CRAFTBENCH: BENCHMARKING LLM-GENERATED PROGRAMS FOR FEATURE-PRESERVING SCIENTIFIC COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

The massive volumes of data produced by scientific computing have driven widespread adoption of error-bounded lossy compression, which can achieve high compression ratios while bounding reconstruction error at every data point. However, many downstream analyses depend on derived quantities and features, such as gradients, distributions, spectra, and topological structures, that may be substantially distorted even when the pointwise error bound is satisfied. Existing feature-preserving methods are typically designed for a specific quantity or feature, or a narrow class of them, requiring substantial redesign when the target changes. Recent advances in LLM-based code generation raise the possibility of automating the adaptation of existing compressors to preserve diverse scientific features. We introduce **CraftBench**, a benchmark for evaluating LLM-generated adaptations of existing compressors using executable feature definitions and task-specific distance functions. CraftBench comprises 83 feature-preservation tasks spanning numerical, spatial, geometric, spectral, composite, and domain-specific analyses. Each task is evaluated under three adaptation modes: error bounding (EB), configuration selection (CS), and error correction (EC). An adaptation is considered successful only if it satisfies four criteria: (1) it reduces the distance of the target derived quantity or feature by at least 50% relative to the base reconstruction; (2) it respects the prescribed pointwise error bound; (3) it is not dominated in feature distance and compressed size by measured uniform-error-bound references, constructed by recompressing the data with progressively tighter global error bounds; and (4) its compressed output remains smaller than a fixed lossless reference. Among 2,475 LLM-generated adaptations, 282 (11.4%) satisfy all four criteria. An additional 509 satisfy the feature-distance reduction, pointwise-error, and lossless-storage requirements but are dominated by the uniform-error-bound references. These results reveal a substantial gap between generating adaptations that improve preservation of the target quantity or feature and producing adaptations that outperform the simple baseline of uniformly tightening the global error bound.

1 INTRODUCTION

Modern scientific simulations and experimental facilities can generate data at terabyte-per-second rates. For example, high-rate 16-megapixel area detectors at LCLS-II can produce approximately 1 TB/s in a single experimental station (Cappello et al., 2025). Such data volumes create substantial challenges for storage, data transfer, checkpoint/restart, visualization, and downstream analysis. Error-bounded lossy compressors, such as SZ3 (Tao et al., 2017; Liang et al., 2023b) and ZFP (Lindstrom, 2014), are therefore widely used to reduce data volumes while providing explicit control over pointwise reconstruction errors (Cappello et al., 2019).

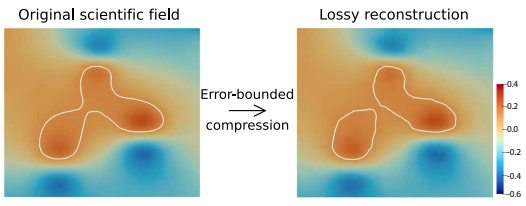


Figure 1: Feature distortion under error-bounded compression. A connected excursion-set region splits into two components after compression.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

However, satisfying a pointwise error bound does not guarantee the preservation of derived features and quantities of interest used in downstream scientific analyses. Compression errors can distort gradients, extrema, topological trees, segmentations, and other structures used in downstream analysis (Li et al., 2025; Yan et al., 2024; Gorski et al., 2025; Di et al., 2025; Cappello et al., 2025). One such structure is an excursion set: the set of locations at which a scalar field exceeds a prescribed threshold. As shown in Figure 1, SZ3 compression with a value-range-relative error bound of 10^{-2} causes a connected excursion-set region to split into two components. This distortion changes the number, spatial extent, and connectivity of detected high-value regions, potentially compromising event detection and segmentation.

Existing feature-preserving compressors typically target a single feature or a family of features with similar structures. Developing these methods often requires feature-specific preservation conditions, modifications to the underlying compressor, or specialized correction procedures (Liang et al., 2023a; Xia et al., 2024; Gong et al., 2023; Yan et al., 2024; Gorski et al., 2025; Li et al., 2025), limiting their reuse across scientific analyses. Even automated approaches such as QPET are restricted to differentiable quantities of interest (Liu et al., 2025).

Recent advances in LLM-based code generation offer an alternative approach to adapting compressors for different scientific features. When a feature can be specified in executable code, an LLM can generate compressor modifications or configurations that are then automatically evaluated for preservation of the target feature. Prior work has explored LLMs for scientific programming, software modification, and program generation guided by executable evaluators (Tian et al., 2024; Jimenez et al., 2024; Romera-Paredes et al., 2024; Novikov et al., 2025). LLM agents have also been applied to optimizing and porting scientific compression kernels (Li et al., 2026a).

To the best of our knowledge, no existing code-generation benchmark evaluates whether generated compressor adaptations preserve scientific features while satisfying compression constraints. Existing benchmarks primarily assess general programming correctness or task-level performance (Chen et al., 2021; Austin et al., 2021; Lai et al., 2023; Jimenez et al., 2024; Ouyang et al., 2025; Chan et al., 2025), leaving feature-preserving scientific compression largely unexplored.

Evaluating feature-preserving compression requires jointly considering feature fidelity, compliance with pointwise error bounds, and storage savings relative to lossless compression. A generated program may execute successfully yet fail to preserve the target feature, violate the prescribed error bound, or incur excessive storage overhead. These interdependent requirements make feature-preserving compression a code-generation setting in which functional correctness alone is insufficient.

