https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

ClimateBenchPress (v1.0): A Benchmark for Lossy Compression of Climate Data

Tim Reichelt[1], Juniper Tyree[2], Milan Klöwer[1], Peter Dueben[3], Bryan N. Lawrence[4,5], Allison H. Baker[6], Sara Faghih-Naini[3], Torsten Hoefler[7], and Philip Stier[1]

[1]University of Oxford, Oxford, United Kingdom
[2]INAR, University of Helsinki, Helsinki, Finland
[3]European Centre for Medium-Range Weather Forecasts, Reading, United Kingdom
[4]Department of Meteorology, University of Reading, Reading, UK
[5]National Centre for Atmospheric Science (NCAS), UK
[6]The NSF National Center for Atmospheric Research, Boulder, CO, United States of America
[7]ETH Zurich, Zurich, Switzerland

Correspondence: Tim Reichelt (tim.reichelt@physics.ox.ac.uk)

Abstract. The rapidly growing volume of weather and climate data, both from models and observations, is increasing the pressure on data centers, restricting scientific analysis, and data distribution. For example, kilometre-scale climate models can generate petabytes of data per simulated month, making it generally infeasible to store all output. To address this challenge, numerous novel compression techniques have been proposed to ease data storage requirements. However, there exist no well-defined benchmarks for rigorously evaluating and comparing the performance of these compressors, including their impact on the data's properties. The lack of benchmarks makes it difficult to design and standardize compressors for weather and climate data, and for scientists to trust that compression errors have no significant impact on their analysis. Here, we address this gap by presenting ClimateBenchPress, a benchmark suite for lossy compression of climate data, which defines both data sets and evaluation techniques. The benchmark covers climate variables following various statistical distributions at medium to very high resolution in time and space, from both numerical models and satellite observations. To ensure a fair comparison between different compressors, each variable comes with a set of maximum error bound checks that the lossy compressors need to pass. By evaluating an initial set of baseline compressors on the benchmark, we gather practical insights for effective application of lossy compression. Our benchmark is open source and extensible: users can easily add new compressors, data sources, and evaluation metrics depending on their own specific use cases.

1 Introduction

Datasets quantifying Earth's weather and climate are rapidly growing in size due to the advent of next-generation km-scale Earth system models (ESMs) (Stevens et al., 2019; Bauer et al., 2021; Rackow et al., 2025; Segura et al., 2025) and an expanding suite of satellite missions monitoring the Earth's atmospheric state (Rossow and Schiffer, 1991; Group and Program, 2017; Stephens et al., 2002; Illingworth et al., 2015). These datasets can easily reach petabyte scale (Bauer et al., 2021),

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

requiring very large data centers with significant energy consumption and slowing down data analysis and distribution. Hence, there is an urgent need for novel compression algorithms that can handle the unique characteristics of climate data.

Lossless compressors cannot provide sufficiently large compression ratios due to the high entropy in the mantissa of the floating-point representation of the data (Klöwer et al., 2021), limiting compressibility following Shannon's source coding theorem (Shannon, 1948). However, climate data comes with inherent uncertainty associated with each data point owing to measurement, model, and simulation errors that vary in space, time, and for different variables. Additionally, the precision required from the input data varies widely depending on the downstream analysis a user might want to run. As an example, computing the expected average global mean temperature rise from CMIP6 simulations requires a lower level of precision than computing the expected local power generation of wind farms from high-resolution storm-resolving models. Therefore, in climate data there is generally a *tolerable compression error*, though defining the exact error tolerances for all data and downstream analyses remains an open problem.

The existence of a tolerable compression error motivates the usage of *lossy compression* algorithms. As the name implies, in contrast to lossless compression, lossy compression no longer guarantees that the original input and the reconstructed output after decompression are identical. However, by allowing for a non-zero reconstruction error, lossy compression is generally able to achieve significantly larger compression ratios compared to lossless compression (Baker et al., 2016; Underwood et al., 2022). There exists already a wide body of work on developing lossy compressors tailored to scientific datasets (Ballester-Ripoll et al., 2019; Underwood et al., 2022; Liang et al., 2022; Lindstrom, 2014; Silver and Zender, 2017) and an emerging field of work towards developing *neural compression* schemes for climate data (Huang and Hoefler, 2023; Han et al., 2024; Mirowski et al., 2024).[1]

However, at the time of writing, no well-established and easily accessible benchmark exists that is focused specifically on the task of compressing climate and weather data. The issue is complicated by the fact that compression algorithm developers tend not to be experts in climate science, and although abundant data exist, much of it is either difficult to access or stored in formats unfamiliar to researchers outside of climate science. This leads a majority of research to focus on evaluating compressors on datasets derived from a single model or data source, as can be seen in Figure 1. Without questioning the importance of any one of these datasets, it is important to validate compressors across a variety of data sources, i.e. data from different models and observational sources and with different variables, because each data source poses different challenges for compression due to differing resolution in space and time, statistical distributions of variables, variable characteristics, associated uncertainty, and scientific objectives. The differences between individual variables can be so stark that they might require different compression approaches altogether (Baker et al., 2017). However, as a sufficiently general compressor should be able to deal with the diversity of variable characteristics found in climate datasets, it is important that a compression benchmark adequately captures this diversity.

[1] In contrast to "traditional" compressors, which use expert-crafted transformations and heuristics to compress the data, neural compression approaches learn compressors from the data itself, sometimes based on large pre-training datasets and sometimes based on overfitting onto the specific data that is about to be compressed.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

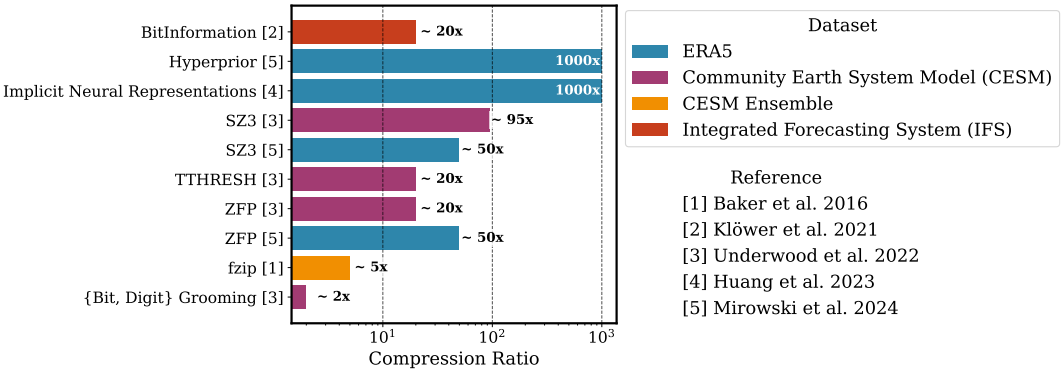

Figure 1. Compression ratios (uncompressed size/compressed size) of different compressors applied to climate data reported in the literature. There are large variations in the reported compression ratios due to differences in: the data sources and variables that are being compressed; the exact techniques used to compute the compression ratio; and the error tolerances. Overall, these differences make it difficult to directly compare the reported compression ratios.

Surveying the literature on lossy compression for climate data, we find that the lack of a common benchmark makes it difficult to rigorously compare the performance of different compression algorithms. Figure 1 provides an overview of the different compression ratios reported in published works. Even for the same compressor, different studies may report drastically varying compression ratios because different datasets, different error metrics, or very different levels of a tolerable compression error are being used. Even the way that the compression ratio is measured can change between authors. Some directly measure the size reduction in the binary data of the data fields, while others measure the difference in the final file size on disk, making the compression ratio dependent on the used file format.

To provide a standardized framework for comparing compression algorithms, we therefore present **ClimateBenchPress**, a novel **climate** data **bench**mark for com**press**ion algorithms. The benchmark contains data on structured grids from a variety of input sources covering the land, ocean and atmospheric variables of the Earth system at different spatial and temporal resolutions. Crucially, the benchmark comes with a pre-defined list of error bounds for each input variable that provides a guideline to make compression performance comparable. These error bounds are calculated with an automated algorithm and validated against bounds provided by experts (Tyree et al., 2026). The code for data processing and evaluation procedures is fully open source and easily extensible, allowing users to add new data sources, compressors, or evaluation metrics.

In summary, our contributions are:

- Introducing ClimateBenchPress, a standardized open-source benchmark to evaluate the performance of lossy compression algorithms on climate datasets, available at https://github.com/ClimateBenchPress.
- Evaluating an initial set of baseline compressors on ClimateBenchPress to provide a reference for the compression performance that downstream users can expect in practice.
- Discussing the trade-offs between different compressors, providing actionable insights for compression users.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Table 1. Description of datasets contained in ClimateBenchPress. The total size of the evaluation set is around 2.95 GB at the moment. Dimensions are given as time (T), longitude ($N_{(\text{lon})}$), latitude ($N_{(\text{lat})}$), vertical levels (L), and variables (V). The data source abbreviations are: Integrated Forecasting System (IFS), Climate Model Intercomparison Project Phase 6 (CMIP6), Next Generation Earth Modelling Systems Project (NextGEMS), Copernicus Atmospheric Monitoring Service (CAMS), European Space Agency's Climate Change Initiative (ESA CCI; observation).

Variable(s)	**Data Source**	**Dimensions** ($T \times N_{(\text{lon})} \times N_{(\text{lat})} \times L \times V$)	**Time Resolution**	**Data Type**	**Data Size**
Sea-level Pressure, North- and Eastward 10m Wind	IFS	$21 \times 1440 \times 721 \times 1 \times 3$	3-Hourly	float64	523.27 MB
Humidity	IFS	$1 \times 1440 \times 721 \times 137 \times 1$	N/A	float64	1137.91 MB
Air Temperature	CMIP6	$12 \times 192 \times 144 \times 19 \times 1$	Monthly	float32	25.21 MB
Sea Surface Temperature	CMIP6	$12 \times 360 \times 300 \times 1 \times 1$	Monthly	float32	5.18 MB
Precipitation, Outgoing Longwave Radiation	NextGEMS	$8 \times 7200 \times 3600 \times 1 \times 2$	3-Hourly	float32	265.42 MB
Nitrogen Dioxide	CAMS	$8 \times 900 \times 451 \times 25 \times 1$	3-Hourly	float32	324.72 MB
Biomass	ESA CCI	$1 \times 405000 \times 157500 \times 1 \times 1$	Yearly	float32	672.69 MB

2 A Benchmark Dataset for Evaluating Climate Model Compressors

2.1 Compression Tasks

The goal of the benchmark is to provide data that comes in the form of multi-dimensional arrays $X \in \mathbb{R}^{T \times N_{(\text{lon})} \times N_{(\text{lat})} \times L \times V}$ with T giving the number of time steps, $N_{(\text{lat})}$ the number of latitude coordinates, $N_{(\text{lon})}$ the number of longitude coordinates, L the number of vertical levels, and V the number of variables/fields. Crucially, $T, N_{(\text{lat})}, N_{(\text{lon})}, L$ and V are all allowed to vary and not assumed constant. Hence, compression algorithms for this benchmark need to be able to deal with multiple grid resolutions in space, time, and number of variables. *Lossy compression* algorithms encode the data into a compressed representation such that, after decoding, the reconstructed array $\hat{X}$ is an approximation to the original input, i.e. $X \approx \hat{X}$.

For the purpose of this benchmark, we made the simplifying assumption that all data is defined on regular longitude-latitude grids. Of course, there are many data sources which do not natively lie on a regular grid, with many ESMs using their own unstructured grids. Regridding the data to a regular grid before passing it to a compressor is a common choice to deal with unstructured grids, hence measuring the performance of lossy compressors on regular grids still provides a useful signal for performance on unstructured grids. Moreover, many scientific compressors implicitly or explicitly assume a regular grid and therefore perform better if the input is transformed to a regular grid. However, the regridding from unstructured to regular grids is often a lossy process in itself, with the exact behavior depending on the details of the source and target grids, the regridding algorithms, and the variables that are being regridded. How to generalize compressors to arbitrary unstructured grids is still an

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

open research problem, and outside the scope of this benchmark, as the additional complexity conflicts with the comparability we want to achieve.[2]

To keep the results of the benchmark easily interpretable, we only include a small representative subset of all available variables in climate data sets in this benchmark. Furthermore, we restrict the overall size of the benchmark dataset to relatively small subsets of data so that it can be easily downloaded onto a researcher's laptop. When selecting the input datasets and variables for our benchmark, we ensured that they cover the various dimensions of the climate data compression problem, including: varying spatial resolution (100 m to 150 km); varying temporal resolution (hourly to monthly); variables with a single vertical layer and variables with multiple vertical layers; data generated from a diversity of numerical models and from observations; different statistical distributions induced by different variables; different aspects of the Earth system, including land, ocean, and atmosphere variables. The specific datasets and variables in the benchmark, chosen to span the aforementioned dimensionalities of the compression problem, are:

- **Sea-level Pressure, 10m Wind Vector, and Humidity** from the Integrated Forecasting System (IFS). Crucially, we use uncompressed IEEE-754 Standard 64-bit floating point outputs.[3]. The IFS model underpins the popular ERA5 reanalysis dataset (Hersbach et al., 2020) but the data distributed through the Copernicus Climate Change Service (C3S) Climate Data Store has already been compressed with linear packing.

- **Air Temperature** from the Coupled Model Intercomparison Project Phase 6 (CMIP6) (Eyring et al., 2016) as 4-dimensional datasets including the vertical and time dimension;

- **Sea Surface Temperature (SST)** fields from CMIP6 contain large regions of NaN values, due to SSTs being undefined on land, providing a significant challenge to compression algorithms.

- **Precipitation and Outgoing Longwave Radiation (OLR)** fields from the Next Generation Earth Modelling Systems (NextGEMS) project (Koldunov et al., 2023; Wieners et al., 2024; Segura et al., 2025) contain data from high-resolution km-scale climate models. The OLR—heavily impacted by the presence of clouds—and precipitation fields provide a test of whether compression algorithms can preserve high-frequency information and zero values in the input.

