to vary across datasets.

D. Use case B: Searching for the highest CR within constraints

In the no-estimation case, we need to run each compressor once and then re-run the optimal compressor. In the estimation case, we only need to run the dataset and error-bound specific predictors once, then we compute the model estimates from these statistics. In practice, computing the model estimates from the statistics takes only nanoseconds compared to other tasks that take an order of milliseconds can be ignored in most cases.

The expected parallel speedup for use case B using estimates vs not using estimates is approximately

$$\frac{\mathcal{M}\left(\mu_{c_i}, n_p\right) + \mu_{c_{opt}}}{\mu_e + \mu_d + \mathcal{W}(\mu_y, \sigma_y, n_c, n_p) + \mu_{c_{opt}}}.$$

Note that Elfving's formula does not apply in this case because we do not have repeated samples from a single distribution and the variance of the compressors can differ. Instead, we need to compute the minimal makespan on n_p processors $\mathcal{M}(\mu_{c_i}, n_p)$ that is the minimal time to schedule all tasks in parallel. While in the general case, minimal makespan is NP-Hard that is not a concern in this case: 1) often there are either more processors than compressors under consideration or only a single processor in which case this problem devolves to $\max_i \mu_{c_i}$ or $\sum_i \mu_{c_i}$ respectively, 2) open-source solvers can solve optimal realistic-sized versions of this problem optimally in less than a second, as real-world use cases seldom have more than 30 compressors, 3) in the worst case, $\mathcal{M}(\mu_{c_i}, n_p)$ can be approximated using list scheduling with an approximation bound of $2 - \frac{1}{n_p}$ of the optimal [35].

For use case B, we can model errors in this way. We get an incorrect solution if and only if we predict another compressor gets a higher compression ratio than the best one. We can estimate the probability of this using the variance of our estimates, and the mean of our compression ratios on a datasets for each compressor is $X_i \sim \mathcal{N}(\mu_{CR_i}, \sigma_{CR_i})$. If we assume independence and normality of CR_i, the probability of an incorrect conclusion is $1 - \Pi_{i=1}^{n} \left(p\left(CR_0 \leq CR_i\right)\right)$ where $P(CR_0 \leq CR_i) = \Phi\left(\frac{\mu_{CR_0} - \mu_{CR_i}}{\sqrt{\sigma^2_{CR_0} + \sigma^2_{CR_i}}}\right)$ this probability degrades to $\Phi\left(\frac{\mu_{CR_0} - \mu_{CR_i}}{\sqrt{\sigma^2_{CR_0} + \sigma^2_{CR_i} + \sigma^2_{CR_{err_i}} + \sigma^2_{CR_{err_0}}}}\right)$ when switching estimates. For example, if there were compressors with mean compression ratios 1, 2, and 3 on a dataset, with a variance of .1 each. Estimate error variances of .0625, .125, .25, and .5 result in expected inversions 3.9%, 6.9%, 12.3%, 20.8% of the time.

E. Use case C: Parallel writes to a file

The expected parallel speedup for use case C is

$$\frac{\mathcal{W}(\mu_c, \sigma_c, n_b, n_p) + \mathcal{W}(\mu_c, \sigma_c, n_b - n_m, n_p)}{T_{est} + \mathcal{W}(\mu_c, \sigma_c, n_b, n_p) + T_{miss}}.$$

Where the time to compute the estimates is $T_{est} = \mathcal{W}(\mu_e + \mu_d + \mu_y, \sqrt{\sigma_e^2 + \sigma_d^2 + \sigma_y^2}, n_b, n_p)$ and the time to handle mispredictions is $T_{miss} = \mathcal{W}(\mu_c, \sigma_c, \max\left(0, \lceil \frac{m n_b}{n_p} - n_m \rceil\right), n_p)$ In the no-estimation case, we need to run compression of each buffer twice, once to get the compression ratio and again to store the data in compressed storage. In the estimation case, we can replace the first set of compression calls with a series of calls that estimate the compression ratio and handle the performance effects of mispredictions using the method from [4]. With probability m, we underpredict the compression ratio in this case, we need to wait and re-compress the data and write the data to an auxiliary location now that we know its true compression ratio.