We introduce **CraftBench**, an execution-based benchmark for evaluating single-response LLM adaptations of scientific compressors for feature preservation (Figure 2). The current implementation of CraftBench supports SZ3 (Liang et al., 2023b) and ZFP (Lindstrom, 2014), and can be extended to additional compressors by implementing the corresponding compressor interfaces. The benchmark comprises 83 feature-preservation tasks, each defined by an executable feature function and a task-specific distance function that compares feature outputs computed from the original and reconstructed fields. CraftBench currently focuses on three adaptation modes: error bounding (EB), configuration selection (CS), and error correction (EC), which modify pointwise error bounds, compressor settings, and reconstructed values. Each generated adaptation is executed and jointly evaluated based on target-feature preservation, compliance with the pointwise error bound, and compressed size.

CraftBench also evaluates whether a generated adaptation outperforms simply tightening the error bound uniformly across the field. For each task and compressor backend, we construct *uniform-error-bound references* by recompressing the data with progressively smaller uniform error bounds. An adaptation fails this criterion if a uniform-error-bound reference is no worse in both feature distance and compressed size and strictly better in at least one. The prompt specifies this criterion but omits the reference values because obtaining them requires multiple additional compression runs that a user would not typically perform before requesting an adaptation.

Across 2,475 benchmark instances, 282 adaptations (11.4%) satisfy all criteria. 509 adaptations meet the feature-distance reduction, pointwise-error, and storage requirements but are dominated by at least one uniform-error-bound reference. Thus, improving the target feature while satisfying compression constraints does not necessarily outperform simply tightening the global error bound.

Our main contributions are:

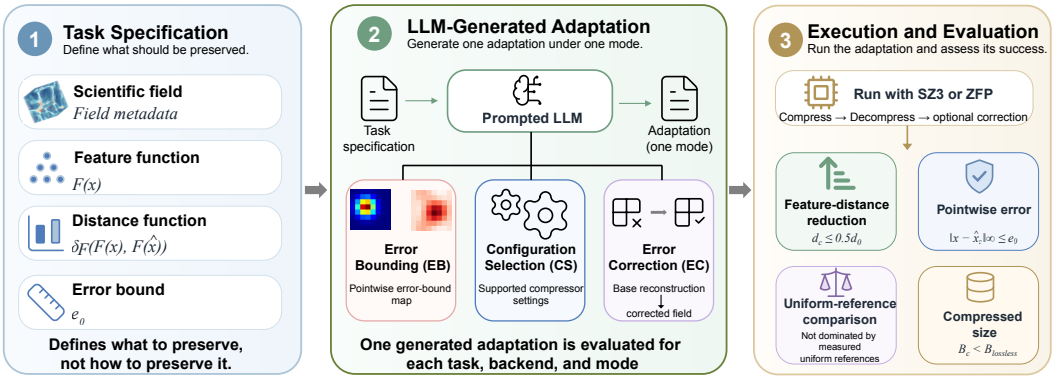


Figure 2: Overview of CraftBench. Each task specifies a scientific field, an executable feature function, a distance function, and a pointwise error bound. An LLM generates one of three types of compressor adaptations, which is executed with SZ3 or ZFP and assessed against four success criteria.

- **A new execution-based benchmark.** We introduce **CraftBench**, the first benchmark for evaluating LLM-generated adaptations for feature-preserving scientific compression. It comprises 83 tasks defined by executable feature and distance functions and supports three adaptation modes.
- **A compression-aware evaluation protocol.** We jointly assess feature-distance reduction, compliance with the pointwise error bound, compressed size, and advantage over uniform-error-bound references obtained by uniformly tightening the global error bound. This protocol distinguishes genuinely effective adaptations from those whose benefits a simple baseline can match.
- **A large-scale empirical study.** We evaluate five LLMs across 2,475 benchmark instances, systematically analyzing performance across scientific features/tasks, compressor backends, and adaptation modes, and identifying the primary causes of failure.

2 Related Work

Scientific data compression. Compression methods broadly fall into lossless and lossy categories. Lossless compression reconstructs data exactly but generally provides limited compression for floating-point scientific data (Lindstrom & Isenburg, 2006; Burtscher & Ratanaworabhan, 2009). Lossy compression achieves higher compression ratios by allowing errors between the original and reconstructed data. Error-bounded lossy compression further restricts the pointwise reconstruction errors according to a user-specified error bound (Cappello et al., 2019; Zhao et al., 2020a).

Representative error-bounded compressors use prediction-, transform-, or neural-network-based methods. Prediction-based compressors, such as SZ, predict data values from neighboring information and quantize the prediction residuals (Tao et al., 2017; Di & Cappello, 2016; 2018; Zhao et al., 2020b; 2021; Liang et al., 2023b; Liu et al., 2022; Tian et al., 2020; 2021; Huang et al., 2023). Transform-based compressors apply block transforms, multilevel decompositions, or tensor decompositions before encoding the resulting coefficients (Lindstrom, 2014; Liang et al., 2022; Ballester-Ripoll et al., 2020; Li et al., 2023). Neural-network-based compressors use learned models to predict or reconstruct data values and encode the remaining residuals under the prescribed error bound (Liu et al., 2021; 2023a). Several methods also automate the selection of compressor components or parameters to optimize compression ratio, bitrate, or reconstruction quality (Underwood et al., 2022; Liu et al., 2023b; 2024). See Di et al. (2025) for a survey. In contrast, CraftBench evaluates generated adaptations for task-specific executable feature definitions.

Feature-preserving compression. Most existing feature-preserving compressors target a specific feature or a narrow feature family and follow three broad approaches. The first derives a spatially varying error-bound map from the target feature definition and applies it to an existing error-bounded compressor. For example, Liang et al. derive cell-wise error bounds for vector-field critical points and aggregate them into a vertex-wise error-bound map (Liang et al., 2023a); Xia et al. derive local bounds from conditions that preserve determinant signs (Xia et al., 2024; 2025); and QPET automatically estimates pointwise bounds for differentiable univariate and multivariate quantities of interest (Liu et al., 2025). The second approach integrates feature-specific constraints directly into

the compression algorithm. TopoSZ derives constraints from contour-tree-induced segmentations and enforces them through error-controlled quantization (Yan et al., 2024). The third approach leaves the base compressor unchanged and adjusts reconstructed data. Gorski et al. (2025) compute and encode adjustments to preserve contour trees for multiple lossy compressors. Other edit-based methods iteratively correct the reconstructed data within the prescribed pointwise error bound until the target feature is preserved (Li et al., 2025; 2026b;c;d).