- **Nitrogen Dioxide** reanalysis data from the Copernicus Atmospheric Monitoring Service (CAMS) includes anthropogenic sources from industrial activity and transportation in the form of visual artifacts such as tracks and point emissions (Inness et al., 2019). We use CAMS data that is stored in 32-bit floating point precision and has not undergone any other compression.[4]

- **Above-Ground Biomass (AGB)** (Santoro et al., 2024) as captured by the European Space Agency's Climate Change Initiative is a measure of the total weight of all living trees in a unit area, which is derived from a multitude of satellite

[2]Similarly, there exist many relevant data modalities that do not lie on a structured grid, e.g. weather station data, sub level-3 satellite data, or measurements from weather balloons, designing compressors for these modalities can require fundamentally different design decisions, which is why we exclude them here.

[3]https://apps.ecmwf.int/ifs-experiments/rd/hplp/

[4]https://apps.ecmwf.int/ifs-experiments/rd/hej6/

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

observations and provided at a resolution of 100m, is relevant to climate modeling, as forests are a major source and sink of atmospheric CO_2.

Table 1 succinctly summarizes the different data sources and their dimensionalities, and Appendix A contains visualizations of the different data sources.

2.1.1 Error Bounds

Compression inherently involves trade-offs between different performance metrics: compression factor (input data size divided by encoded size), speed, and error. If we are willing to tolerate a high error, we are able to compress more and vice versa. For example, just replacing all the input data in an array with their mean will lead to a high compression factor—we only need to store a single floating point number, the mean, instead of the entire array—but will incur a large error. This is also known in the compression literature as the *rate-distortion trade-off* (Thomas and Joy, 2006).

Therefore, in designing our benchmark it is important to specify error bounds for each variable that the compressors should satisfy. Otherwise, compressors can report large compression ratios by allowing for large errors. Unfortunately, there are no well-defined, agreed upon, standard error bounds for each variable. This is also partly due to the fact that the level of compression error that is tolerable is inherently application dependent, as mentioned in the introduction. Consequently, we do not select a single error bound but rather a set of three error bounds, which allows us to explore the Pareto front of optimal compressors, i.e. to understand whether there are some compressors which are able to achieve high compression ratios for low error bounds and others that produce high compression ratios for larger error bounds.

Additionally, we distinguish between *pointwise absolute* and *relative* error bounds.

Definition 1. *Let $\mathbb{F}_p$ denote the set of numbers representable with precision p in the IEEE 754 Standard for Floating-Point Arithmetic. For input data $X \in \mathbb{F}_p^{T\times N_{(\text{lon})}\times N_{(\text{lat})}\times L\times V}$, decompressed output $\hat{X} \in \mathbb{F}_p^{T\times N_{(\text{lon})}\times N_{(\text{lat})}\times L\times V}$, and a variable index $0 \le v \le V$, let $X_{i,v}$ and $\hat{X}_{i,v}$ denote the entries corresponding to the ith element obtained by flattening the first four dimensions (time, longitude, latitude, level) of X and $\hat{X}$ onto a single axis. Then*

1. *the* pointwise absolute error bound $b_{abs} > 0$ *is satisfied if* $|X_{i,v} - \hat{X}_{i,v}| \le b_{abs}$ *or* $X_{i,v} \equiv \hat{X}_{i,v}$,[5]
2. *the* pointwise relative error bound $b_{rel} > 0$ *is satisfied if* $|X_{i,v} - \hat{X}_{i,v}| \le b_{rel} \cdot |X_{i,v}|$ *or* $X_{i,v} \equiv \hat{X}_{i,v}$,

for all entries $i \in \{1, \dots, M\}$ with $M = T \cdot N_{(\text{lon})} \cdot N_{(\text{lat})} \cdot L$ denoting the total number of entries.

Relative error bounds are useful because absolute error bounds are not necessarily suitable for fields that vary largely in their magnitude, as e.g. many tracer concentrations. Notably, this specific formulation of the relative error bound also ensures it is defined for $X_i = 0$, compared to the alternative formulation of the relative error bound in terms of $|X_i - \hat{X}_i|/|X_i|$.

The type of bound we use for each variable is listed in Table 2. By default, we select an absolute error bound for each variable, but we resort to relative error bounds for the biomass, nitrogen dioxide, and precipitation variables, because these

[5]The equivalence condition ensures that the bound is still satisfied for matching infinity and NaN values. For example, without this condition the case $X_{i,v} = \infty$ and $\hat{X}_{i,v} = \infty$ would violate the bound because $\infty - \infty =$ NaN under IEEE 754.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Table 2. Individual error bounds for different input variables in ClimateBenchPress. Definition 1 describes the meaning of absolute and relative error bound, and Equations (1) and (2) define how the three different error bounds $b^{(\text{low})}, b^{(\text{mid})}$, and $b^{(\text{high})}$ are computed in practice. "Non-Zero Range" indicates the smallest and largest non-zero absolute value, and the "Expert" column indicates bounds obtained by consulting domain scientists at ECMWF on what the maximum tolerable error for the variable should be, entries marked with a (*) are average rather than pointwise error bounds.

Variable	Non-Zero Range	SI Unit	Type	$b^{(\text{low})}$	$b^{(\text{mid})}$	$b^{(\text{high})}$	Expert
10m u Wind Component	[3e−7, 29]	m s^{-1}	Absolute	0.0056	0.0620	0.1000	0.01 (*)
10m v Wind Component	[3.8e−7, 32]	m s^{-1}	Absolute	0.0039	0.0610	0.1000	0.01 (*)
Mean Sea Level Pressure	[9.5e4, 10.5e5]	Pa	Absolute	1.50	6.00	7.90	1.0
Specific Humidity	[1.3e−11, 2.6e−2]	kg kg^{-1}	Relative	0.17 %	0.94 %	1.42 %	1 %
Biomass	[1.0, 400]	Mg ha^{-1}	Relative	14.29 %	30.77 %	33.33 %	N/A
Nitrogen Dioxide	[1.7e−14, 4.6e−7]	kg kg^{-1}	Relative	2.44e−2 %	3.12 %	12.50 %	N/A
Precipitation Rate	[1.4e−45, 0.03]	kg m^{-2} s^{-1}	Relative	1.10 %	10.71 %	17.52 %	1 % (*)
Outgoing Longwave Radiation	[72, 352]	W m^{-2}	Absolute	0.0280	0.2100	0.4900	0.1 (*)
Air Temperature	[150, 310]	K	Absolute	0.0119	0.0740	0.1038	0.05
Sea Surface Temperature	[3.6e−5, 35]	°C	Absolute	6.10e−5	9.80e−5	1.80e−4	0.01

three variables have large variations in magnitude. Additionally, the biomass and precipitation fields contain many entries with 0s, and our definition of a relative error bound ensures that these values have to be preserved exactly.

To select the specific numerical value for each error bound, we leverage an automated mechanism (Tyree et al., 2026). Concretely, the mechanism relies on the uncertainty information derived from the official ensemble statistics published along with ERA5 (Hersbach et al., 2020), which are based on a reduced resolution 10-member ensemble that incorporates the uncertainty from the data assimilation process and the physics parameterizations. The ERA5 ensemble data provides an ensemble mean $\mu_{i,v}$ and ensemble spread $\sigma_{i,v}$ for variable v at grid point i, where the index i ranges over the time, longitude, latitude, and level dimensions.[6] Hence, $\{\sigma_{i,v}\}_{i=1}^{M}$ for $M = T \cdot N_{(\text{lon})} \cdot N_{(\text{lat})} \cdot L$ represents the collection of ensemble spread data for variable v over all time steps, latitude-longitude points, and a specific vertical level. For pressure-level (not single-level) variables, we first compute the error bounds for the 1000hPa, 850hPa, 500hPa, and 50hPa levels as below and then use the average error bound for this benchmark.

For each variable, we compute three error bounds, $b^{(\text{low})}, b^{(\text{mid})}$, and $b^{(\text{high})}$ that are based on the percentiles of the ensemble spread data $\{\sigma_{i,v}\}_{i=1}^{M}$. These three bounds are meant to capture a variety of use-cases, from high error tolerance (e.g. visualization) to low tolerance (e.g. computing a function over the data). To derive absolute bounds, we use the 100%, 99%, and 95% percentiles of the ensemble spreads $\sigma_{i,v}$, i.e.

$$b^{(\text{low})}_{\text{absolute}} = P_{100\%}(\{\sigma_{i,v}\}_{i=1}^{M}), \qquad b^{(\text{mid})}_{\text{absolute}} = P_{99\%}(\{\sigma_{i,v}\}_{i=1}^{M}), \qquad b^{(\text{high})}_{\text{absolute}} = P_{95\%}(\{\sigma_{i,v}\}_{i=1}^{M}). \tag{1}$$

[6] We use the ensemble mean and spread data as published on the Copernicus Climate Data Store (Hersbach et al., 2023) which contains ensemble data at 3-hourly intervals. To compute the bounds we leverage all the data available for June and December 2023.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Relative error bounds are calculated in a similar manner, with the only change that the ensemble spread values are normalized by the ensemble mean

$$b^{(\text{low})}_{\text{relative}} = P_{100\%}(\{\sigma_{i,v}/\mu_{i,v}\}_{i=1}^{M}), \qquad b^{(\text{mid})}_{\text{relative}} = P_{99\%}(\{\sigma_{i,v}/\mu_{i,v}\}_{i=1}^{M}), \qquad b^{(\text{high})}_{\text{relative}} = P_{95\%}(\{\sigma_{i,v}/\mu_{i,v}\}_{i=1}^{M}). \tag{2}$$

Note that we only compute the percentile over non-zero per-point error bounds. Intuitively, this process makes the heuristic assumption that variables with larger ensemble spreads require larger error bounds because there is more uncertainty in estimating their exact value.

The bounds for the sea surface temperature (ERA5 parameter name: `sst`), air temperature (`t`), outgoing longwave radiation (`avg_tnlwrf`), humidity (`q`), precipitation (`avg_tprate`), sea level pressure (`msl`), and wind vector variables (`u10`, `v10`) are all computed in this manner from the ERA5 ensemble data. For the biomass and nitrogen dioxide variable, there exists no matching ERA5 variable, and therefore, we revert to another mechanism to compute the error bounds. The biomass dataset also publishes standard deviation estimates, which represent the precision in the biomass point estimate, so we instead plug those standard deviations into Equation (2) to compute relative error bounds. For nitrogen dioxide, we revert to using the bounds published in Klöwer et al. (2021), which are computed using information theoretic analysis.

Table 2 lists the exact error bounds we use for each variable. *These bounds are not meant to represent general recommendations for which errors are tolerable or which error bounds should be used in practice. Instead, these bounds serve as a mechanism to enable a fair comparison of compression ratios. Users of lossy compression should always validate separately what error bound is acceptable for their specific use case.*

To check whether the derived bounds are realistic, Tyree et al. (2026) additionally consulted domain scientists at ECMWF. Provided with existing errors from standard GRIB compression, the scientists drew on their experience with ERA5 data to specify the maximal absolute or relative errors they would consider acceptable. To not bias their judgements, scientists did not get access to the computed error bounds from Table 2. With this process, we were able to obtain "expert bounds" for the sea level pressure, humidity, precipitation, outgoing longwave radiation, air temperature and sea surface temperature variables. In comparison to our computed bounds, the expert bounds mostly lie between $b^{(\text{low})}$ and $b^{(\text{high})}$, with the exception of the sea level pressure and precipitation variables. For both these variables, the lowest computed error bound, $b^{(\text{low})}$, is only slightly larger than the expert error bound. In general, the comparison to the expert bounds shows that our selected bounds are values that could realistically be picked in practice and are therefore suitable to be used as a base for comparing lossy compression algorithms.

2.2 Quantitative Evaluation Metrics

While the presented error bounds make the compression ratios between different compressors comparable, we still need a wider set of evaluation metrics to meaningfully compare their performance. As is standard in the compression literature, in this benchmark we will distinguish between two different **types of evaluation measures**: *distortion metrics* and *compression metrics*. Distortion metrics measure how much the reconstructed data differs from the original input, while compression metrics measure how much the data has been compressed and under what computational cost, e.g. encoding/decoding throughput. The

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

next two sections introduce the specific metrics we compute as part of our benchmark. Notably though, our benchmark is easily extensible and users can readily add their own metrics for their own use cases.

2.2.1 Distortion metrics

Any distortion metric will only capture certain aspects of information loss, e.g. the maximum error bounds provided to the compressors do not take into account any spatial correlations of the errors, which is why we need multiple metrics to develop a more holistic picture of compressor performance. We selected the metrics so that each metric measures a distinct qualitative aspect of the compression performance. We purposefully made an effort to restrict the total number of metrics to avoid information overload when analyzing the benchmark results. Additionally, the distortion metrics need to be easy to understand for climate scientists, to help them inform their decision on which compressor to use in practice. Overall, the distortion metrics in the benchmark are as follows.

Mean absolute error (MAE) defined as $\frac{1}{N}\sum_i |X_i - \hat{X}_i|$ provides a measure of the average pointwise reconstruction error incurred by the compressor. Other metrics that are averages of pointwise errors, such as the peak signal-to-noise ratio (PSNR), could be used as well but we chose the MAE because it allows us to measure the error in the same units as the input data.

Maximum absolute error (MaxAbsError) defined as $\max_i |X_i - \hat{X}_i|$, provides a mechanism to investigate whether the provided error bound is exceeded for any data point, or not.

Spectral Error defined as $\mathrm{SL}(X,\hat{X}) = \sum_l \Big(S(X,l) - S(\hat{X},l)\Big)^2$ where $S(X,l)$ denotes the radially averaged power spectral density at wavelength l. This metric allows us to check whether the compressors preserve high-frequency signal information.

Data Structural Similarity Index Measure (DSSIM) (Baker et al., 2023) aims to measure the visual similarity between the compressed and uncompressed data as it would be perceived by the human eye. Compared to pointwise error metrics it also takes into account the spatial correlations of the data. In particular, DSSIM is an adaption of the SSIM measure, which is widely used in the compression literature, to scientific floating point data.

2.2.2 Compression Metrics

The standard measure to compare compression performance is the **compression ratio**, which is computed by dividing the size of the input data by the size of the encoding; hence, larger compression ratios are preferable because they correspond to smaller compressed files. For the benchmark we define the input data size as the number of bytes necessary to store the raw input arrays without any metadata. An input array $X \in \mathbb{R}^{T \times N_{(\mathrm{lon})} \times N_{(\mathrm{lat})} \times L \times V}$ with $M = T \times N_{(\mathrm{lon})} \times N_{(\mathrm{lat})} \times L \times V$ entries stored in 32-bit floating point precision will therefore be defined as having an input size of $4 \cdot M$ bytes. This definition of the input size ensures that our compression ratio measurements are independent of the data format that is used to store the data and that we are measuring the pure compression performance of the compression algorithm and not the ability of a compressor to deal efficiently with a specific data format.