F. Training Time

In addition to our use cases, we can also model the time that it takes to produce a model. The speedup from reducing

training time and potentially changing training methods is

$$\frac{\mu_t + \mathcal{W}\left(\mu_d + \mu_e + \mu_c, \sqrt{\sigma_d^2 + \sigma_e^2 + \sigma_c^2}, n_b, n_p\right)}{\mu_{t'} + \mathcal{W}\left(\mu_{d'} + \mu_{e'} + \mu_c, \sqrt{\sigma_{d'}^2 + \sigma_{e'}^2 + \sigma_c^2}, n_{b'}, n_p\right)}.$$

In both cases, we need to run the compressor and the error bound specific and dataset specific metrics on each of the buffers in parallel. The key differences come from the difference between n_b and $n_{b'}$ and from the speed and runtime consistency of the dataset and error bound specific predictors.

VI. Experimental Evaluation

A. Experimental Setup

We begin with a few aspects in common to all our experiments. We ran evaluations on the compressors listed in the Background. We present a small representative subset of these results in this results section due to space. Unless otherwise mentioned we focus on the SZ3 compressor because it is especially difficult to predict producing the worst results for our method and to enable comparisons to related work (e.g. [3], [6], [22]), but have conducted similar studies on ZFP and SPERR. We also focus on absolute point-wise error bounds of 1e-3 unless otherwise specified for space in the paper, running experiments with other pointwise absolute bounds 1e-4 and 1e-6 finding similar results. We interfaced with the compressors using LibPressio [36] from its Julia bindings to facilitate the comparisons. Additional results are available at Zenodo[3].

1) Data: We use datasets from SDRBench [37]: NYX (cosmology), Hurricane (weather prediction), and Miranda (hydrodynamics turbulence simulation) chosen for diversity, availability, and use in prior work. These datasets are all natively 3D datasets. We convert them to 2D datasets by slicing along the slowest incrementing dimension to increase the volume of training and testing (c.f. [3], [38]).

2) Evaluation System: Large-scale evaluations were performed on machines with an Intel Xeon Phi 7230, with 96GB of DDR4 Ram selected for availability. Performance experiments were run on nodes with 11th Generation Intel Core i7-1185H, a Nvidia A2000 GPU, and 32 GB of DDR4 RAM where experiments could be run in isolation.

3) Evaluating Estimation of Compressors: Common predictors used to compare the accuracy of compression estimation methods are the 10% quantile, 50%, and 90% quantile of the median absolute percentage error [3]. These predictors are robust against extremely accurate or inaccurate estimations of compression ratio and provide a concise summary of how well a method works across an entire dataset in a way that discourages over-fitting. These are computed according to the procedure outlined in Algorithm 2, a k-fold cross-validation procedure. For each fold, compute the specified predictors, and observe the compression ratio for the training and testing data (see lines 4-7). The predictors are often divided into groups that are dependent on the error bound (`eb_predictors`) and those specific to the buffer but agnostic to the error bound (`dset_predictors`) to reduce re-computation (see line 6). Next, fit a model on the training data, and predict on the testing data (line 8). For each prediction, compute the absolute value of the true minus the predicted compression ratio, and convert it to a percent (line 11-14). After all predictions for a fold are completed, compute the median of the fold, and report the 10%, 50%, and 90% quantiles of the medians from the folds (line 18).

[3]https://zenodo.org/record/8150806

Algorithm 2 Prediction Error Evaluation and Quantification Procedure

Input: Dataset D, user-specified error bound e
Output: 10%,50%, 90% Quantiles of the Median Absolute Percentage Error

```
1: medape ← []
2: for train, test ∈ kfold(D) do
3:     true_cr ← [], predictors ← []
4:     for d ∈ train do
5:         true_cr.append(size(compress(d,e))
6:         predictors.append([dset_predictors(d), eb_predictors(d,e)])
7:     end for
8:     model ← train(true_cr, predictors)
9:     ape ← []
10:    for d ∈ test do
11:        true_cr ← size(compress(d,e))
12:        predictors ← [dset_predictors(d), eb_predictors(d,e)]
13:        pred_cr ← predict(model, predictors)
14:        ape.append(100 (true_cr - pred_cr) / true_cr)
15:    end for
16:    medape.append(median(ape))
17: end for
18: return quantile(medape, [0.1, 0.5, 0.9])
```

B. Major Result 1: Accuracy – In Sample Prediction

Existing work has largely focused on evaluating the accuracy of what we describe as in-sample prediction – training and testing on different subsets of the same field from the same application. In-sample prediction is important because it represents 1) an ideal case for compression ratio estimation where data is homogeneous and 2) represents a case where a model can be produced for each field of an application.