The EB and EC modes in CraftBench correspond to adaptation before compression and after decompression. CS adapts the settings exposed by a compressor without modifying its algorithm.

LLMs for scientific compression. Prior work on LLMs for lossy compression has focused on execution performance rather than preserving task-specific features. Li et al. (2026a) evaluate LLM coding agents for optimizing and porting SZ-family compression kernels across different hardware architectures, measuring execution success and throughput. CraftBench instead evaluates whether LLM-generated compressor adaptations preserve specified scientific features, satisfy the pointwise error bound, and provide meaningful data compression.

Code-generation benchmarks. Most code-generation benchmarks evaluate functional correctness or performance, without jointly requiring feature-distance reduction, pointwise-error compliance, and data reduction. HumanEval and MBPP assess standalone programming problems (Chen et al., 2021; Austin et al., 2021), DS-1000 evaluates data-science code using widely adopted Python libraries (Lai et al., 2023), and BigCodeBench emphasizes complex instructions and the compositional use of function calls from multiple libraries (Zhuo et al., 2025). SciCode evaluates research-level scientific coding problems decomposed into subproblems (Tian et al., 2024), and SWE-bench evaluates repository-level issue resolution in existing software projects (Jimenez et al., 2024). Other benchmarks score generated programs by execution speed or competition metrics, including GPU-kernel optimization and machine-learning engineering (Ouyang et al., 2025; Chan et al., 2025). LLM-guided search systems such as FunSearch and AlphaEvolve further refine programs against executable evaluators over multiple iterations (Romera-Paredes et al., 2024; Novikov et al., 2025).

3 Benchmark Construction

In this section, we describe the construction of the CraftBench tasks. Each task is constructed by selecting a scientific feature, defining executable functions to compute and compare it, pairing it with compatible scientific data, and creating prompts tailored to each compressor and adaptation mode. The executable definitions specify what must be preserved without prescribing how to preserve it.

We implement the executable feature and distance functions according to the scientific definitions of the selected features. We manually review each implementation against its mathematical definition and verify its execution and outputs on the benchmark data. Appendix B provides the feature definitions. We select tasks to broadly represent features examined in prior compression and application studies, including value distributions, local derivatives and spatial statistics, geometric and topological structures, spectral and multiscale features, composite features, and domain-specific quantities.

3.1 Scientific Features and Task Definitions

The features evaluated in CraftBench are motivated by prior scientific compression studies, application studies, and feature-preserving compression methods (Di et al., 2025; Cappello et al., 2025; Soler et al., 2018; Liang et al., 2023a; Xia et al., 2024; Gong et al., 2023; Yan et al., 2024; Li et al., 2025; 2026b;c;d). We evaluate these features on floating-point scientific data from three application domains: turbulence, climate, and cosmology (Kalnay et al., 1996; Almgren et al., 2013; Zhao et al., 2020a). Specifically, the benchmark uses turbulence enstrophy from the OME-Zarr Open SciVis collection (McCormick, 2026), NCEP/NCAR air temperature for climate (Kalnay et al., 1996), and Nyx density from SDRBench for cosmology (Zhao et al., 2020a). Multiple tasks may use the same scientific field while evaluating different features. General-purpose tasks use any fields compatible with the feature definition, whereas domain-specific tasks use variables with the required scientific meaning. Data details are provided in Appendix C.

Each CraftBench task targets a specific scientific feature and is evaluated on a compatible scientific field. Because these features produce different types of outputs, each task uses a feature-specific

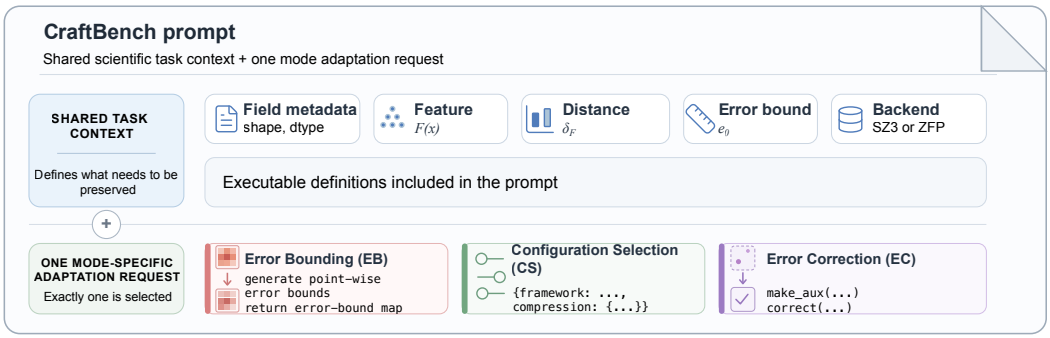


Figure 3: Structure of a CraftBench prompt. All adaptation modes share the task, executable feature, distance functions, and pointwise error bound, but each requests a different compressor adaptation.

distance function rather than a single metric shared across all tasks. CraftBench provides executable functions for computing the target feature from a data field and measuring the difference between feature outputs from the original and reconstructed fields.

3.2 Prompt Construction

For each task, CraftBench constructs a prompt that includes dataset metadata (i.e., the variable, units, preprocessing, shape, and data type) along with the executable feature and distance functions, the pointwise error bound e_0, the baseline feature distance d_0, a target distance of $0.5d_0$ requiring a clear improvement over the baseline, the base compressor, the success criteria, and the output format.