Another important aspect is the computational cost of applying a compressor, which we measure in two different ways. Firstly, for each compressor we keep track of the **compression and decompression throughput**, i.e. what is the wall clock

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

time for compressing a fixed chunk of data? This metric is relevant because it allows us to extrapolate the time it would take to compress/decompress datasets of arbitrary size. On the other hand, raw throughput measurements are generally flawed because they are noisy and depend on the specific hardware and software environment which the benchmark code was executed in, including what other processes were executed at the same time. Hence, we additionally record the **instruction count** for both compressing and decompressing the data. Our benchmark ensures that the baseline compressors will reproducibly execute with exactly the same instruction counts on different machines. Therefore, instruction counts are a more objective measure of compressor complexity.

Notably, the prioritization between different compression metrics might change depending on the different compression tasks, use cases, and compute and storage resources available. Not only that, but the prioritization between encoding and decoding complexity might change depending on the use case as well. For example, if we have a large archival dataset, such as ERA5, that gets accessed by thousands of users, then the compression ratio and decompression speed should be maximized. In this case, it is important that scientists are able to download and decompress the data quickly on their personal machines or institutional clusters. However, it is less important that the data can be compressed quickly because this only has to be done once. Therefore, compression times of multiple hours or even days might be acceptable. Of course, other use cases could have different priorities. For example, in operational weather forecasting, the time and computational resources that can be allocated to compression are constrained by the rate at which new forecasts need to be produced.

2.3 Implementation Details

All the code to download the data and reproduce the experimental results for the benchmark is accessible open-source at https://github.com/ClimateBenchPress. The data from the different sources outlined in Table 1 are converted into Zarr arrays (Miles et al., 2020) with canonicalised dimensions $T \times N_{(\text{lon})} \times N_{(\text{lat})} \times L \times V$ to provide a common data format for the benchmark. To compare a new compressor, the minimum requirement is that it can be made to compress a Zarr array (e.g. by individually compressing the numpy arrays for each variable), measure the byte size of the compressed representation, and output a Zarr array with the decompressed output. In the benchmark, we use the simplistic numcodecs[7] compression framework, which provides a minimal API for compressing n-dimensional arrays. To add a new compressor to the reproducible benchmark baseline, it must be wrapped inside a numcodecs codec.

For the results presented in this paper, we additionally wanted to ensure the fairness, reproducibility, and comparability of the performance of different compressors. To compress the input data, we use the numcodecs-rs[8] project, which adapts many popular scientific compressors to the numcodecs compression API.[9] For the comparison of the baseline compressors in Section 3, different variables are compressed separately. Evaluating different multi-variable compression approaches is left for future extensions to the benchmark. To guarantee cross-platform reproducibility, the numcodecs-rs project compiles its

[7] https://github.com/zarr-developers/numcodecs
[8] https://github.com/juntyr/numcodecs-rs
[9] https://github.com/zarr-developers/numcodecs

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

compressors to WebAssembly bytecode inside an isolated Nix build environment and publishes them as pure Python packages that are compatible with the numcodecs API.

WebAssembly (WASM) is a cross-platform bytecode format that was initially designed to safely execute compiled programs on web pages. Given its focus on security and cross-web-browser compatibility, WebAssembly is designed to facilitate performant and fully sandboxed execution (Spies and Mock, 2021).

In our experimental evaluation, we use the Wasmtime[10] WebAssembly reference engine. While the core WebAssembly instruction set is not entirely deterministic, non-deterministic instructions are very limited by design (WebAssembly Working Group, 2019), and any single-threaded core WebAssembly program can be transformed to have bit-reproducible execution. The WebAssembly 3.0 draft specification even defines a deterministic profile for this use case (WebAssembly Community Group, 2025). It is furthermore possible to instrument any WebAssembly program with a virtual bytecode instruction counter, which can be used as a deterministic, fully reproducible, user-space performance metric that does not suffer from measurement noise or platform differences (Binder and Hulaas, 2006).

Since the Python packages provided by numcodecs-rs ship with the pre-compiled cross-platform WebAssembly bytecode and only use deterministic instructions, the wrapped compressors execute exactly the same on any platform, thus allowing this benchmark to report reproducible, platform-independent, and noise-free results. Overall, the usage of numcodecs-rs guarantees that, apart from wall-time execution speed measurements, all results and metrics are reproducible on any CPU architecture, from large HPC systems to interactive demonstrations on local machines.

3 Results

In order to gain an understanding of the current state of lossy compression algorithms for climate data, we evaluate a set of baseline compressors on ClimateBenchPress. For readers who are new to lossy compression, the presented results here should be seen as a helpful guideline for how to interpret the results of the benchmark and understand the different trade-offs involved when choosing between different codecs. For readers who are interested in using lossy compression for their own work, the results here can help narrow down the compressors they might want to consider. As mentioned earlier though, the benchmark is mainly designed to be a tool to *compare* different compressors and *not to validate* them. It is important that all users of lossy compression run their own use case-specific validations before applying lossy compression. This is especially important because different compressors can have substantially different qualitative behavior that can be hard to capture using general-purpose error metrics, as we will discuss in Section 3.3.

3.1 Baseline Compressors

As baselines, we mainly consider compressors that were specifically developed for floating-point data and scientific applications. A key feature of lossy compressors designed for scientific applications is a mechanism to control the maximum pointwise error (relative and/or absolute error) incurred during the compression process. A notable exception in our list of baseline codecs

[10]https://github.com/bytecodealliance/wasmtime

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Table 3. Supported types of error bounds for different compressors. The support for relative error bounds for SZ3, ZFP, and ZFP-ROUND is in brackets because while these compressors have some mechanism to bound the relative error, we found in our experiments that they tend not to strictly adhere to the provided error bounds.

Compressor	Pointwise Abs. Error	Pointwise Rel. Error	Average Abs. Error
Bit Rounding	✗	✓	✗
Stochastic Rounding	✓	✗	✗
SZ3	✓	(✓)	(✓)
ZFP	✓	(✓)	✗
ZFP-ROUND	✓	(✓)	✗
JPEG2000	✗	✗	✓
SPERR	✓	✗	✓

is the JPEG2000 codec, which was developed for image compression but has been increasingly applied to scientific data as well (Woodring et al., 2011; Silver and Zender, 2017; Pellicer-Valero et al., 2025). Since it has seen decades of development and continuous improvement, it can provide a reference point for what compression ratios may be possible with traditional compression schemes. We now briefly describe our baseline compressors.

Bit Rounding (Zender, 2016; Delaunay et al., 2019) rounds the floating point mantissa to a specified number of bits. This gives a transparent mechanism to control the desired precision in the data and provides an interpretable mechanism for understanding what information is lost in the compression stage. Similarly, **Stochastic Rounding** uses random noise to round the input data to the nearest specified decimal precision. The goal of the added noise is to avoid any systematic biases in the compressed data due to the rounding process, which can otherwise remove small gradients in the data. We combine both of these rounding mechanisms with a lossless compressor. For each rounding mechanism we will evaluate two choices of lossless compressors: **Zstd**, a widely used byte compression codec designed for web data, and **PCO** (Loncaric et al., 2025), a novel lossless compressor designed specifically for numerical data. **SZ3 (Liang et al., 2022)** is a prediction-based error-bounded lossy compressor that uses Lorenzo predictors (Ibarria et al., 2003) and spline-based interpolators to exploit local correlations in the input data; in our experiments we use version 3.2.2. **ZFP (Lindstrom, 2014)** instead decorrelates the data by applying an orthogonal transform to $4 \times \cdots \times 4$ sub-blocks of X. We additionally evaluate a more recent variant of ZFP called **ZFP-ROUND** (Hammerling et al., 2019) that improved the rounding mechanism of the transform coefficients based on feedback from the climate science community. **JPEG2000 (Skodras et al., 2001)** uses a wavelet-based transform to decorrelate the data and then quantizes individual entries in the transformed input space. JPEG2000 does not support floating-point data; instead, we linearly quantize the data. JPEG2000 does not provide a mechanism to bound the maximum pointwise error; instead, we convert the absolute or relative error bounds to peak signal-to-noise ratio (PSNR), which can be passed to the compressor as a target. Notably though, PSNR is a measure of *average* rather than maximum error. Finally, **SPERR** (Li et al., 2023) is also a wavelet-based transform compressor but with the added feature of allowing for bounding the maximum pointwise error.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

The exact versions of the compressors that we tested in the benchmark are given in Table B1.

As highlighted in Table 3, most compressors do not have support for specifying both absolute and relative error bounds. Therefore, we sometimes need to convert between absolute and relative error bounds in order to use the error bounds from Table 2 for a specific compressor. Specifically, according to Definition 1, a pointwise relative error bound, b_{rel}, satisfies $|X_i - \hat{X}_i| \leq b_{\text{rel}} \cdot |X_i|$ for all indices i, whereas a pointwise absolute error bound satisfies $|X_i - \hat{X}_i| \leq b_{\text{abs}}$. For relative error bounds the RHS value of the inequality varies depending on the input value $|X_i|$, whereas the RHS value is fixed for the absolute error bound. To convert a relative error bound, b_{rel}, to an absolute error bound, we simply find the smallest non-zero RHS value that appears in the dataset, i.e. $b'_{\text{abs}} = \min_i b_{\text{rel}} \cdot |X_i|$, where the minimum is taken over all *non-zero* values. This ensure that if a compressor only supports absolute error bounds, passing b'_{abs} as an error bound will ensure the original relative error bound is satisfied on all non-zero entries. Similarly, to convert an absolute error bound, b_{abs}, to a relative error bound b'_{rel} we want to ensure that every possible RHS value of the inequality is bounded by the original absolute error bound, i.e. $b'_{\text{rel}} \cdot |X_i| \leq b_{\text{abs}}$ for all i, which is satisfied by picking $b'_{\text{rel}} = b_{\text{abs}}/(\max_i |X_i|)$. Other approaches to converting relative to absolute error bounds and vice versa are feasible, e.g. Liang et al. (2018) describe an alternative method for bounding the ratio-relative error using a logarithmic transform.

3.2 Quantitative Results

A single evaluation sweep over all baseline compressors, input variables, error bounds, and evaluation metrics produces a large set of results that can be difficult to analyze. Our two main tools to summarize our benchmark results will be: 1) Scorecards, inspired by their use to evaluate different forecasting models at ECMWF[11] and in WeatherBench (Rasp et al., 2024); and 2) rate-distortion plots, which are common in the compression literature.

The scorecards in Figure 2 provide a quick visual overview of which compressors are performing well on the different performance metrics. We observe that JPEG2000 gives good compression ratios but tends to fail the provided error bound; this is not surprising as JPEG2000 has no in-built mechanism to constrain the pointwise error. SPERR, the other wavelet-based compressor, is mostly able to achieve comparable compression ratios but does not always adhere to the error bound due to known issues in encoding the error bound correction (Fallin and Burtscher, 2024). SZ3 is also able to provide compression ratios that are often on the same order of magnitude as those of JPEG2000 and SPERR. However, the benchmark also exposes some interesting failure cases for SZ3. For example, both the air and sea surface temperature variables ("ta" and "tos" columns in the scorecard, respectively) contain NaN values in the input data, but instead of preserving these NaN values, SZ3 fills in these regions in the input data with interpolated numerical values.[12] ZFP and ZFP-ROUND exhibit the same failure case, which is especially problematic because it is a silent failure; no error was raised during the compression process. JPEG2000

[11] https://sites.ecmwf.int/ifs/scorecards/

[12] Since our discovery of this issue, we have submitted a patch to SZ3, https://github.com/szcompressor/SZ3/pull/91, which has been published in v3.3. However, v3.3 introduced an out-of-bounds access bug, which we reported in https://github.com/szcompressor/SZ3/issues/119. At the time of writing, SZ3 v3.4 that would include the fix has not yet been published.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