We present visual results for our method in-sample prediction to enable direct comparisons to existing work in Figures 6a and 6c made using Algorithm 2. In the figures, the black dots represent the actual observations vs the predicted values using our method, the black line represents the line $predicted = actual$ the optimal prediction, we address the confidence intervals in these plots in Section VI-D. The tight clustering about the optimal prediction line and the consistency of the variance indicates high-quality prediction.

We also include results for our methods across 4 datasets with 3 different compressors and 2 error bounds to show the effectiveness of our approach in a wide variety of contexts in Figure 4. This plot shows 3 fields from each of four separate datasets on the x-axis. On the y-axis, it shows a box and whiskers plot for the median absolute percentage errors as determined by Algorithm 2. The legend shows the average and maximum error for each compressor and the error bound across all datasets. The largest median absolute percentage error is 5.3% for SPERR. We get an overall average median absolute percentage error of 1.2. This extremely high accuracy

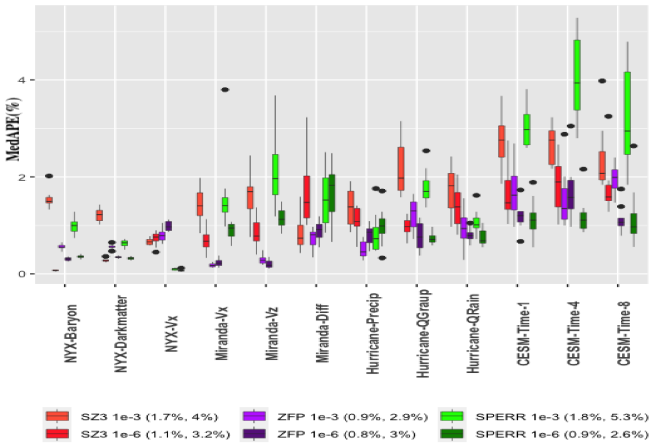


Fig. 4: Summary of the performance of our approach. Y axis shows the distribution of median absolute percentage errors for the k-fold validation procedure from Algorithm 2. X axis shows the particular field and dataset. dots are values more than 1.5 $\times$the interquartile range greater or less than the top or bottom quartile. The values in () represent the average error and max error for a compressor across all experiments.

enables the various use cases that we presented. We compare our approach to leading approaches in Table II across leading methods mentioned in Section III on the Miranda VX dataset. **Key Finding 1** We observe substantially lower estimation error at 10%, 50%, and 90% levels with a 3.8$\times$ improvement in the median error over [3] the leading competitor.

C. Major Result 2: Robustness – Out of Sample Prediction

Predicting in-sample, while useful, has several key limitations addressed by using what we call out-of-sample prediction: 1) by using out-of-sample prediction we can speed less time collecting training data (see Section VI-E). 2) we should be more robust to the particularities of any one dataset. In this section, we consider how our approach compares in terms of accuracy for out-of-sample prediction. For out-of-sample prediction, we consider multiple fields from the hurricane dataset and train on a subset of them and predict on a different one. We choose the Hurricane dataset because it has a relatively large, but manageable, number of fields. Again, we use the procedure in Algorithm 2 to assess the accuracy of predictions.

We show results for CLOUD and PRECIP fields in Figures 6b and 6d respectively. We observe that despite training on different fields produces comparable accuracy to the in-sample cases shown in Figures 6a and 6c with results clustered the optimal prediction line and consistency of residuals again indicates a high quality prediction. In Table II we conduct out-of-field prediction for all fields in Hurricane using the 4 most similar fields with our method and the method from Underwood [3]. We don't consider Lu [22] or [6] as they do not support out of sample prediction. In the worst case, prior approaches like Underwood [3] have very extreme outlier

Type	Method	10%	MedAPE	90%
Out-of-Sample	Underwood [3]	1.11e11	1.30e11	1.35e11
	Proposed method	12.5	12.5	17.2
In-Sample	Underwood [3]	0.9	2.7	3.8
	Tao [6]	82	90	93
	Lu [22]	157	193	256
	Proposed method	0.16	0.71	3.5

TABLE II: Worst Out of Feild Prediction Error on Hurricane trained with 4 fields for SZ3. Tao [6] and Lu [22] do not support out of sample prediction so are ommited In-Sample Predication Accuracy on 2D Slices of Miranda VX for SZ3 - 1e-6

mispredictions. For example, when predicting with V with TC, U, CLOUD, and PRECIP, Underwood [3] get a median absolute percentage error of nearly 1.35e11%. In contrast, our approach get a worst 90% APE 17.2 across all fields. **Key Finding 2** We vastly improve accuracy on out of sample prediction from 1.35e11% to 17.2% compared to [3].