The model does not have access to the field values when generating an adaptation. At execution time, however, in the EB mode, the generated function receives the original field and assigns spatially varying error bounds before compression. In the EC mode, the generated function receives both the original field and the baseline reconstruction, then modifies the reconstructed values after decompression. In the CS mode, the model specifies compressor settings, which are used to compress the original field. Figure 3 illustrates the prompt structure, and Appendix D.2 summarizes the prompt templates.

3.3 Benchmark Statistics

CraftBench contains 83 tasks, summarized in Table 1. These include 60 general-purpose tasks spanning four categories: numerical and distributional, differential and spatial, geometric and topological, and spectral and multiscale. The benchmark also includes 15 composite tasks combining multiple general-purpose features and 8 domain-specific tasks associated with scientific variables.

Each model is evaluated on $83 \times 2 \times 3 = 498$ raw benchmark instances, covering 83 tasks, two compressor backends, and three adaptation modes. For ZFP, the baseline reconstruction already achieves zero feature distance for `component_count_minsize`, leaving no room for further feature-distance reduction. We therefore exclude this task from success-rate calculations for all three ZFP adaptation modes, resulting in 495 instances per model.

4 CraftBench Tasks and Adaptation Modes

This section formalizes the feature-preservation tasks constructed in Section 3, describes the three adaptation modes evaluated by CraftBench, and defines the success criteria for adaptations.

4.1 Feature-Preservation Tasks

Each task is defined by a data field x, a feature function F, a distance function δ_F, and a pointwise error bound e_0, as described in Section 3. The function F extracts the feature from a data field, while δ_F compares two feature outputs. Given a reconstructed field $\hat{x}$, the feature distance is

$$d_F(x, \hat{x}) = \delta_F\big(F(x), F(\hat{x})\big), \tag{1}$$

Task group	Tasks	Representative examples
Numerical and distributional	9	Histogram, quantiles, moments
Differential and spatial	24	Gradient magnitude, Laplacian, spatial correlation
Geometric and topological	17	Excursion-set components, persistence, Euler characteristic
Spectral and multiscale	10	Spectral entropy, spectral energy, multiscale residuals
Composite	15	Quantiles + spectral energy; persistence + extrema arrangement
Domain-specific	8	Enstrophy distribution, zonal-mean temperature, density-spectrum slope
Total	83	

Table 1: Task groups in CraftBench with representative examples. The first four groups contain the 60 general-purpose tasks; the remaining tasks are composite or domain-specific.

where smaller values indicate better feature preservation. For composite tasks, CraftBench defines the feature distance as the maximum normalized distance across the component features, preventing a large error in one feature from being masked by better preservation of the others. Appendix B.1 provides the complete task catalog and distance definitions.

4.2 Adaptation Modes

CraftBench currently focuses on three adaptation modes targeting different parts of the compression workflow: error bounding (EB), which generates a pointwise error-bound map before compression; configuration selection (CS), which selects compressor settings; and error correction (EC), which modifies reconstructed values after decompression.

Error bounding (EB). The model generates a spatial error-bound map that assigns an error bound e_i to each data point, where $0 \leq e_i \leq e_0$. A value of $e_i = 0$ requires exact reconstruction at that data point. SZ3 applies the generated bounds directly to individual data points. Because ZFP operates on 4×4 blocks in 2D fields and $4 \times 4 \times 4$ blocks in 3D fields, each ZFP block uses the smallest generated error bound among the data points in that block.

Configuration selection (CS). The model selects a supported compressor configuration. This mode evaluates selection among compressor settings rather than modifications to the internal compression algorithm. For SZ3, the model can configure predictor, interpolation, and quantization settings. For ZFP, the model can select fixed-accuracy, fixed-rate, fixed-precision, or expert mode. In the CS mode, explicit error-bound controls are fixed at the prescribed e_0: SZ3 configurations use e_0, and ZFP fixed-accuracy configurations must also use tolerance e_0. Because ZFP's fixed-rate, fixed-precision, and expert modes do not directly enforce the prescribed error bound, we check their final reconstructions against e_0; configurations that violate e_0 fail the pointwise-error criterion. SZ3 and ZFP expose different configuration spaces, so their configuration success rates are not directly comparable.

Error correction (EC). The model generates a procedure that compares the original field x with the baseline reconstruction $\hat{x}_{\text{base}}$ and produces correction data. During decompression, these data are applied to the reconstruction to obtain a corrected field that satisfies the pointwise error bound. The correction data are stored using the smaller representation (raw or losslessly compressed with Zstd (Collet & Kucherawy, 2021)), and their storage overhead is included in the compressed size.

4.3 Success Criteria

For each task and backend, CraftBench uses a base reconstruction obtained with the prescribed error bound e_0. An adaptation succeeds only if it satisfies: (C1) reduction in target feature distance, (C2) compliance with the pointwise error bound, (C3) non-domination by uniform-error-bound references, and (C4) compressed size smaller than the lossless reference. We define these criteria below.

Feature-distance reduction (C1). Let d_0 denote the feature distance of the base reconstruction and d_c that of the generated adaptation. We require a 50% reduction in feature distance, $d_c \leq 0.5\, d_0$. We use the same relative reduction target for all tasks because feature distances may have different numerical scales. We further report results under 25% and 75% reduction targets in Appendix F.5.

Pointwise error (C2). Let $\hat{x}_c$ denote the final reconstruction produced by the adaptation. It must satisfy the prescribed pointwise error bound, $\|x - \hat{x}_c\|_\infty \leq e_0$.