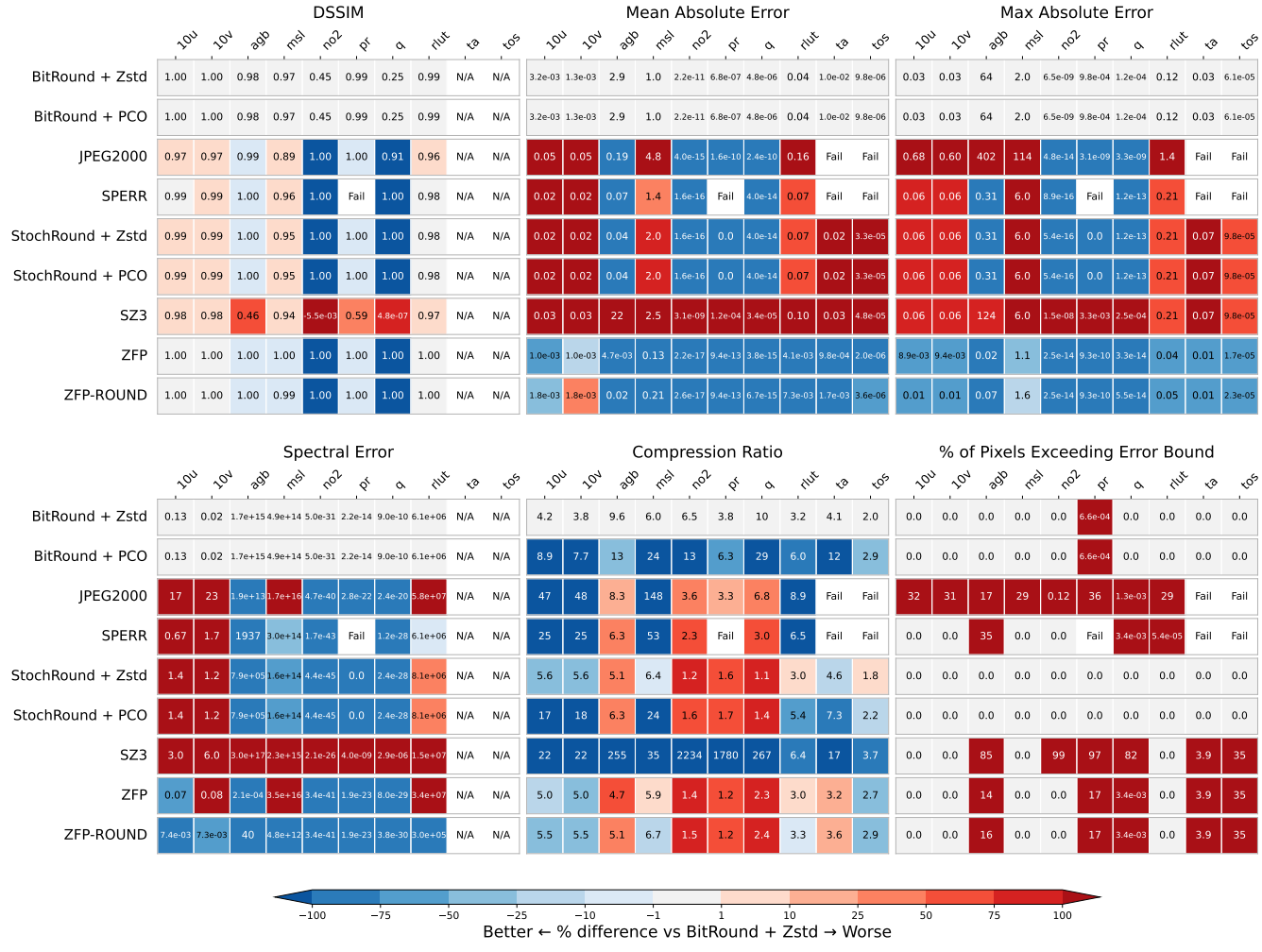

Figure 2. Overview scorecard to summarize the compression benchmark results. Each row shows the results for a different compressor, while each panel indicates a different metric, and each specific column shows the metrics for one of the compressed input variables. The panel "% of pixels exceeding error bound" indicates, for each compressor, the percentage of the inputs that exceed the variable-specific error bound. All the other metrics are described in Section 2.2. This scorecard shows the specific metrics evaluated on the “mid” level error bound; scorecards for the other error bounds are listed in Appendix C, “Fail” entries denote cases in which running the compressor raised an error. We found that the DSSIM and Spectral Error metrics are not robust to large regions of NaN values, so we exclude their computation for variables with NaNs (“ta” and “tos” variables); this is indicated by the “N/A” entries. The variable abbreviations correspond to the following inputs from Section 2.1: 10u = IFS 10 meter u-wind component, 10v = IFS 10 meter v-wind component, agb = ESA CCI Above ground biomass estimate, msl = IFS mean sea level pressure, no2 = CAMS Nitrogen Dioxide, pr = NextGEMS precipitation, q = IFS humidity, rlut = NextGEMS outgoing longwave radiation, ta = CMIP6 air temperature, tos = CMIP6 sea surface temperature.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

and SPERR fail on NaN inputs as well, but they raise an error instead of infilling any values.[13] In the case of JPEG2000, the failure can be attributed to the quantisation from floating-point numbers to integers, which does not account for NaN values.

Additionally, many compressors struggle to strictly adhere to relative error bounds; see Table 2 for the list of variables with relative error bounds. While SZ3 allows users to specify a relative error bound constraint, which is transformed to a range-relative absolute error bound with the formula $b'_{\text{abs}} = b_{\text{rel}} \cdot (\max_i X_i - \min_i X_i)$. Thus, SZ3 does not strictly preserve relative error bounds. ZFP, ZFP-ROUND,[14] and SPERR also struggle with relative error bounds as they all fail to adhere to the error bound for 0 values, e.g. in the biomass dataset large fraction of the dataset are zero in the regions representing the ocean.[15]

Even BitRound fails to adhere to the relative error bound on the precipitation data. Upon investigation of the ICON precipitation data, we identified that the failure is due to subnormal floating-point numbers being present in the ICON data. BitRounding fails to adhere to the error bound because the implementation that we are using computes the number of mantissa bits to keep as $k = -\lfloor \log_2(b_{\text{rel}}) \rfloor - 1$ from the relative error bound b_{rel}, but this formula fails to hold for subnormal values. This failure case provides an illuminating example of why we need rigorous evaluation checks for lossy compressors. In many ways, this is a benign failure. The values are so small, smaller than $\approx 1.1754942107 \times 10^{-38}$ kg m^{-2} s^{-1}, that they are physically implausible and a numerical artifact from the model. Furthermore, BitRound will round these values to 0, so the impact on calculating any downstream statistics, e.g. total precipitation, will be insignificant. At the same time, exactly because these values are numerical artifacts from the model, it is important that lossy compression algorithms do not "mask/round away" these unphysical values, or at least make users aware of this special case. If this particular implementation of BitRounding were deployed in a way that all model outputs are lossily compressed before being written to disk, the model developers would have no way of knowing that the model produces subnormal values as outputs.[16]

Overall, our results demonstrate the need for a benchmark as a testing ground for compression developers. None of the failure cases of the compressors are necessarily difficult to overcome from an algorithm design perspective (failure on NaNs and adherence to relative error bounds), but the lack of an existing benchmark means that, currently, compression developers often do not consider these requirements and edge cases.

While scorecards provide a detailed overview of the compressor performance across different metrics and inputs, they make it difficult to visually assess the trade-off between compression ratio and distortion errors. Rate-distortion plots, as those shown in Figure 3, allow us to assess exactly this trade-off. With rate-distortion plots, we can get an understanding of what compressors lie in the Pareto frontier of giving both high compression ratios and low levels of distortion, for different definitions of distortion.

Figure 3 shows that JPEG2000, SZ3, and SPERR generally provide the highest compression ratios across the different input variables. This, however, needs to be seen in the context of the results from the scorecards that showed these compressors

[13] Notably, the SPERR HDF5 (https://github.com/NCAR/H5Z-SPERR) plugin explicitly checks for NaNs and does not have this failure case.

[14] For both version of ZFP, we always run in "fixed-accuracy" mode and transform relative errors to absolute errors.

[15] Note that in our definition of the relative error bound as $|X_i - \hat{X}_i| \leq |X_i| \cdot b_{\text{rel}}$, for a true value of $X_i = 0$ the bound is satisfied if and only if $\hat{X}_i = 0$. Hence, in the benchmark, applying a relative error bound check implicitly also checks that 0 values are preserved in the data.

[16] SPERR also fails on the precipitation data because the existence of subnormal numbers in the data means the converted absolute error bound that is passed to the compressor is very small which causes SPERR to error.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

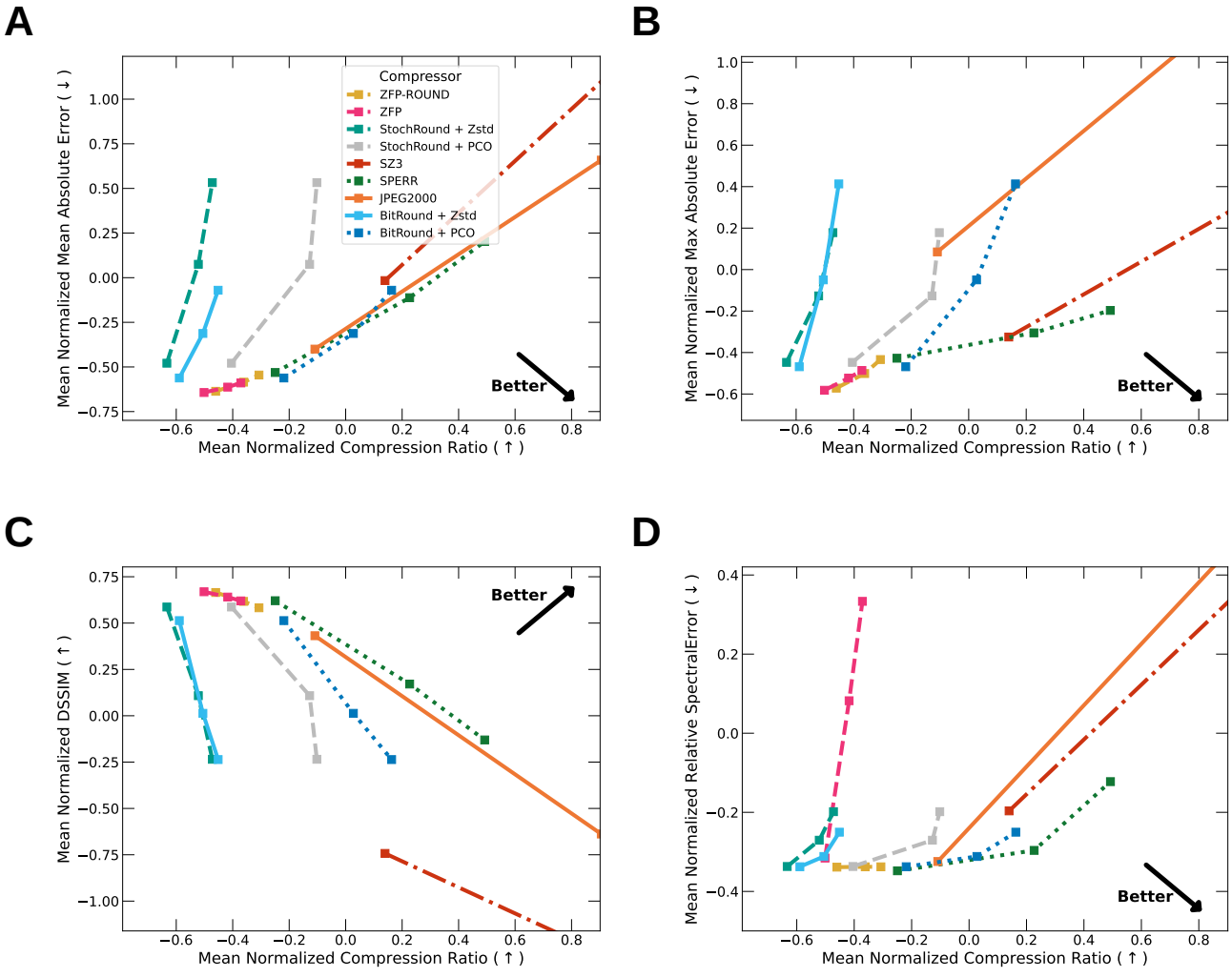

Figure 3. Rate-distortion plots for multiple distortion metrics: (**A**) normalized mean absolute error; (**B**) normalized maximum absolute error; (**C**) normalized DSSIM; (**D**) normalized spectral error. Each plot shows the compression ratio on the x-axis and a different distortion metric on the y-axis. As we saw in the scorecards from Figure 2, the absolute values of the metrics can vary quite significantly between input variables. If we would simply average the metrics over variables, the average would be dominated by only a small number of variables which achieve outlier values on a given metric. Hence, for each metric and variable we first normalize the metric before averaging them. Specifically, we have a set of metric values $\{m_{v,b,c}\}$ for input variable v, error bound b, and compressors c. We compute the mean and standard deviation over all compressors and error bounds to compute **normalized values**: $n_{v,b,c} = (m_{v,b,c} - \text{mean}_{c,b}(\{m_{v,b,c}\}))/\text{std}_{c,b}(\{m_{v,b,c}\}))$. This provides normalized scores, and the plotted values are the normalized scores for a given compressor and error bound averaged over all variables, i.e. $n_{c,b} = \text{mean}_v(\{n_{v,c}\})$. Axes limits were selected to exclude outlier values; the high compression ratios of SZ3 and JPEG2000 are achieved through exceeding the provided error bounds. Appendix D shows the uncropped rate-distortion curves.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

often fail to strictly adhere to the provided error bound. BitRounding in combination with the PCO lossless compressor, "BitRound + PCO" in the plots, is surprisingly competitive and often provides compression ratios that are even competitive with more sophisticated compressors such as SPERR. The notable jump in the compression ratios between "BitRound + Zstd" and "BitRound + PCO" highlights that we can still get quite significant improvements in the compression performance by using lossless compressors that are specifically designed for numerical data. Overall, these results suggest that the simple combination of BitRound + PCO is a very competitive alternative for users who are interested in not exceeding pointwise error bounds.

Finally, Figure 4 displays the last aspect of compressor performance that we have left out so far: compression cost. We measure cost in terms of bytecode instruction count per encoded/decoded byte of data. We focus here on instruction count in favor of wall clock throughput measurements because the platform-independent-bytecode instruction counts are reproducible between different CPU hardware architectures. Throughput measurements, on the other hand, tend to be noisy due to their dependence on, e.g. hardware details and what other processes are running on the same machine at the same time. Wall clock throughput measurements are shown in the supplementary in Figure E1. [17] Unsurprisingly, JPEG2000 is the most expensive compressor and, generally, encoding is more costly than decoding for all compressors except for both versions of ZFP. Even though BitRound and StochRound are simple lossy compression mechanisms, the lossless codecs PCO and Zstd add significant complexity to the entire compression procedure so that their compression cost is comparable to more complex lossy compressors such as SZ3, ZFP, and ZFP-Round.

3.2.1 Sensitivity of Compressor Performance to Chunking

As a default, we are passing the input data to compressors as a single chunk because of the relatively small overall size of our datasets. Popular file formats, like NetCDF and Zarr, have explicit support for storing data in chunks, which allows for efficient access of subsets of large datasets because it avoids the need to load the full dataset into memory.

As most climate datasets are stored in a chunked representation, we conducted a sensitivity analysis to study the impact that chunking has on compression performance. Determining good chunk sizes for a dataset is mainly application-specific because it depends on the typical access patterns to the data. Here, we chose a simple heuristic that ensures the chunk dimensions are roughly proportional and clean divisors of the full dataset dimensions. Additionally, the heuristic ensures each chunk is at least 4 MB in size, following recent guidance for CMIP7 datasets (Hassell and Cimadevilla Alvarez, 2025).