D. Major Result 3: Bounded – Conformal Prediction

Prior work has focused on providing point estimates for compressibility. By introducing conformal prediction, we now have statistical bounds on each estimate produced by our methods. This allows us to 1) describe the uncertainty in individual point estimates 2) compute and measure trade-offs between accuracy of estimations and the amount and diversity of data collected. We present confidence intervals from the conformal prediction in Figure 6. For example, we can observe the greater uncertainty in the out-of-sample prediction cases compared to the in-sample prediction. Lastly, we computed the percentage of cases that the $predicted = actual$ line that escapes the confidence interval for each of the plots. We found that the percentage lines up with what is specified in the theory for conformal prediction suggesting that we have met the underlying assumptions for these methods. **Key Finding 3** Conformal Prediction provides actual bounds on the occurrence of mispredictions.

E. Major Result 4: Cheaper to Train

Prior approaches have generally used all available data to train the model. However, with the introduction of out-of-sample prediction, we can now potentially train on a subset of fields from an application saving time gathering the training predictors to save time producing a model. However, this immediately begs the question of how to determine which data to train on. We can begin with a small sample from each field, and from that, we can look at the similarity between fields as estimated using our predictors. From this, we can drive a methodology to determine the order to explore the fields more completely. We present one such table in Table III. Our proposal involves assessing the similarity between two distinct fields by evaluating the spatial smoothness within the context of predicting CR. Previous analyses have consistently shown that compressibility is intrinsically linked to spatial smoothness, such as [3], [26]. Hence, we suggest a practical solution that involves examining the relative decay of singular values

of the covariance matrix of the blocks $X^b, b = 1, 2, \ldots, B$ (denoted as Σ in Section IV-A) for the 2D slices of the data.

To establish a comprehensive measure of spatial smoothness for the two fields, we propose employing the Mahalanobis distance. This distance metric quantifies the dissimilarity between the distributions of relative decay in singular values for the fields under consideration. By employing this approach, we effectively capture the variability in spatial smoothness between the two fields using a concise, single metric. Furthermore, it facilitates the selection of appropriate training fields for precise estimation of compressibility in out-of-sample scenarios.

We consider the impact of doing this filtering by field in Figure 5. We start by defining a field of interest – we will not include the field of interest in the training set. We add the fields to the training process in order by the most similar fields to the field of interest. Generally, we observe that as we add fields, we can see that the median absolute percentage error and the uncertainty generally improve as the number of fields increases according to the order that we propose (with the minor exceptions of QRain, QGraup, Qsnow, Precip, however in these cases the MedAPE was already small in these cases), and the 10% and 90% MedAPE tightens as the number of fields increases.

Let us consider the case of training a predictor to provide coverage for fields CLOUD, QCLOUD, PRECIP, QGRAUP, QRAIN, QSNOW, QICE, TC, and V from the Hurricane dataset to an accuracy of least 8%. A field is covered if we sample that field, or we sample the set of fields that get at least 8% accuracy without field prediction using our selection criteria above. Using an open source SAT solver, our method identifies a minimal training set of CLOUD, QCLOUD, QGRAUP, QSNOW, and TC in less than a millisecond[4]. Even without leveraging this approach, we are 1.42× faster to train than [3] due to the speed improvement of our metrics. **Key Finding 4** Leveraging our approach to limit the volume of training data needed to obtain coverage results in a speedup of 2.56× relative to training a separate model for each field.

F. Major Result 5: Lightweight for run-time performance

In this section, we empirically test the speedup for two use cases we modeled in the previous section. These results were run on a slice of the Hurricane CLOUD dataset. Figure 7 shows the speedup using predictions vs not using predictions for 5 compressors and 4 prediction methods for 50 search iterations. We see that depending on the speed of the compressors, and their consistency each method can achieve some can get speedups for each compressor. However, unlike the other methods, we show speedups for this use case for each compressor considered whereas other methods demonstrate some slowdowns relative to not using predictions. Also, important is the accuracy – only results from Underwood and our method were sufficiently accurate to get good estimates of the tuning

[4]Most applications use fewer than 13 fields thus SAT is applicable, For large numbers of fields, we use a greedy algorithm that provides a 2-approximation of SAT that runs in O(N). Regardless, the ordering determined by SAT is not used in estimation, and is not on the critical path.