Uniform-error-bound references (C3). The uniform-error-bound references provide a baseline for determining whether an adaptation provides an advantage over simply tightening the global error bound when both feature preservation and compressed size are considered. The prompt specifies the uniform-error-bound reference criterion and its dominance rule, but does not provide the reference values. Providing these values would require the user to perform a sweep at tighter error bounds and measure the resulting feature distances and compressed sizes. We therefore reserve the reference values for evaluation.

For each task and compressor, CraftBench constructs 25 uniform-error-bound references using

$$e_k = e_0\, 2^{-k/4}, \qquad k = 0, \ldots, 24, \tag{2}$$

covering global error bounds from e_0 to $e_0/64$. The same references are used across all adaptation modes, and the $k = 0$ reference coincides with the base reconstruction. Let d_k and B_k denote the feature distance and compressed size of reference k, and let B_c denote the total stored size of the generated adaptation. A reference dominates an adaptation if

$$d_k \leq d_c, \qquad B_k \leq B_c, \qquad \text{and} \qquad \big(d_k < d_c \;\vee\; B_k < B_c\big). \tag{3}$$

An adaptation passes the comparison only if none of the uniform-error-bound references dominate it.

Compressed size (C4). The final compressed size must satisfy $B_c < B_{\text{lossless}}$, where B_{lossless} is the size of the original field compressed losslessly with FPZIP (Lindstrom & Isenburg, 2006). For every adaptation mode, B_c includes all stored data required to reconstruct the final field. In the EB mode, B_c includes the lossy compressed data and stored error-bound information; in the CS mode, B_c includes the compressed stream produced by the selected configuration; and in the EC mode, B_c includes the base lossy stream and the losslessly compressed correction data.

5 Results

5.1 Experimental Setup

We evaluate five models: Gemini-3.8-Flash (Google, 2026), Claude-Sonnet-5 (Anthropic, 2026), Kimi-K2.5 (Kimi Team et al., 2026), GPT-5.3-Codex (OpenAI, 2026), and Qwen3-Coder-Next (Cao et al., 2026). Model settings are provided in Appendix E. Each model has 495 evaluated instances after excluding three ZFP cases with zero baseline feature distance. We use $e_0 = 0.03(\max(x) - \min(x))$ throughout; CraftBench can also be evaluated at other error bounds.

We use one model generation per benchmark instance. Deterministic parsing or format recovery may use the same output, but compilation, execution, or evaluation failures do not trigger regeneration. Generated programs run on CPU jobs with four cores and 16 GiB of memory.

5.2 Overall Performance

Observation. Overall, 282 of 2,475 adaptations (11.4%) succeed, with model-level rates ranging from 4.4% to 17.2%.

Figure 4 compares success rates across models, compressor backends, and adaptation modes. 1997 instances (80.7%) execute successfully but fail one or more criteria, while 196 instances (7.9%) fail before producing a valid result. Appendix F.2 provides a detailed breakdown of these failures.

Across the 495 evaluation settings, 173 (34.9%) are successful for at least one of the five models. Gemini-3.8-Flash obtains 85 successes among 495 instances (17.2%), Claude-Sonnet-5 obtains 83 (16.8%), Kimi-K2.5 obtains 59 (11.9%), GPT-5.3-Codex obtains 33 (6.7%), and Qwen3-Coder-Next obtains 22 (4.4%). Among the successful adaptations, the median ratio of the final compressed size B_c to the base compressed size B_0 is 1.42, with the middle 50% ranging from 1.16 to 2.02.

5.3 Behavior Across Compressors and Adaptation Modes

Observation. EC achieves the highest overall success rate at 14.7%, compared with 11.8% for EB and 7.8% for CS.

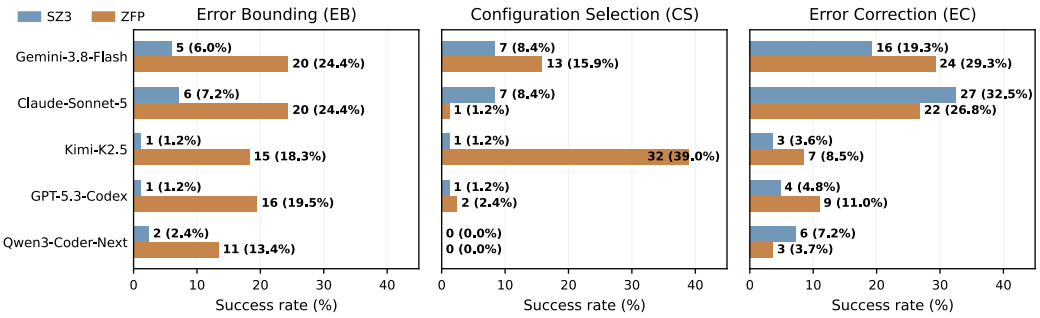


Figure 4: Success rates across five models, two compressor backends, and three adaptation modes. Labels show successes and percentages. SZ3 uses 83 tasks per model and mode, while ZFP uses 82 after excluding `component_count_minsize` with $d_0 = 0$.

Kimi-K2.5 succeeds on 32 of 82 ZFP configuration instances, including 30 expert-mode successes. These successes use 8 distinct parameter combinations, and 15 of the 30 use the same setting: `minbits` = 0, `maxbits` = 0, `maxprec` = 16, and `minexp` = -20. To examine whether task-specific configuration selection contributes to this performance, we apply the most frequent successful configuration unchanged to all 82 instances. The fixed configuration succeeds on 34 of 82 instances. This result indicates that Kimi-K2.5's success rate does not by itself demonstrate a benefit from task-specific configuration selection. We provide a more detailed comparison in Appendix F.7.