Figures 5 and 6 show that the overall impacts of chunking on the compression ratio and mean absolute error are small. Across datasets and compressors, there is no clear trend on whether chunking improves the compression ratio or not. The difference in compression ratios between the chunked and the not chunked data is generally < 1, demonstrating that the overall effect is small. On variables with a relative error bound, the SZ3 compressor provides a notable exception to this. This is due to the aforementioned fact that SZ3 internally transforms the relative error bound into an absolute error bound constraint with

[17] The wall clock throughput measurements in the supplementary Figure E1 should only be used as a relative comparison between different compressors because our compilation of each compressor to WebAssembly incurs an overhead (Jangda et al., 2019). Effectively, our compilation to WebAssembly trades off execution speed in favor of reproducibility and portability, which we prioritized for this benchmark.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

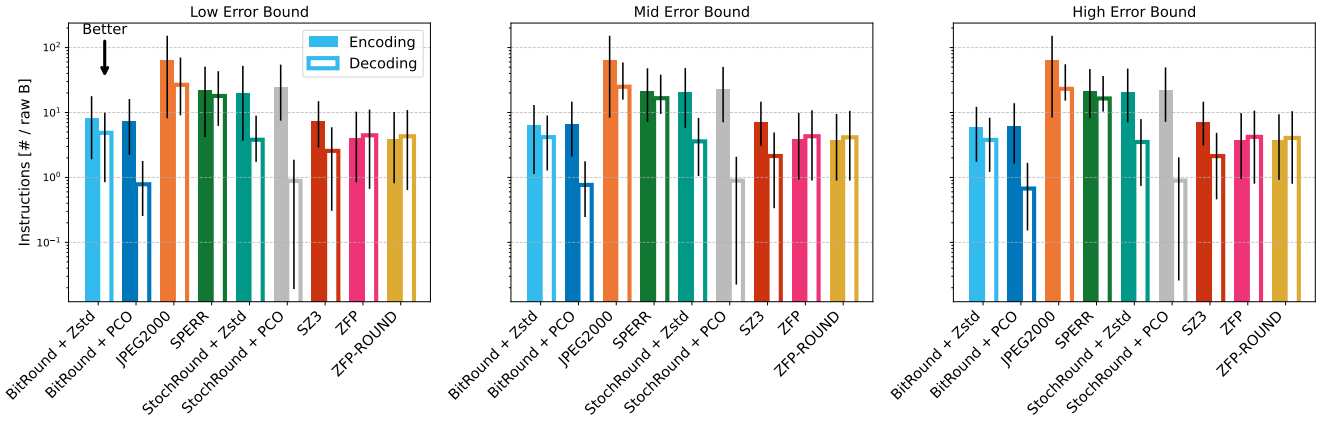

Figure 4. Comparison of encoding and decoding cost, measured in number of instructions per encoded/decoded byte [# / raw B], for individual compressors and error bound levels. Solid bars indicate the encoding cost and hollow bars the decoding cost, bar height shows the median value across all variables and the uncertainty bars indicate the 25% and 75% quantiles. Note that the instruction count metric has no measurement noise – the variance only comes from the combination across several variables of different sizes. The instruction count in the JPEG2000 compressor currently does not include the cost of a per-element cast to `float64` and `uint32` during linear quantization preprocessing, which add two instructions per 4 (`float32`) or 8 (`float64`) bytes during both encoding and decoding.

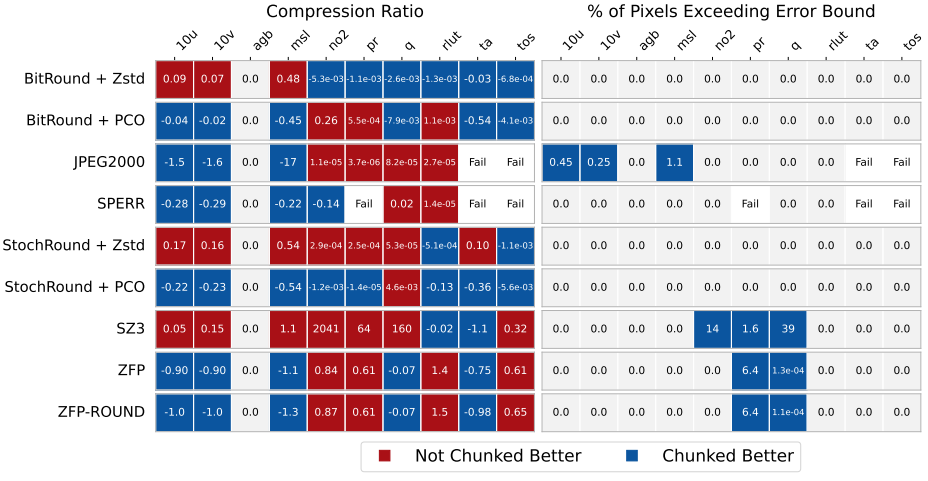

Figure 5. Difference in compression ratio and percentage of pixels exceeding the error bound between the not chunked and the chunked data. For a given metric, let m_{nc} denote the score for the data that is not chunked and m_{c} denote the score for the data that is chunked, then the scorecard here shows the difference $d = m_{\text{nc}} - m_{\text{c}}$. For the above-ground biomass variables, the chunking algorithm keeps the whole dataset as a single chunk and therefore, the difference in the metrics is always 0.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

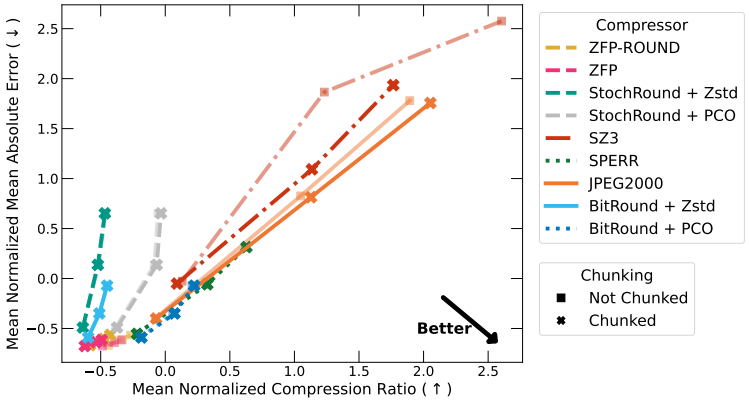

Figure 6. Impact of chunking on normalized mean absolute error and compression ratio. Lines with crosses indicate results computed on chunked data and faded lines with squares indicate results computed on not chunked data.

the relationship $b'_{\text{abs}} = b_{\text{rel}} \cdot (\max_i X_i - \min_i X_i)$. If the data is passed in chunks to SZ3, the range of the data in the chunk, $(\max_i X_i - \min_i X_i)$, will vary between chunks and therefore SZ3 will in effect apply different error bounds to different chunks. This leads to the large observed difference in compression ratios for SZ3 in the nitrogen dioxide and precipitation variables.

Overall, the impact of chunking on compressor performance depends on the specific characteristics of both the input data and the used compressor. While we considered only one explicit chunking scheme here, we encourage benchmark participants to further explore different chunking choices and report how they affect compressor performance. For example, some chunking strategies might include artifacts at the chunk boundaries that could negatively affect the reconstruction error.

3.3 Qualitative Results

While the quantitative evaluation metrics from the previous section provide a good initial comparison of the different compressors, we found significant qualitative differences in the behavior of the baseline compressors.

Error Distributions. Different lossy compressors generally have different pointwise error distributions, as highlighted previously in e.g Lindstrom (2017). Figure 7 illustrates the distribution of pointwise errors for the IFS sea level pressure variable, revealing that different compressors lead to substantially different error distributions. Compressing data with SZ3, SPERR, and StochRound leads to roughly uniform error distributions within the provided error bound; this also explains why in the scorecard in Figure 2, these compressors often produce higher mean and maximum absolute errors. Overall, this is quite a desirable property in a lossy compressor. After all, if a user provides a certain error bound to the compressor, we generally want to achieve the highest possible compression ratio for that error bound, which could benefit from exploiting the full range of the provided error bound.

ZFP and ZFP-ROUND tend to lead to normally distributed errors, so to satisfy the error bounds ZFP's error distribution tends to be more concentrated around 0 (Lindstrom, 2017). The fairly conservative error distribution of BitRound is due to the

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

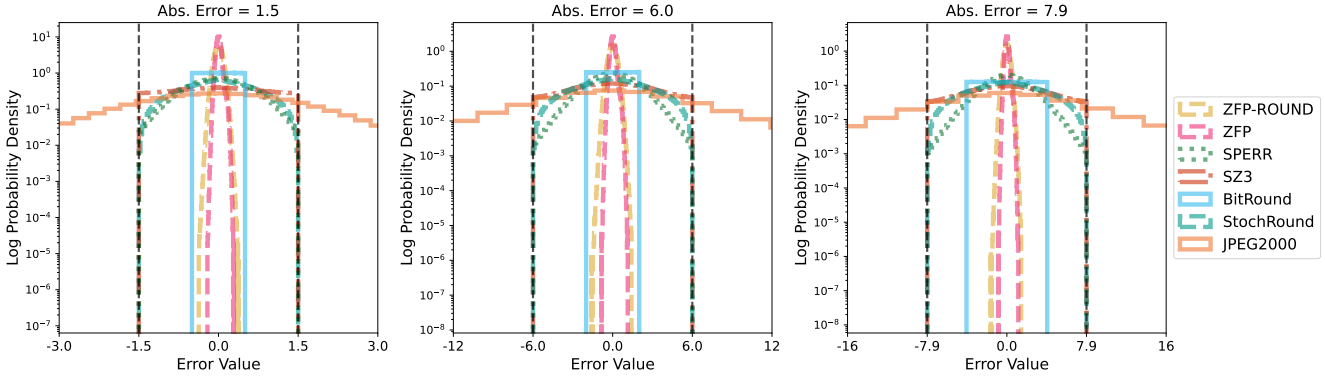

Figure 7. Error distributions for IFS sea level pressure for the three error bound levels. The black dashed vertical lines indicate the values of the error bounds. We do not differentiate between different combinations of StochRound and BitRound with lossless compressors because, by definition, the lossless compressors do not change the error distribution.

fact that BitRounding is an operation that inherently allows users to control the relative error but not the absolute error. When converting between absolute and relative errors in our benchmark, we have to be very conservative such that the transformation of the error bound ensures that no point will exceed the absolute error bounds. A clear outlier is JPEG2000 because it has a much wider error distribution due to only optimizing for a global average error, as discussed before.

NaN Behavior. We mentioned in Section 3.2 that several compressors struggle on inputs that contain NaN values. In particular, JPEG2000 and SPERR raise errors and refuse to compress any data with NaN entries. SPERR is the only compressor that does this natively; the JPEG2000 failure is caused by our wrapper due to having to convert the input floating point data to integer data. As shown in Figure 8, the SZ3, ZFP, and ZFP-ROUND compressors actually replace some of the NaN values with non-NaN floating-point values. In the case of SZ3, the values are smooth interpolations of the sea-surface temperatures. While the differences clearly stand out for sea surface temperature fields, this behavior might be more difficult to spot if the input fields only contain a small number of NaN values, e.g. when representing temporarily missing observation values. Similarly, ZFP infills some floating point data at the edges of NaN boundaries.

Spatial Distribution of Errors. Figure 8 also demonstrates that the different compressors not only lead to different global error distributions, as shown in Figure 7, the different compressors also have different *spatial error patterns*. BitRounding is inherently an operation that controls the relative error and therefore the magnitude of its reconstruction error will vary with the magnitude of the data, as we can clearly see in Figure 8. The stochastic nature of StochRound leads to noisier error distributions. For SZ3 and ZFP, the spatial error distribution is determined by their algorithm design. While there exist error metrics that explicitly take into account the spatial distribution of errors (Poppick et al., 2018), whether any given spatial pattern in errors is problematic is fundamentally dependent on the specific downstream analyses a user wants to conduct. It is therefore important for lossy compression users to run their own validation that is specific to their use case.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

4 Conclusions

ClimateBenchPress addresses the need for a standardized benchmark of lossy compression algorithms for weather and climate data sets. The benchmark covers a diversity of different data sources (CMIP6, IFS, NextGEMS, CAMS, ESA CCI) with grid resolutions ranging from ∼100 m to ∼150 km. We select a small subset of variables from these data sources that cover land, ocean, and atmosphere components of the Earth system. For each variable in the benchmark, we derive quantitative error bounds with an automated algorithm that analyses the ensemble spread of each variable in the ERA5 ensemble (Tyree et al., 2026). Our benchmark design keeps the total size of the input data below a couple of gigabytes for easier evaluation and reproducibility, downloads to laptops, and compression without chunking in WebAssembly.

Our evaluation of a set of nine baseline compressors revealed that there are significant differences in their quantitative and qualitative performance. The benchmark uncovered edge cases that existing compressors struggle with, such as properly preserving NaN values. BitRounding in combination with the lossless compressor PCO generally provided robust performance across different variables, although it failed in properly handling sub-normal floating-point numbers, and generally only provides modest compression ratios of around 10. Higher compression ratios are achieved by SZ3, JPEG2000, and SPERR, often reaching compression ratios of 20-100 or more, but these compressors have various failure cases, which means care needs to be taken when they are applied in practice. Specifically, SZ3 silently filled in NaN values, a bug that has now been addressed due to our feedback, and struggles with relative error bounds; JPEG2000 does not provide support for NaN values or point-wise error bounds at all, and SPERR sometimes exceeds the user-provided error bound. None of the technical issues are infeasible to fix, but their existence demonstrates the need for a robust benchmark to guide the development of compressors for climate data.

We emphasize that the benchmark is not intended as an exhaustive validation suite, but rather as a tool for assessing the comparative performance of lossy compressors. It is designed to expose the trade-offs that each compression algorithm makes and to provide an estimate of the rough order of magnitude compression ratio that users can expect in practice. The benchmark does not replace data- and model-specific validations of lossy compression algorithms. For example, as shown in Section 3.3, different compressors exhibit qualitatively distinct behaviors, often leading to varying spatial patterns in their errors, which has also been previously highlighted in the literature (Hammerling et al., 2019). Whether these spatial patterns are significant is inherently dependent on the specific analyses that users intend to conduct on the data.

While we kept the total size of the benchmark dataset small, an important avenue for future work is to provide an additional benchmark that challenges users to compress the full output of a high-resolution climate model run, possibly terabytes of data, and measure the maximum achievable compression ratio. Such a challenge would complement the current benchmark because the sheer size of the output data inherently increases the barrier to entry.

ClimateBenchPress is intended to serve as a guideline for compression developers designing the next generation of algorithms. All code and data for ClimateBenchPress are openly accessible at https://github.com/ClimateBenchPress, and we specifically encourage compression developers to add their own compressors to the benchmark. The codebase is also easily ex-

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

tensible, allowing the addition of new data sources and metrics, thereby enabling the wider community to validate compression algorithms on their own datasets.