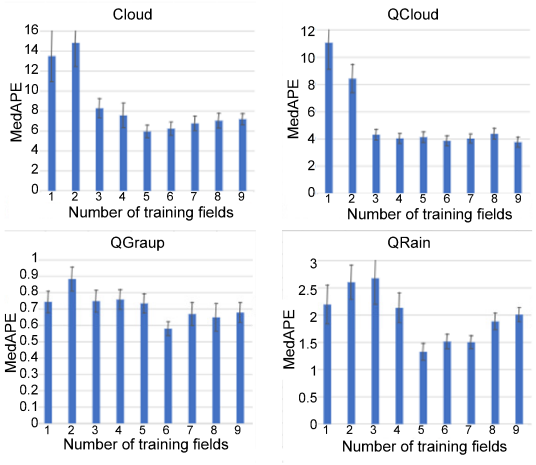
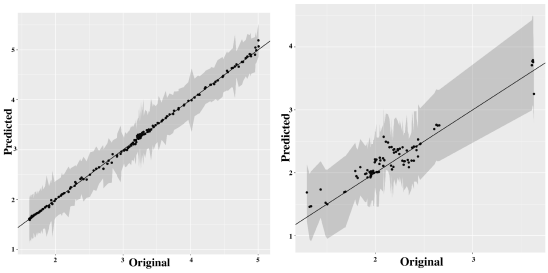
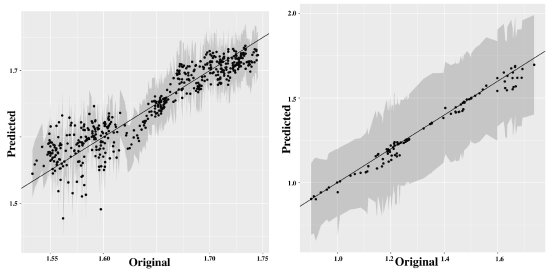


Fig. 5: multi-field training accuracy for hurricane



(a) Miranda VZ in-sample (b) Hurricane CLOUD out-of-sample



(c) NYX-Baryon in-sample (d) Hurricane PRECIP out-of-sample

Fig. 6: In-Sample and Out-of-Sample Predicted vs Actual using Conformal Prediction

of the compressors. Tao [6] gets a speedup of .55, Underwood [3], 2.13, and our approach gets 3.05. We exclude the results from Lu [22] here because Lu's approach cannot predict the compression ratio for non-SZ2 and ZFP compressors. **Key Finding 5** We achieve a roughly 3× performance improvement relative to not using estimates and improvement over the next best method [3].

G. Major Result 6: Speedup Analysis

For use case A, accuracy is of paramount importance because of the exponential degradation in the accuracy of the search. For this approach, dataset-specific but error-bound agnostic predictors can be somewhat costly compared to the cost of a single compressor invocation, provided that the error-bound specific predictors are fast. For this use case, it is

	Cloud	QCloud	Precip	QGraup	QRain	QSnow	QIce	TC	U	V	W	QVapor
Cloud	8.9	39.0	46.3	24.7	34.4	24.5	11.1	44.8	79.7	190.8	129.4	8600.3
QCloud	39.0	8.8	35.0	13.1	17.2	17.5	49.7	43.8	59.6	851.3	28.7	76238.9
Precip	46.3	35.0	8.9	39.8	61.7	45.3	67.1	42.5	59.3	838.7	57.8	78542.2
QGraup	24.7	13.1	39.8	8.9	13.7	16.2	30.7	43.4	68.3	424.3	32.8	32865.2
QRain	34.4	17.2	61.7	13.7	8.9	19.7	39.5	51.2	71.9	458.4	31.1	36086.1
QSnow	24.5	17.5	45.3	16.2	19.7	8.9	28.6	54.4	84.5	372.7	34.4	26429.4
QIce	11.1	49.7	67.1	30.7	39.5	28.6	8.9	48.8	78.0	187.4	166.8	8081.2
TC	44.8	43.8	42.5	43.4	51.2	54.4	48.8	8.9	20.4	211.8	72.5	21712.6
U	79.7	59.6	59.3	68.3	71.9	84.5	78.0	20.4	8.9	221.5	87.2	24025.3
V	190.8	851.3	838.7	424.3	458.4	372.7	187.4	211.8	221.5	8.9	3386.6	2346.2
W	129.4	28.7	57.8	32.8	31.1	34.4	166.8	72.5	87.2	3386.6	8.9	326839.3
QVapor	8600.3	76238.9	78542.2	32865.2	36086.1	26429.4	8081.2	21712.6	24025.3	2346.2	326839.3	8.2