Qwen3-Coder-Next selects ZFP's fixed-accuracy mode in all 82 instances and obtains no successes. With the error bound fixed at e_0, fixed-accuracy reproduces the base reconstruction and cannot improve the feature distance. In 78 of 82 ZFP configuration instances, Claude-Sonnet-5 selects fixed-accuracy mode with an error bound below e_0. ZFP fixed-accuracy configurations are required to use the error bound e_0. Allowing a smaller error bound would reproduce a uniform-tightening reference, so the adaptation would have identical feature distance and compressed size and would not be considered dominated under C3. We therefore treat these outputs as invalid configurations.

5.4 Results Across Task Groups

Observation. Success rates range from 7.7% to 20.0% across task groups, with domain-specific tasks achieving 20.0% and geometric/topological tasks 17.0%.

Table 2 reports success rates across the six task groups. The remaining four groups all have success rates below 12%, indicating that the higher rates are concentrated in the domain-specific and geometric/topological groups. Because the groups differ in task composition and feature definitions, these results should not be interpreted as differences in intrinsic difficulty.

Task group	Tasks	Successes	Rate
Numerical and distributional	9	31 / 270	11.5%
Differential and spatial	24	57 / 720	7.9%
Geometric and topological	17	84 / 495	17.0%
Spectral and multiscale	10	23 / 300	7.7%
Composite	15	39 / 450	8.7%
Domain-specific	8	48 / 240	20.0%

Table 2: Success rates across the six CraftBench task groups, aggregated over models, compressor backends, and adaptation modes.

5.5 Analysis of Unsuccessful Adaptations

Observation. Single-criterion failures are concentrated in C1 and C3: 552 adaptations fail only C1, 509 fail only C3, 1 fails only C2, and 26 fail only C4.

Insufficient feature-distance reduction. This failure is especially common for the SZ3 configuration, where 397 of 399 unsuccessful adaptations do not achieve the required feature-distance reduction. To distinguish model-selection failures from limitations of the configuration space, we evaluate all legal SZ3 configurations for each task. We do not exhaustively search the ZFP CS, EB, or EC adaptation spaces because they are too large or infinite. As shown in Table 3, 71 of 83 tasks have no configuration that achieves the required 50% feature-distance reduction.

Table 3: Achievability under the SZ3 configuration space.

Outcome	Tasks	Fraction
Cannot meet feature criterion	71	85.5%
Meet feature and error criteria, but fail C3	2	2.4%
Meet all four criteria	10	12.0%
Total	83	100.0%

Configuration failures. Invalid ZFP configurations also arise from generated settings outside the configuration space allowed by CraftBench, as further discussed in Section 5.3.

Uniform-reference domination. Another 509 adaptations (353 EB, 93 EC, and 63 CS) satisfy the feature-distance reduction, pointwise-error, and storage criteria but are dominated by at least one measured uniform-error-bound reference. Most dominating references are strictly better in feature distance and compressed size: 306 of 353 EB failures, 81 of 93 EC failures, and 37 of 63 CS failures. The uniform-reference criterion and dominance rule are in the prompt, but the measured reference values are not provided to the model. Because the official C3 sweep ends at $e_0/64$, tighter uniform bounds could still dominate some successful candidates. We therefore extend the sweep to $e_0/4096$; only 10 of the 282 successes become dominated under this additional analysis (Appendix F.8).

Correction storage. Successful correction adaptations are sparser and require less correction storage than unsuccessful ones, with medians of 189 versus 512 modified points and 663 versus 2057 stored bytes. Among the 25 correction adaptations that satisfy the other three criteria but fail the lossless-size criterion, 23 are generated by GPT-5.3-Codex (Appendix F.4).

6 Conclusion and Limitations

We introduce CraftBench, an execution-based benchmark for evaluating LLM-generated adaptations of error-bounded lossy compressors to preserve scientific features specified in executable code. Across 2,475 instances, 282 (11.4%) satisfy all four criteria. Another 509 satisfy the feature-distance reduction, pointwise-error, and storage criteria but are dominated by a uniform-error-bound reference.

Success rates vary across models and adaptation settings. For the SZ3 CS mode, we evaluate every legal configuration supported by CraftBench and find that only 10 of the 83 tasks admit at least one configuration satisfying all four criteria. Across these 10 tasks, 16 of the 50 model-generated configurations (32.0%) are successful. Thus, the low overall success rate for SZ3 configuration reflects both the limited adaptation space exposed by this mode and the models' imperfect selection among configurations for tasks where a successful solution exists.

Several limitations constrain the interpretation of these results. First, CraftBench covers 83 tasks and two compressors but uses only three fixed evaluation fields, with 74 tasks evaluated on the NCEP/NCAR field. Appendix F.9 additionally evaluates the 2D adaptations on a larger field, but generalization across a broader range of datasets and compression methods remains to be studied. Second, we evaluate only one pointwise error bound and one generation per instance; results at other e_0 values and sampling variability remain unquantified. Third, C3 compares each adaptation against uniform-error-bound references obtained by uniformly tightening the fixed-accuracy bound, but these reference values are not available to the model. Satisfying C3 therefore neither establishes that the improvement arises from task-specific adaptation nor rules out domination by other global ZFP configurations. Although we exhaustively evaluate the finite SZ3 configuration space in Appendix F.6, we do not characterize the achievable success rates for ZFP CS, EB, or EC. Finally, satisfying the executable criteria demonstrates feature preservation under the benchmark conditions but does not provide the formal guarantees offered by manually designed feature-preserving compressors.

Reproducibility Statement

CraftBench specifies the benchmark tasks, adaptation interfaces, and success criteria in Sections 3 and 4. Appendix B provides the feature and distance definitions, Appendix C describes the datasets and preprocessing, Appendix D provides the execution protocol, prompt templates, and evaluation procedure, and Appendix E reports the model names and generation settings used in the evaluation. Additional result breakdowns are provided in Appendix F. We will release the CraftBench implementation, benchmark tasks, prompt templates, generated model outputs, and evaluation metadata.