Code and data availability. All code and data is available open source at https://github.com/ClimateBenchPress. Frozen code versions that were used to produce the results presented in this paper are available at https://doi.org/10.5281/zenodo.18015682 (Reichelt and Tyree, 2025), to download the datasets used in the benchmark, and https://doi.org/10.5281/zenodo.18152639 (Reichelt and Tyree, 2026), to run and evaluate the compressors.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

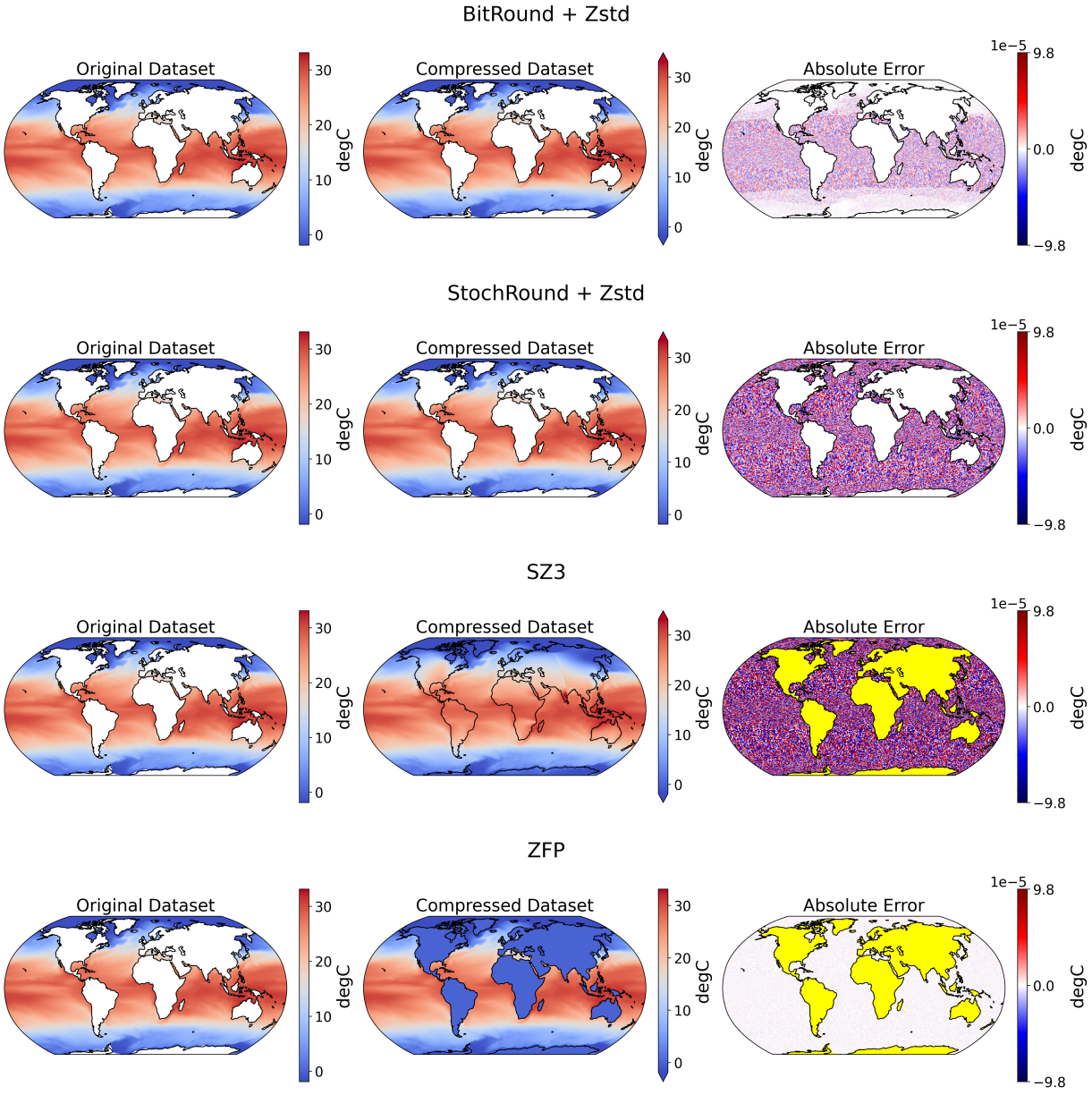

Figure 8. CMIP6 Sea Surface Temperature Errors. For each compressor, the leftmost panel shows the original input field, the middle panel shows the reconstructed field after compressing and decompressing the input, and the right panel plots the error. Yellow indicates regions where input values are NaNs but the compressor produces non-NaN values. Note that errors are non-zero but significantly below the error bounds; best viewed digitally.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Appendix A: Sample Benchmark Data

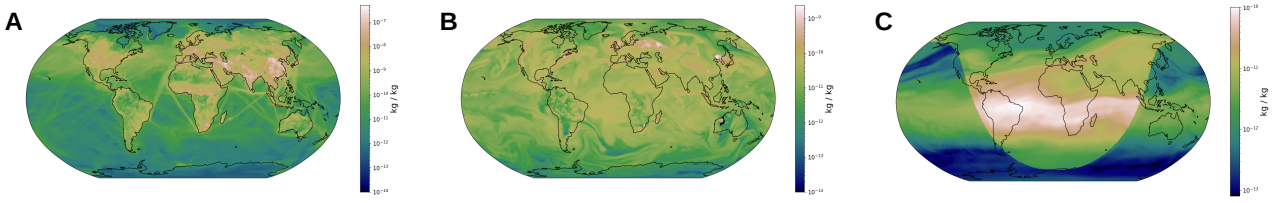

Figure A1. Nitrogen Dioxide reanalysis data from the Copernicus Atmospheric Monitoring Service (CAMS). (**A**) pressure level 1000hPa; (**B**) pressure level 500hPa; (**C**) pressure level 1hPa.

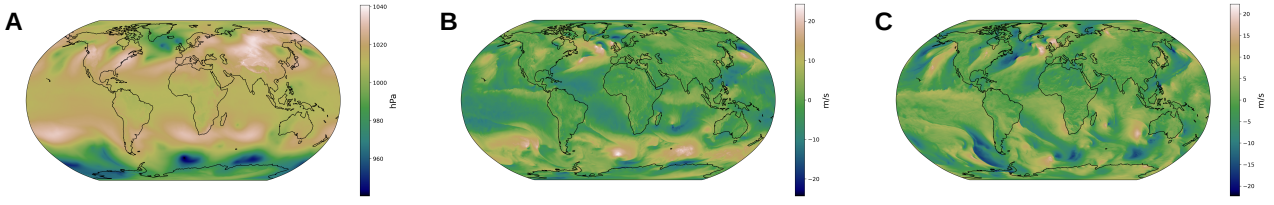

Figure A2. Variables from the IFS dataset. (**A**) sea-Level Pressure; (**B**) 10m u component of wind; (**C**) 10m v component of wind.

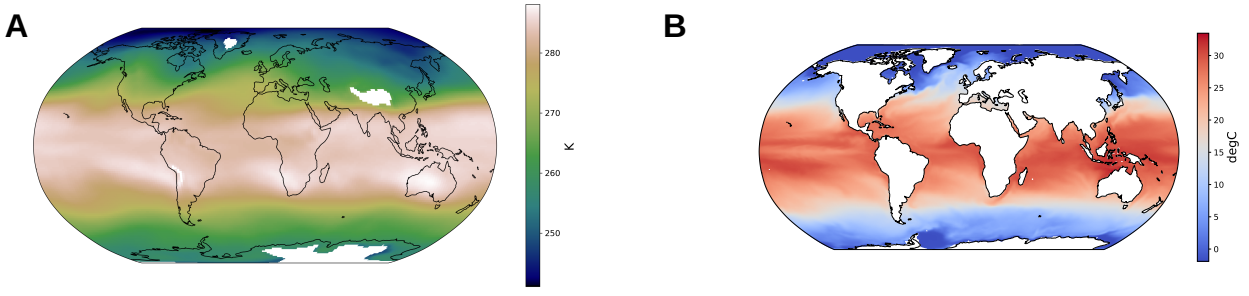

Figure A3. CMIP6 data for the ACCESS-ESM1-5 model. (**A**) air temperature at 700hPa; (**B**) sea surface temperature.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

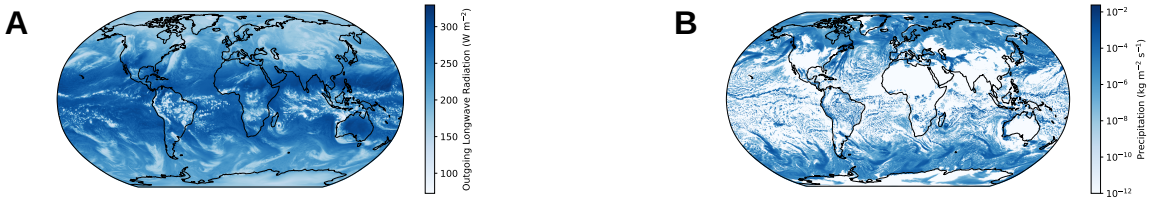

Figure A4. Sample output fields from the ICON model in the NextGEMS intercomparison project. (**A**) outgoing longwave radiation; (**B**) precipitation.

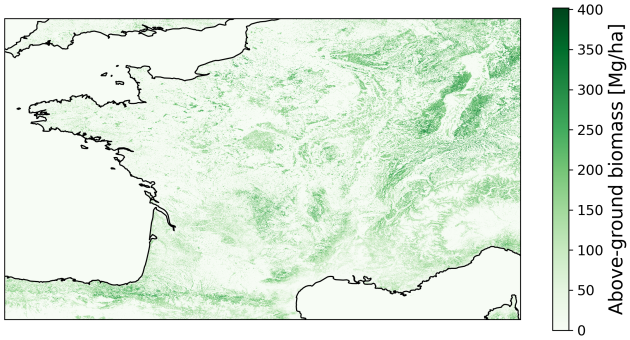

Figure A5. Sample output of the ESA biomass dataset.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Appendix B: Compressor Versions

Table B1. Versions of the compressors used in this benchmark. For each compressor, we provide the published version, if applicable, and the name and version of the corresponding numcodecs-rs Python package.

Compressor	Version	Python package
Bit Rounding	-	numcodecs-wasm-bit-round==0.4.0
Stochastic Rounding	-	numcodecs-wasm-stochastic-rounding==0.2.0
SZ3	3.2.2	numcodecs-wasm-sz3==0.7.0
ZFP	1.0.1	numcodecs-wasm-zfp-classic==0.4.0
ZFP-ROUND	1.0.1	numcodecs-wasm-zfp==0.6.0
JPEG2000	OpenJPEG 2.5.3	numcodecs-wasm-jpeg2000==0.3.0
SPERR	0.8.2	numcodecs-wasm-sperr==0.2.0
PCO	0.4.6	numcodecs-wasm-pco==0.3.0
Zstd	1.5.7	numcodecs-wasm-zstd==0.4.0

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Appendix C: Scorecards

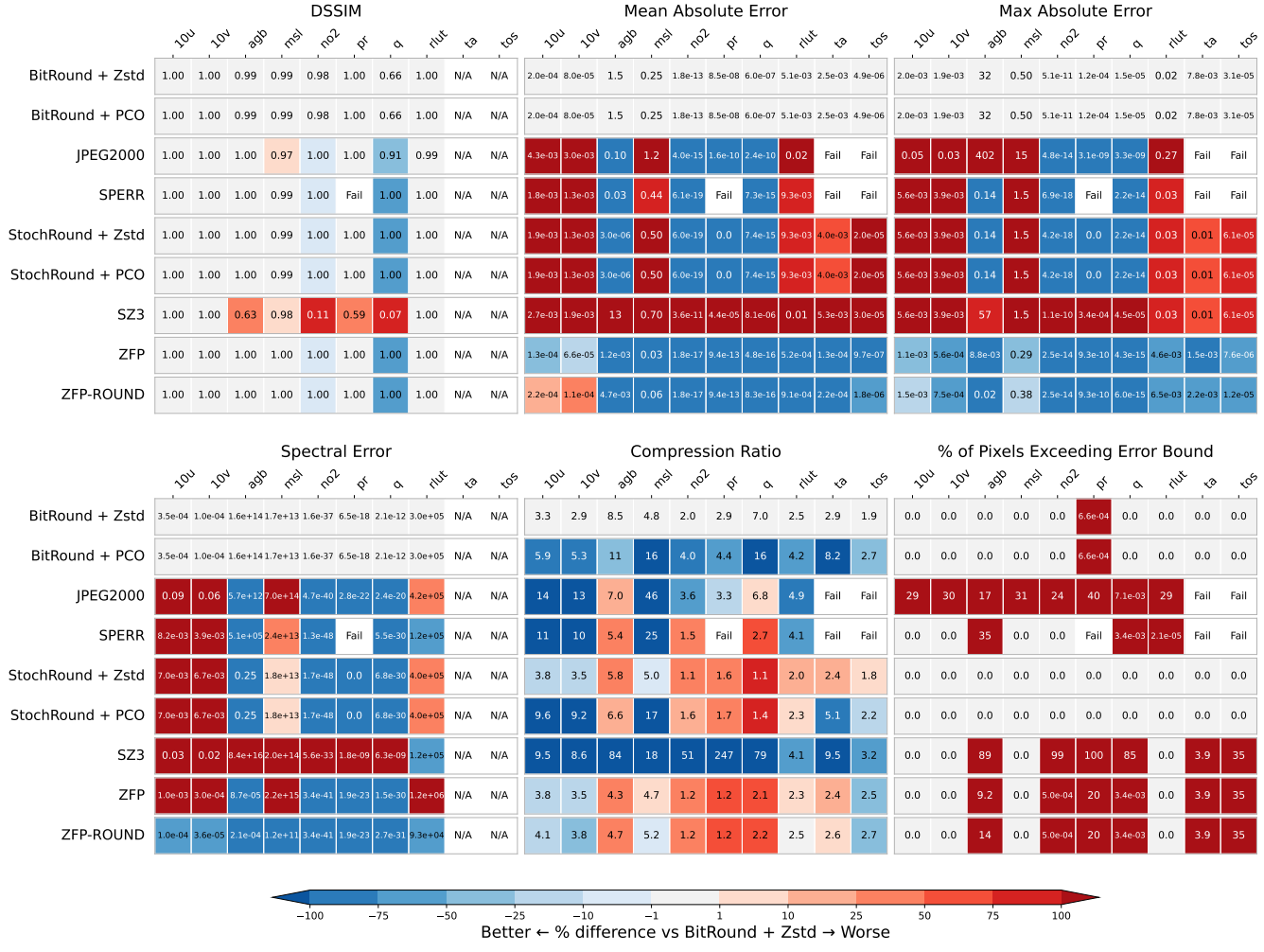

Figure C1. Scorecard for the **low** error bound. Figure conventions as in Figure 2.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