TABLE III: Field Similarity for the Hurricane Dataset

also possible to achieve a speedup if the error-bound specific calculations are approximately the same cost as the compressor if they offer more consistent timing than a compressor – which is true of all of the predictors and compressors we tried. For example, suppose that a compressor, dataset predictors, and error bounds specific predictors all had a runtime of mean and standard deviation of 1, but the error bound specific predictors had a standard deviation of .33, a speedup of 2.56× is possible over 100,000 search iterations on 40 processors like was used in [1] to find configurations that satisfied climate codes.

For use case B, accuracy is of moderate importance. It still needs to be accurate, but how accurate depends on the average relative difference in the compression ratios. For competitive compressors for which the compression ratios are similar, little estimation error can be tolerated, but when there are larger gaps between compressors on specific datasets – which intuitively happens when the patterns in the data correspond to the compression principles of a particular compressor as opposed to others – the noise from predictions can be much higher as seen in [6]. For this use case, the performance of the dataset and error-bound specific predictors can also be somewhat high compared to the runtime of any one compressor and still achieve a speedup.

For use case C, accuracy is of mild importance because mispredictions are compensated for by the algorithm. Additionally, the user can over-allocate storage relative to the prediction to decrease the possibility of under-allocation. With this, we can determine a factor α which corresponds to the percentage of mispredictions on the dataset. With our approach based on conformal prediction, we can easily choose this parameter and determine a priori our space vs speed tradeoffs relative to a traditional approach. What is interesting about use case C, is that for the serial case, there is a maximal speedup of ≈ 2, but in the parallel case, higher speedups are possible because of the overhead of the parallel reduction. **Key Finding 6** Our novel modeling gives us insight into how much improvements in estimation accuracy or speed can affect speedup for various use cases.

VII. Conclusions and Future Work

In this paper, we demonstrated both empirical and analysis that show that fast and accurate compression ratio estimates can accelerate a variety of real-world compression use cases in parallel. We additionally advance the state of the art in compression ratio estimation by introducing a black box method that is accurate, robust, cheaper to train, and lightweight. For future work, we want to take this foundation to other use cases of compression ratio estimation.

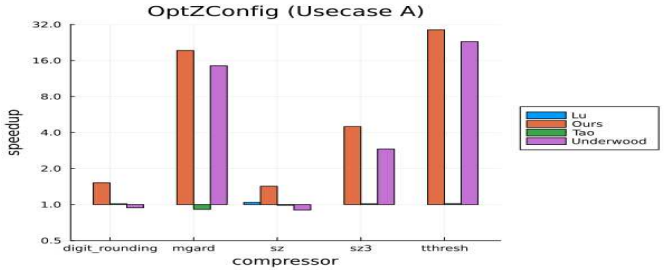


Fig. 7: Speedup for use case A. Lu only supports SZ so is excluded for other compressors

Acknowledgments

This research was supported by the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of two DOE organizations – the Office of Science and the National Nuclear Security Administration. The material was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research (ASCR), under contract DE-AC02-06CH11357, and supported by the National Science Foundation under Grant OAC-2003709 and OAC-2104023. We acknowledge the computing resources provided on Bebop (operated by Laboratory Computing Resource Center at Argonne) and on Theta and JLSE (operated by Argonne Leadership Computing Facility).

References

[1] R. Underwood, J. C. Calhoun, S. Di, A. Apon, and F. Cappello, "OptZConfig: Efficient Parallel Optimization of Lossy Compression Configuration," *IEEE Transactions on Parallel and Distributed Systems*, pp. 1–1, 2022, conference Name: IEEE Transactions on Parallel and Distributed Systems.

[2] M. H. Rahman, S. Di, K. Zhao, R. Underwood, L. Guanpeng, and F. Cappello, "A Feature-Driven Fixed-Ratio Lossy Compression Framework for Real-World Scientific Datasets." Anaheim, California: IEEE Computer Society, Apr. 2023.