AI Use Statement

We used generative AI tools to assist with literature and repository searches, coding, debugging, and code review for the benchmark, and with polishing the manuscript to improve clarity and readability. All AI-assisted code and text were reviewed by the authors. The research methodology, experimental design, analyses, results, and conclusions were developed and verified by the authors.

References

Ann S. Almgren, John B. Bell, Mike J. Lijewski, Zarija Lukić, and Ethan Van Andel. Nyx: A massively parallel AMR code for computational cosmology. *The Astrophysical Journal*, 765 (1):39, 2013. doi: 10.1088/0004-637X/765/1/39. URL https://doi.org/10.1088/0004-637X/765/1/39.

Anthropic. Introducing Claude Sonnet 5. https://www.anthropic.com/news/claude-sonnet-5, June 2026.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc V. Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. doi: 10.48550/arXiv.2108.07732.

Rafael Ballester-Ripoll, Peter Lindstrom, and Renato Pajarola. TTHRESH: Tensor compression for multidimensional visual data. *IEEE Transactions on Visualization and Computer Graphics*, 26(9): 2891–2903, 2020. doi: 10.1109/TVCG.2019.2904063.

Martin Burtscher and Paruj Ratanaworabhan. FPC: A high-speed compressor for double-precision floating-point data. *IEEE Transactions on Computers*, 58(1):18–31, 2009. doi: 10.1109/TC.2008.131.

Ruisheng Cao, Mouxiang Chen, Jiawei Chen, Zeyu Cui, Yunlong Feng, Binyuan Hui, Yuheng Jing, Kaixin Li, Mingze Li, Junyang Lin, Zeyao Ma, Kashun Shum, Xuwu Wang, Jinxi Wei, Jiaxi Yang, Jiajun Zhang, Lei Zhang, Zongmeng Zhang, Wenting Zhao, and Fan Zhou. Qwen3-Coder-Next technical report. *arXiv preprint arXiv:2603.00729*, 2026.

Franck Cappello, Sheng Di, Sihuan Li, Xin Liang, Ali Murat Gok, Dingwen Tao, Chun Hong Yoon, Xin-Chuan Wu, Yuri Alexeev, and Frederic T. Chong. Use cases of lossy compression for floating-point data in scientific data sets. *The International Journal of High Performance Computing Applications*, 33(6):1201–1220, 2019. doi: 10.1177/1094342019853336.

Franck Cappello, Robert Underwood, Yuri Alexeev, Alison Baker, Ebru Bozdağ, Martin Burtscher, Kyle Chard, Jackie Chen, Sheng Di, Kyle Gerard Felker, Paul Christopher O'Grady, Hanqi Guo, Yafan Huang, Sian Jin, Peng Jiang, Petter Johansson, Shaomeng Li, Xin Liang, Erik Lindahl, Peter Lindstrom, Zarija Lukić, Magnus Lundborg, Danylo Lykov, Masaru Nagaso, Kento Sato, Amarjit Singh, Seung Woo Son, Shihui Song, William Tang, Dingwen Tao, Jiannan Tian, Kazutomo Yoshii, and Kai Zhao. What to support when you're compressing: The state of practice, gaps, and opportunities for scientific data compression. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1966–1979, 2025. doi: 10.1145/3712285.3759856.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *The Thirteenth International Conference on Learning Representations*, 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, et al. Evaluating large language models trained on code. 2021.

Yann Collet and Murray S. Kucherawy. Zstandard compression and the 'application/zstd' media type. Technical Report RFC 8878, Internet Engineering Task Force, February 2021.

Sheng Di and Franck Cappello. Fast error-bounded lossy HPC data compression with SZ. In *2016 IEEE International Parallel and Distributed Processing Symposium*, pp. 730–739, 2016. doi: 10.1109/IPDPS.2016.11.

Sheng Di and Franck Cappello. Optimization of error-bounded lossy compression for hard-to-compress HPC data. *IEEE Transactions on Parallel and Distributed Systems*, 29(1):129–143, 2018. doi: 10.1109/TPDS.2017.2749300.

Sheng Di, Jinyang Liu, Kai Zhao, Xin Liang, Robert Underwood, Zhaorui Zhang, Milan Shah, Yafan Huang, Jiajun Huang, Xiaodong Yu, Congrong Ren, Hanqi Guo, Grant Wilkins, Dingwen Tao, Jiannan Tian, Sian Jin, Zizhe Jian, Daoce Wang, Md Hasanur Rahman, Boyuan Zhang, Shihui Song, Jon Calhoun, Guanpeng Li, Kazutomo Yoshii, Khalid Ayed Alharthi, and Franck Cappello. A survey on error-bounded lossy compression for scientific datasets. *ACM Computing Surveys*, 57 (11):287:1–287:38, 2025. doi: 10.1145/3733104.

D. Donnelly and B. W. Rust. The fast Fourier transform for experimentalists, part I: Concepts. *Computing in Science & Engineering*, 7(2):80–88, 2005. doi: 10.1109/MCSE.2005.42.

G.E. Elsinga, T. Ishihara, and J.C.R. Hunt. Intermittency across Reynolds numbers – the influence of large-scale shear layers on the scaling of the enstrophy and dissipation in homogenous isotropic turbulence. *Journal of Fluid Mechanics*, 974:A17, 2023. doi: 10.1017/jfm.2023.799.