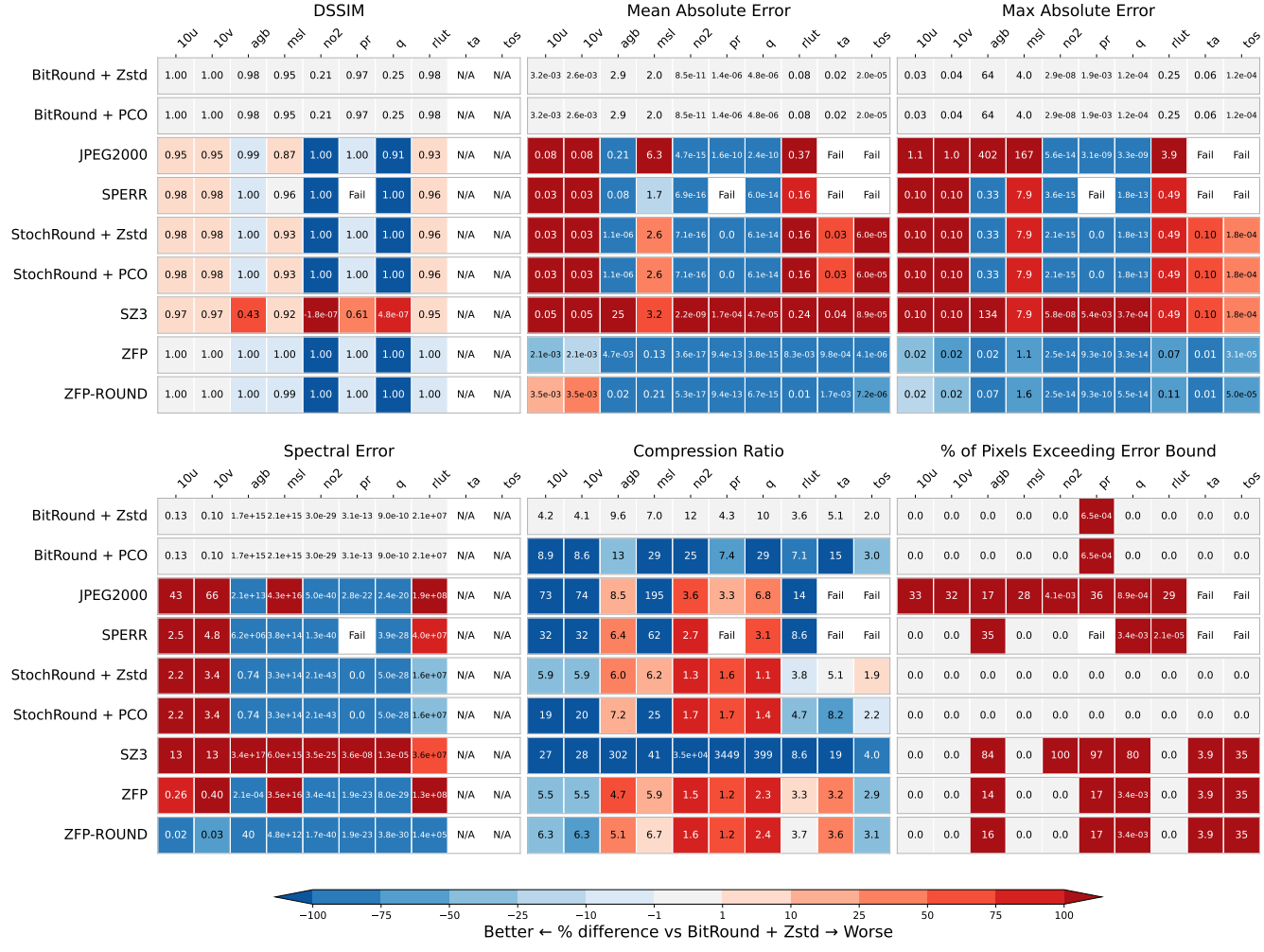

Figure C2. Scorecard for the **high** error bound. Figure conventions as in Figure 2.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Appendix D: Full Rate-Distortion Plots

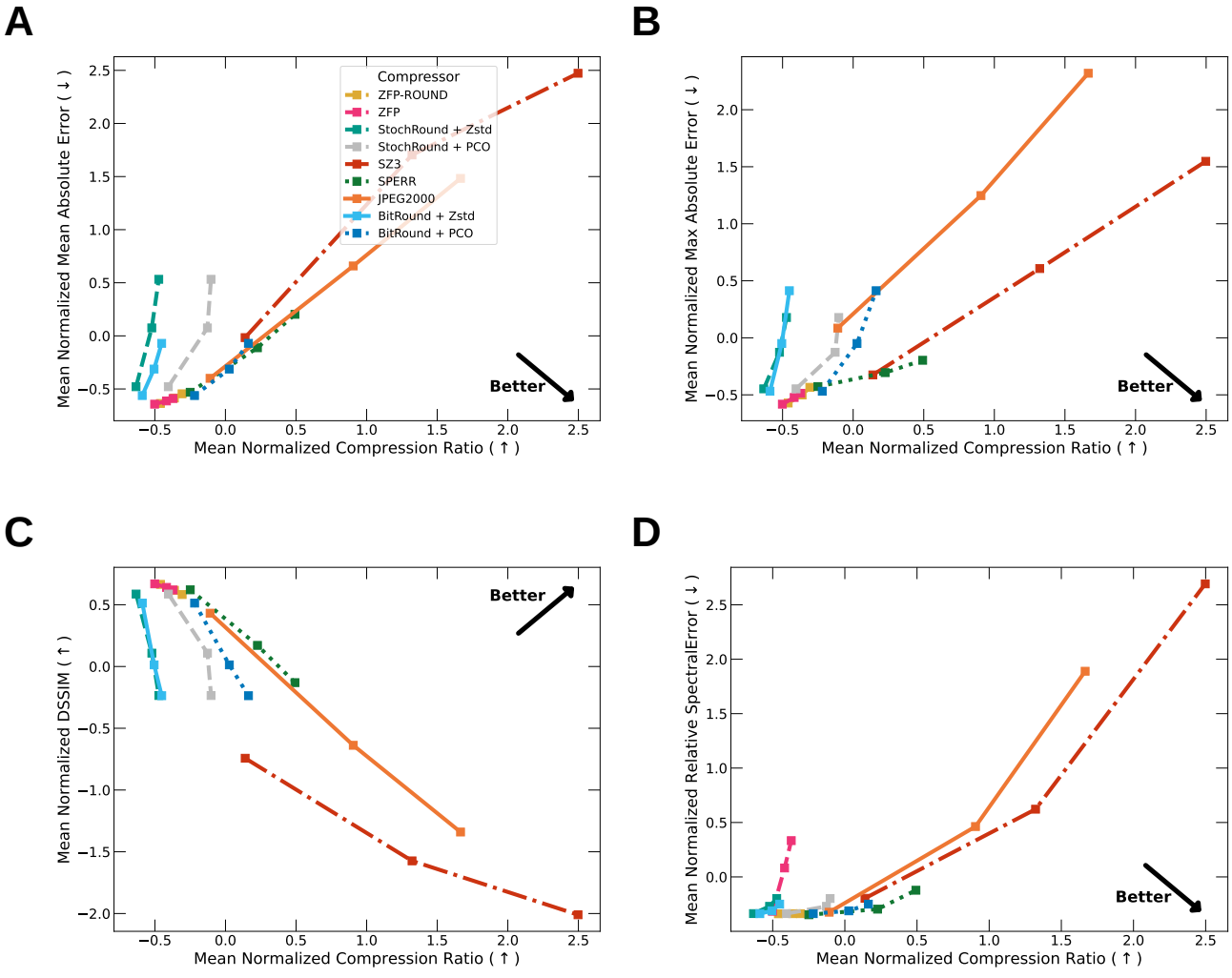

Figure D1. Full Rate-Distortion curves for multiple metrics without the cropped axes: (**A**) normalized mean absolute error; (**B**) normalized maximum absolute error; (**C**) normalized DSSIM; (**D**) normalized spectral error. Conventions as in Figure 3.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Appendix E: Throughput Measurements

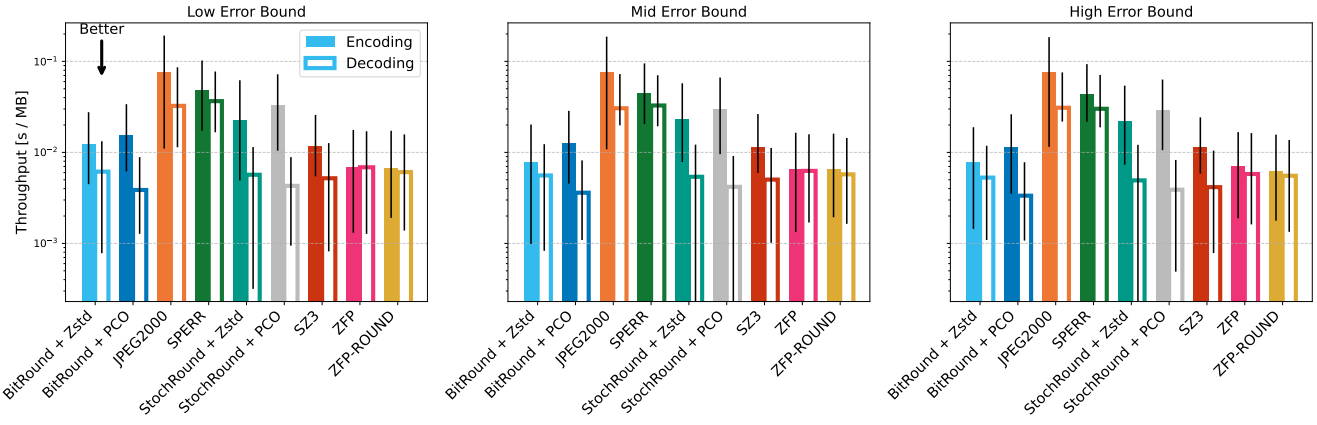

Figure E1. Comparison of encoding and decoding throughput, measured in [s / MB], for individual compressors and error bound levels. Solid bars indicate the encoding time and hollow bars the decoding time, bar height is showing the median value across all variables, and the uncertainty bars indicate the 25 % and 75 % quantiles. Note that these throughput measurements should only be seen as a relative comparison. Our benchmarking code incurs a significant amount of overhead (Jangda et al., 2019) through the compilation to WebAssembly and to ensure reproducibility. Measurements are run on a AMD EPYC 9654 CPU and utilize a single thread.

Author contributions. TR, JT, MK, PS led the benchmark design and conceptualization. JT developed the numcodecs-rs compressor wrappers. TR and JT both contributed to the software development of the benchmark. TR led drafting the manuscript with input from JT, MK, PS. PD, AB, SFN, TH, BL provided feedback on the benchmark design and reviewed and edited the manuscript.

Competing interests. The contact author has declared that none of the authors has any competing interests.

Acknowledgements. Tim Reichelt and Philip Stier acknowledge funding from the EU's Horizon Europe program under grant agreement number 10113184 and also acknowledge funding from UK Research and Innovation (UKRI). Tim Reichelt also received funding from ARIA and DSIT and Pillar VC under the Encode: AI for Science Fellowship. Juniper Tyree and Sara Faghih-Naini are funded by the ESiWACE3 Centre of Excellence, funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) under grant agreement number 101093054. Juniper Tyree wishes to acknowledge CSC – IT Center for Science, Finland, for computational resources. Milan Klöwer acknowledges funding from the Natural Environment Research Council under grant number UKRI191. Peter Dueben acknowledges funding from the WeatherGenerator project, funded by the European Union under grant agreement No 101187947.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

References

Baker, A. H., Hammerling, D. M., Mickelson, S. A., Xu, H., Stolpe, M. B., Naveau, P., Sanderson, B., Ebert-Uphoff, I., Samarasinghe, S., De Simone, F., et al.: Evaluating lossy data compression on climate simulation data within a large ensemble, Geoscientific Model Development, 9, 4381–4403, 2016.

Baker, A. H., Xu, H., Hammerling, D. M., Li, S., and Clyne, J. P.: Toward a Multi-method Approach: Lossy Data Compression for Climate Simulation Data, in: High Performance Computing, edited by Kunkel, J. M., Yokota, R., Taufer, M., and Shalf, J., pp. 30–42, Springer International Publishing, Cham, ISBN 978-3-319-67630-2, https://doi.org/10.1007/978-3-319-67630-2_3, 2017.

Baker, A. H., Pinard, A., and Hammerling, D. M.: On a structural similarity index approach for floating-point data, IEEE Transactions on Visualization and Computer Graphics, 2023.

Ballester-Ripoll, R., Lindstrom, P., and Pajarola, R.: TTHRESH: Tensor compression for multidimensional visual data, IEEE transactions on visualization and computer graphics, 2019.

Bauer, P., Stevens, B., and Hazeleger, W.: A digital twin of Earth for the green transition, Nature Climate Change, 11, 80–83, 2021.

Binder, W. and Hulaas, J.: Using Bytecode Instruction Counting as Portable CPU Consumption Metric, Electronic Notes in Theoretical Computer Science, 153, 57–77, https://doi.org/https://doi.org/10.1016/j.entcs.2005.10.032, proceedings of the Third Workshop on Quantitative Aspects of Programming Languages (QAPL 2005). Available from: doi:10.1016/j.entcs.2005.10.032, 2006.

Delaunay, X., Courtois, A., and Gouillon, F.: Evaluation of lossless and lossy algorithms for the compression of scientific datasets in netCDF-4 or HDF5 files, Geoscientific Model Development, 12, 4099–4113, 2019.

Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., and Taylor, K. E.: Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization, Geoscientific Model Development, 9, 1937–1958, 2016.

Fallin, A. and Burtscher, M.: Lessons Learned on the Path to Guaranteeing the Error Bound in Lossy Quantizers, arXiv preprint arXiv:2407.15037, 2024.

Group, G.-R. C. W. and Program, G.-R. S.: Noaa goes-r series advanced baseline imager (abi) level 1b radiances, 2017.

Hammerling, D. M., Baker, A. H., Pinard, A., and Lindstrom, P.: A collaborative effort to improve lossy compression methods for climate data, in: 2019 IEEE/ACM 5th International Workshop on Data Analysis and Reduction for Big Scientific Data (DRBSD-5), pp. 16–22, IEEE, 2019.

Han, T., Chen, Z., Guo, S., Xu, W., and Bai, L.: CRA5: Extreme Compression of ERA5 for Portable Global Climate and Weather Research via an Efficient Variational Transformer, https://doi.org/10.48550/arXiv.2405.03376, 2024.

Hassell, D. and Cimadevilla Alvarez, E.: cmip7repack: Repack CMIP7 netCDF-4 datasets, https://doi.org/10.5281/zenodo.17550920, 2025.

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., et al.: The ERA5 global reanalysis, Quarterly Journal of the Royal Meteorological Society, 2020.

Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., and Thépaut, J.-N.: ERA5 hourly data on single levels from 1940 to present, https://doi.org/10.24381/cds.adbb2d47, accessed on 16-12-2025, 2023.

Huang, L. and Hoefler, T.: Compressing Multidimensional Weather and Climate Data into Neural Networks, https://doi.org/10.48550/arXiv.2210.12538, 2023.

Ibarria, L., Lindstrom, P., Rossignac, J., and Szymczak, A.: Out-of-core compression and decompression of large n-dimensional scalar fields, in: Computer Graphics Forum, vol. 22, pp. 343–348, Wiley Online Library, 2003.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Illingworth, A. J., Barker, H., Beljaars, A., Ceccaldi, M., Chepfer, H., Clerbaux, N., Cole, J., Delanoë, J., Domenech, C., Donovan, D. P., et al.: The EarthCARE satellite: The next step forward in global measurements of clouds, aerosols, precipitation, and radiation, Bulletin of the American Meteorological Society, 96, 1311–1332, 2015.

Inness, A., Ades, M., Agustí-Panareda, A., Barré, J., Benedictow, A., Blechschmidt, A.-M., Dominguez, J. J., Engelen, R., Eskes, H., Flemming, J., et al.: The CAMS reanalysis of atmospheric composition, Atmospheric Chemistry and Physics, 19, 3515–3556, 2019.

Jangda, A., Powers, B., Berger, E. D., and Guha, A.: Not so fast: analyzing the performance of webassembly vs. native code, in: Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference, USENIX ATC '19, p. 107–120, USENIX Association, USA, ISBN 9781939133038, https://doi.org/10.48550/arXiv.1901.09056, 2019.

Klöwer, M., Razinger, M., Dominguez, J. J., Düben, P. D., and Palmer, T. N.: Compressing Atmospheric Data into Its Real Information Content, Nature Computational Science, 1, 713–724, https://doi.org/10.1038/s43588-021-00156-2, 2021.

Koldunov, N., Kölling, T., Pedruzo-Bagazgoitia, X., Rackow, T., Redler, R., Sidorenko, D., Wieners, K.-H., and Ziemen, F. A.: nextGEMS: output of the model development cycle 3 simulations for ICON and IFS, https://doi.org/10.26050/WDCC/nextGEMS_cyc3, 2023.

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

Liang, X., Di, S., Tao, D., Chen, Z., and Cappello, F.: An Efficient Transformation Scheme for Lossy Data Compression with Point-Wise Relative Error Bound, in: 2018 IEEE International Conference on Cluster Computing (CLUSTER), pp. 179–189, https://doi.org/10.1109/CLUSTER.2018.00036, 2018.

Liang, X., Zhao, K., Di, S., Li, S., Underwood, R., Gok, A. M., Tian, J., Deng, J., Calhoun, J. C., Tao, D., et al.: Sz3: A modular framework for composing prediction-based error-bounded lossy compressors, IEEE Transactions on Big Data, 2022.

Lindstrom, P.: Fixed-rate compressed floating-point arrays, IEEE transactions on visualization and computer graphics, 2014.

Lindstrom, P.: Error distributions of lossy floating-point compressors, Tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2017.

Loncaric, M., Jeppesen, N., and Zinberg, B.: Pcodec: Better Compression for Numerical Sequences, https://arxiv.org/abs/2502.06112, 2025.

Miles, A., Kirkham, J., Durant, M., Bourbeau, J., Onalan, T., Hamman, J., Patel, Z., shikharsg, Rocklin, M., raphael dussin, Schut, V., de Andrade, E. S., Abernathey, R., Noyes, C., sbalmer, pyup.io bot, Tran, T., Saalfeld, S., Swaney, J., Moore, J., Jevnik, J., Kelleher, J., Funke, J., Sakkis, G., Barnes, C., and Banihirwe, A.: zarr-developers/zarr-python: v2.4.0, https://doi.org/10.5281/zenodo.3773450, 2020.

Mirowski, P., Warde-Farley, D., Rosca, M., Grimes, M. K., Hasson, Y., Kim, H., Rey, M., Osindero, S., Ravuri, S., and Mohamed, S.: Neural Compression of Atmospheric States, https://doi.org/10.48550/arXiv.2407.11666, 2024.

Pellicer-Valero, O. J., Aybar, C., and Valls, G. C.: Video Compression for Spatiotemporal Earth System Data, arXiv preprint arXiv:2506.19656, 2025.

Poppick, A., Nardi, J., Feldman, N., Baker, A., and Hammerling, D.: A statistical analysis of compressed climate model data, Proc. DRBSD, pp. 1–6, 2018.

Rackow, T., Pedruzo-Bagazgoitia, X., Becker, T., Milinski, S., Sandu, I., Aguridan, R., Bechtold, P., Beyer, S., Bidlot, J., Boussetta, S., Deconinck, W., Diamantakis, M., Dueben, P., Dutra, E., Forbes, R., Ghosh, R., Goessling, H. F., Hadade, I., Hegewald, J., Jung, T., Keeley, S., Kluft, L., Koldunov, N., Koldunov, A., Kölling, T., Kousal, J., Kühnlein, C., Maciel, P., Mogensen, K., Quintino, T., Polichtchouk, I., Reuter, B., Sármány, D., Scholz, P., Sidorenko, D., Streffing, J., Sützl, B., Takasuka, D., Tietsche, S., Valentini, M., Vannière, B., Wedi, N., Zampieri, L., and Ziemen, F.: Multi-year simulations at kilometre scale with the Integrated Forecasting System coupled to FESOM2.5 and NEMOv3.4, Geoscientific Model Development, 18, 33–69, https://doi.org/10.5194/gmd-18-33-2025, 2025.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Rasp, S., Hoyer, S., Merose, A., Langmore, I., Battaglia, P., Russell, T., Sanchez-Gonzalez, A., Yang, V., Carver, R., Agrawal, S., et al.: WeatherBench 2: A benchmark for the next generation of data-driven global weather models, Journal of Advances in Modeling Earth Systems, 16, e2023MS004 019, 2024.

Reichelt, T. and Tyree, J.: ClimateBenchPress data-loader, https://doi.org/10.5281/zenodo.18015682, 2025.

Reichelt, T. and Tyree, J.: ClimateBenchPress compressor, https://doi.org/10.5281/zenodo.18152639, 2026.

Rossow, W. B. and Schiffer, R. A.: ISCCP cloud data products, Bulletin of the American Meteorological Society, 72, 2–20, 1991.

Santoro, M., Cartus, O., Quegan, S., Kay, H., Lucas, R. M., Araza, A., Herold, M., Labrière, N., Chave, J., Rosenqvist, Å., et al.: Design and performance of the Climate Change Initiative Biomass global retrieval algorithm, Science of remote sensing, 10, 100 169, 2024.

Segura, H., Pedruzo-Bagazgoitia, X., Weiss, P., Müller, S. K., Rackow, T., Lee, J., Dolores-Tesillos, E., Benedict, I., Aengenheyster, M., Aguridan, R., Arduini, G., Baker, A. J., Bao, J., Bastin, S., Baulenas, E., Becker, T., Beyer, S., Bockelmann, H., Brüggemann, N., Brunner, L., Cheedela, S. K., Das, S., Denissen, J., Dragaud, I., Dziekan, P., Ekblom, M., Engels, J. F., Esch, M., Forbes, R., Frauen, C., Freischem, L., García-Maroto, D., Geier, P., Gierz, P., González-Cervera, A., Grayson, K., Griffith, M., Gutjahr, O., Haak, H., Hadade, I., Haslehner, K., ul Hasson, S., Hegewald, J., Kluft, L., Koldunov, A., Koldunov, N., Kölling, T., Koseki, S., Kosukhin, S., Kousal, J., Kuma, P., Kumar, A. U., Li, R., Maury, N., Meindl, M., Milinski, S., Mogensen, K., Niraula, B., Nowak, J., Praturi, D. S., Proske, U., Putrasahan, D., Redler, R., Santuy, D., Sármány, D., Schnur, R., Scholz, P., Sidorenko, D., Spät, D., Sützl, B., Takasuka, D., Tompkins, A., Uribe, A., Valentini, M., Veerman, M., Voigt, A., Warnau, S., Wachsmann, F., Wacławczyk, M., Wedi, N., Wieners, K.-H., Wille, J., Winkler, M., Wu, Y., Ziemen, F., Zimmermann, J., Bender, F. A.-M., Bojovic, D., Bony, S., Bordoni, S., Brehmer, P., Dengler, M., Dutra, E., Faye, S., Fischer, E., van Heerwaarden, C., Hohenegger, C., Järvinen, H., Jochum, M., Jung, T., Jungclaus, J. H., Keenlyside, N. S., Klocke, D., Konow, H., Klose, M., Malinowski, S., Martius, O., Mauritsen, T., Mellado, J. P., Mieslinger, T., Mohino, E., Pawłowska, H., Peters-von Gehlen, K., Sarré, A., Sobhani, P., Stier, P., Tuppi, L., Vidale, P. L., Sandu, I., and Stevens, B.: nextGEMS: entering the era of kilometer-scale Earth system modeling, EGUsphere, 2025, 1–39, https://doi.org/10.5194/egusphere-2025-509, 2025.

Shannon, C. E.: A mathematical theory of communication, The Bell system technical journal, 27, 379–423, 1948.

Silver, J. D. and Zender, C. S.: The Compression–Error Trade-off for Large Gridded Data Sets, Geoscientific Model Development, 10, 413–423, https://doi.org/10.5194/gmd-10-413-2017, 2017.

Skodras, A., Christopoulos, C., and Ebrahimi, T.: The JPEG 2000 still image compression standard, IEEE Signal processing magazine, 18, 36–58, 2001.

Spies, B. and Mock, M.: An Evaluation of WebAssembly in Non-Web Environments, in: 2021 XLVII Latin American Computing Conference (CLEI), pp. 1–10, https://doi.org/10.1109/CLEI53233.2021.9640153, available from: doi:10.1109/CLEI53233.2021.9640153, 2021.

Stephens, G. L., Vane, D. G., Boain, R. J., Mace, G. G., Sassen, K., Wang, Z., Illingworth, A. J., O'connor, E. J., Rossow, W. B., Durden, S. L., et al.: The CloudSat mission and the A-Train: A new dimension of space-based observations of clouds and precipitation, Bulletin of the American Meteorological Society, 83, 1771–1790, 2002.

Stevens, B., Satoh, M., Auger, L., Biercamp, J., Bretherton, C. S., Chen, X., Düben, P., Judt, F., Khairoutdinov, M., Klocke, D., et al.: DYAMOND: the DYnamics of the Atmospheric general circulation Modeled On Non-hydrostatic Domains, Progress in Earth and Planetary Science, 2019.

Thomas, M. and Joy, A. T.: Elements of information theory, Wiley-Interscience, 2006.

Tyree, J., Klöwer, M., Faghih-Naini, S., and Dueben, P.: Towards universal compression error bounds for weather and climate data, (Manuscript in Preparation), 2026.

https://doi.org/10.5194/egusphere-2026-60
Preprint. Discussion started: 9 February 2026

Underwood, R., Bessac, J., Di, S., and Cappello, F.: Understanding the Effects of Modern Compressors on the Community Earth Science Model, in: 2022 IEEE/ACM 8th International Workshop on Data Analysis and Reduction for Big Scientific Data (DRBSD), pp. 1–10, IEEE, Dallas, TX, USA, ISBN 978-1-66546-337-9, https://doi.org/10.1109/DRBSD56682.2022.00006, 2022.

WebAssembly Community Group: WebAssembly Specification Release 3.0 (Draft 2025-09-16), available from: https://webassembly.github.io/spec/versions/core/WebAssembly-3.0-draft.pdf [Accessed: 05.12.2025], 2025.

WebAssembly Working Group: WebAssembly Core Specification, available from: https://www.w3.org/TR/2019/REC-wasm-core-1-20191205 [Accessed: 23.01.2024], 2019.

Wieners, K.-H., Rackow, T., Aguridan, R., Becker, T., Beyer, S., Cheedela, S. K., Dreier, N.-A., Engels, J. F., Esch, M., Frauen, C., Klocke, D., Kölling, T., Pedruzo-Bagazgoitia, X., Putrasahan, D., Sidorenko, D., Schnur, R., Stevens, B., and Zimmermann, J.: nextGEMS: output of the production simulations for ICON and IFS, https://www.wdc-climate.de/ui/entry?acronym=nextGEMS_prod, 2024.

Woodring, J., Mniszewski, S., Brislawn, C., DeMarle, D., and Ahrens, J.: Revisiting wavelet compression for large-scale climate data using JPEG 2000 and ensuring data precision, in: 2011 IEEE symposium on large data analysis and visualization, pp. 31–38, IEEE, 2011.

Zender, C. S.: Bit Grooming: statistically accurate precision-preserving quantization with compression, evaluated in the netCDF Operators (NCO, v4. 4.8+), Geoscientific Model Development, 9, 3199–3211, 2016.