[3] R. Underwood, J. Bessac, D. Krasowska, J. C. Calhoun, S. Di, and F. Cappello, "Black-Box Statistical Prediction of Lossy Compression Ratios for Scientific Data," *ARXIV*, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2305.08801

[4] S. Jin, D. Tao, H. Tang, S. Di, S. Byna, Z. Lukic, and F. Cappello, "Accelerating Parallel Write via Deeply Integrating Predictive Lossy Compression with HDF5," Jun. 2022, arXiv:2206.14761 [cs]. [Online]. Available: http://arxiv.org/abs/2206.14761

[5] S. Jin, S. Di, J. Tian, S. Byna, D. Tao, and F. Cappello, "Improving Prediction-Based Lossy Compression Dramatically via Ratio-Quality Modeling," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, May 2022, pp. 2494–2507, iSSN: 2375-026X.

[6] D. Tao, S. Di, X. Liang, Z. Chen, and F. Cappello, "Optimizing Lossy Compression Rate-Distortion from Automatic Online Selection between SZ and ZFP," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 8, pp. 1857–1871, Aug. 2019, number: 8 Citation Key Alias: taoOptimizingLossyCompression2019.

[7] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948. [Online]. Available: http://plan9.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf

[8] P. Lindstrom, "Fixed-Rate Compressed Floating-Point Arrays," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2674–2683, Dec. 2014, number: 12.

[9] D. Tao, S. Di, Z. Chen, and F. Cappello, "Significantly Improving Lossy Compression for Scientific Data Sets Based on Multidimensional Prediction and Error-Controlled Quantization," in *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2017, pp. 1129–1139.

[10] 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)*, Apr. 2021, pp. 1643–1654, iSSN: 2375-026X.

[11] C. S. Zender, "Bit Grooming: statistically accurate precision-preserving quantization with compression, evaluated in the netCDF Operators (NCO, v4.4.8+)," *Geoscientific Model Development*, vol. 9, no. 9, pp. 3199–3211, Sep. 2016, number: 9. [Online]. Available: https://gmd.copernicus.org/articles/9/3199/2016/

[12] X. Delaunay, A. Courtois, and F. Gouillon, "Evaluation of lossless and lossy algorithms for the compression of scientific datasets in NetCDF-4 or HDF5 formatted files," Numerical Methods, preprint, Nov. 2018. [Online]. Available: https://gmd.copernicus.org/preprints/gmd-2018-250/gmd-2018-250.pdf

[13] M. Ainsworth, O. Tugluk, B. Whitney, and S. Klasky, "Multilevel techniques for compression and reduction of scientific data—the univariate case," *Computing and Visualization in Science*, vol. 19, no. 5-6, pp. 65–76, Dec. 2018, number: 5-6. [Online]. Available: http://link.springer.com/10.1007/s00791-018-00303-9

[14] ——, "Multilevel Techniques for Compression and Reduction of Scientific Data-Quantitative Control of Accuracy in Derived Quantities," *SIAM Journal on Scientific Computing*, vol. 41, no. 4, pp. A2146–A2171, Jan. 2019, number: 4. [Online]. Available: https://epubs.siam.org/doi/10.1137/18M1208885

[15] ——, "Multilevel Techniques for Compression and Reduction of Scientific Data—The Multivariate Case," *SIAM Journal on Scientific Computing*, vol. 41, no. 2, pp. A1278–A1303, Jan. 2019, number: 2 tex.ids: ainsworthMultilevelTechniquesCompression2019. [Online]. Available: https://epubs.siam.org/doi/10.1137/18M1166651

[16] J. Lee, Q. Gong, J. Choi, T. Banerjee, S. Klasky, S. Ranka, and A. Rangarajan, "Error-Bounded Learned Scientific Data Compression with Preservation of Derived Quantities," *Applied Sciences*, vol. 12, no. 13, p. 6718, Jul. 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/13/6718

[17] R. Ballester-Ripoll, P. Lindstrom, and R. Pajarola, "TTHRESH: Tensor Compression for Multidimensional Visual Data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 9, pp. 2891–2903, Sep. 2020, number: 9 Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[18] S. Li and J. Clyne, "Lossy Scientific Data Compression With SPERR," Copernicus Meetings, Tech. Rep. EGU22-946, Mar. 2022, conference Name: EGU22. [Online]. Available: https://meetingorganizer.copernicus.org/EGU22/EGU22-946.html

[19] R. M. Gray and T. Hashimoto, "Rate-Distortion Functions for Non-stationary Gaussian Autoregressive Processes," in *Data Compression Conference (dcc 2008)*, Mar. 2008, pp. 53–62, iSSN: 2375-0359.

[20] J. Gibson, "Rate Distortion Functions and Rate Distortion Function Lower Bounds for Real-World Sources," *Entropy*, vol. 19, no. 11, p. 604, Nov. 2017. [Online]. Available: http://www.mdpi.com/1099-4300/19/11/604

[21] E. Lei, H. Hassani, and S. S. Bidokhti, "Neural Estimation of the Rate-Distortion Function for Massive Datasets," in *2022 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2022, pp. 608–613, iSSN: 2157-8117.

[22] T. Lu, Q. Liu, X. He, H. Luo, E. Suchyta, J. Choi, N. Podhorszki, S. Klasky, M. Wolf, T. Liu, and Z. Qiao, "Understanding and Modeling Lossy Compression Schemes on HPC Scientific Data," in *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. Vancouver, BC: IEEE, May 2018, pp. 348–357. [Online]. Available: https://ieeexplore.ieee.org/document/8425188/

[23] J. Wang, Q. Chen, T. Liu, Q. Liu, and X. He, "Zperf: A Statistical Gray-Box Approach to Performance Modeling and Extrapolation for Scientific Lossy Compression," *IEEE Transactions on Computers*, pp. 1–14, 2023, conference Name: IEEE Transactions on Computers.

[24] Z. Qin, J. Wang, Q. Liu, J. Chen, D. Pugmire, N. Podhorszki, and S. Klasky, "Estimating Lossy Compressibility of Scientific Data Using Deep Neural Networks," *IEEE Letters of the Computer Society*, vol. 3, no. 1, pp. 5–8, Jan. 2020, number: 1 Conference Name: IEEE Letters of the Computer Society.

[25] C. Wang and H. Zhao, "Spatial heterogeneity analysis: Introducing a new form of spatial entropy," *Entropy*, vol. 20, no. 6, p. 398, 2018.

[26] V. Goyal, "Theoretical foundations of transform coding," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 9–21, 2001.

[27] V. K. Goyal, J. Zhuang, and M. Veiterli, "Transform coding with backward adaptive updates," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1623–1633, 2000.

[28] P. Lindstrom and M. Isenburg, "Fast and Efficient Compression of Floating-Point Data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1245–1250, Sep. 2006, number: 5 Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[29] J.-H. Liu, M.-F. Tsai, L. Chen, and C. C.-P. Chen, "Accurate and analytical statistical spatial correlation modeling based on singular value decomposition for vlsi dfm applications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 4, pp. 580–589, 2010.

[30] A. E. Lamont, J. K. Vermunt, and M. L. Van Horn, "Regression mixture models: Does modeling the covariance between independent variables and latent classes improve the results?" *Multivariate Behav Res*, vol. 51, no. 1, pp. 35–52, 2016.

[31] P. Toccaceli, "Introduction to conformal predictors," *Pattern Recognition*, vol. 124, p. 108507, 2022.

[32] A. Solari and V. Djordjilović, "Multi split conformal prediction," *Statistics & Probability Letters*, vol. 184, p. 109395, 2022.

[33] M. Fontana, G. Zeni, and S. Vantini, "Conformal prediction: a unified review of theory and new challenges," *Bernoulli*, vol. 29, no. 1, pp. 1–23, 2023.

[34] G. Elfving, "The Asymptotical Distribution of Range in Samples from a Normal Population," *Biometrika*, vol. 34, no. 1/2, pp. 111–119, Jan. 1947. [Online]. Available: https://www.jstor.org/stable/2332515

[35] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM Journal on Applied Mathematics*, vol. 17, no. 2, p. 416–429, Mar 1969.

[36] R. Underwood, V. Malvoso, J. C. Calhoun, S. Di, and F. Cappello, "Productive and Performant Generic Lossy Data Compression with LibPressio," in *2021 7th International Workshop on Data Analysis and Reduction for Big Scientific Data (DRBSD-7)*, Nov. 2021, pp. 1–10.

[37] K. Zhao, S. Di, X. Lian, S. Li, D. Tao, J. Bessac, Z. Chen, and F. Cappello, "SDRBench: Scientific Data Reduction Benchmark for Lossy Compressors," in *2020 IEEE International Conference on Big Data (Big Data)*, Dec. 2020, pp. 2716–2724.

[38] D. Krasowska, J. Bessac, R. Underwood, J. C. Calhoun, S. Di, and F. Cappello, "Exploring Lossy Compressibility through Statistical Correlations of Scientific Datasets," in *2021 7th International Workshop on Data Analysis and Reduction for Big Scientific Data (DRBSD-7)*, Nov. 2021, pp. 47–53.