Qian Gong, Chengzhu Zhang, Xin Liang, Viktor Reshniak, Jieyang Chen, Anand Rangarajan, Sanjay Ranka, Nicolas Vidal, Lipeng Wan, Paul Ullrich, Norbert Podhorszki, Robert Jacob, and Scott Klasky. Spatiotemporally adaptive compression for scientific dataset with feature preservation – a case study on simulation data with extreme climate events analysis. In *2023 IEEE 19th International Conference on e-Science (e-Science)*, pp. 1–10, 2023. doi: 10.1109/e-Science58273.2023.10254796.

Google. Introducing Gemini 3.8 Flash and 3.8 Flash Cyber. `https://blog.google/innovation-and-ai/models-and-research/gemini-models/3-8-flash-and-3-8-flash-cyber/`, September 2026.

Nathaniel Gorski, Xin Liang, Hanqi Guo, Lin Yan, and Bei Wang. A General Framework for Augmenting Lossy Compressors With Topological Guarantees . *IEEE Transactions on Visualization & Computer Graphics*, 31(06):3693–3705, June 2025. ISSN 1941-0506. doi: 10.1109/TVCG.2025.3567054.

Yafan Huang, Sheng Di, Xiaodong Yu, Guanpeng Li, and Franck Cappello. cuSZp: An ultra-fast GPU error-bounded lossy compression framework with optimized end-to-end performance. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 43:1–43:13, 2023. doi: 10.1145/3581784.3607048.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=VTF8yNQM66`.

E. Kalnay, M. Kanamitsu, R. Kistler, W. Collins, D. Deaven, L. Gandin, M. Iredell, S. Saha, G. White, J. Woollen, Y. Zhu, M. Chelliah, W. Ebisuzaki, W. Higgins, J. Janowiak, K. C. Mo, C. Ropelewski, J. Wang, A. Leetmaa, R. Reynolds, Roy Jenne, and Dennis Joseph. The ncep/ncar 40-year

reanalysis project. *Bulletin of the American Meteorological Society*, 77(3):437 – 472, 1996. doi: 10.1175/1520-0477(1996)077⟨0437:TNYRP⟩2.0.CO;2.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, et al. Kimi K2.5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-Tau Yih, Daniel Fried, Sida Wang, and Tao Yu. DS-1000: A natural and reliable benchmark for data science code generation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18319–18345, 2023. URL https://proceedings.mlr.press/v202/lai23b.html.

Changqing Li, Shouwei Gao, Kai Zhao, Sheng Di, and Wenqian Dong. Evaluating LLM coding agents on SZ-family lossy compression across architectures. *arXiv*, 2026a. doi: 10.48550/arXiv.2606.17058.

Shaomeng Li, Peter Lindstrom, and John Clyne. Lossy scientific data compression with SPERR. In *2023 IEEE International Parallel and Distributed Processing Symposium*, pp. 1007–1017, 2023. doi: 10.1109/IPDPS54959.2023.00104.

Yuxiao Li, Xin Liang, Bei Wang, Yongfeng Qiu, Lin Yan, and Hanqi Guo. MSz: An efficient parallel algorithm for correcting Morse–Smale segmentations in error-bounded lossy compressors. *IEEE Transactions on Visualization and Computer Graphics*, 31(1):130–140, 2025. doi: 10.1109/TVCG.2024.3456337.

Yuxiao Li, Mingze Xia, Xin Liang, Bei Wang, and Hanqi Guo. Preserving discrete Morse–Smale complexes in error-bounded lossy compression. *IEEE Transactions on Visualization and Computer Graphics*, 32(7):6593–6609, 2026b. doi: 10.1109/TVCG.2026.3684385.

Yuxiao Li, Mingze Xia, Xin Liang, Bei Wang, and Hanqi Guo. EXaCTz: Guaranteed extremum graph and contour tree preservation for distributed- and GPU-parallel lossy compression. *arXiv preprint arXiv:2604.01397*, 2026c. doi: 10.48550/arXiv.2604.01397.

Yuxiao Li, Mingze Xia, Xin Liang, Bei Wang, Robert Underwood, Sheng Di, Hemant Sharma, Dishant Beniwal, Franck Cappello, and Hanqi Guo. pMSz: A distributed parallel algorithm for correcting extrema and Morse-Smale segmentations in lossy compression. In *2026 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 159–172, Los Alamitos, CA, USA, May 2026d. IEEE Computer Society. doi: 10.1109/IPDPS65963.2026.00025.

Xin Liang, Ben Whitney, Jieyang Chen, Lipeng Wan, Qing Liu, Dingwen Tao, James Kress, David Pugmire, Matthew Wolf, Norbert Podhorszki, and Scott Klasky. MGARD+: Optimizing multilevel methods for error-bounded scientific data reduction. *IEEE Transactions on Computers*, 71(7):1522–1536, 2022. doi: 10.1109/TC.2021.3092201.

Xin Liang, Sheng Di, Franck Cappello, Mukund Raj, Chunhui Liu, Kenji Ono, Zizhong Chen, Tom Peterka, and Hanqi Guo. Toward feature-preserving vector field compression. *IEEE Transactions on Visualization and Computer Graphics*, 29(12):5434–5450, 2023a. doi: 10.1109/TVCG.2022.3214821.

Xin Liang, Kai Zhao, Sheng Di, Sihuan Li, Robert Underwood, Ali M. Gok, Jiannan Tian, Junjing Deng, Jon C. Calhoun, Dingwen Tao, Zizhong Chen, and Franck Cappello. SZ3: A modular framework for composing prediction-based error-bounded lossy compressors. *IEEE Transactions on Big Data*, 9(2):485–498, 2023b. doi: 10.1109/TBDATA.2022.3201176.

Peter Lindstrom. Fixed-rate compressed floating-point arrays. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2674–2683, 2014. doi: 10.1109/TVCG.2014.2346458.

Peter Lindstrom and Martin Isenburg. Fast and efficient compression of floating-point data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1245–1250, 2006. doi: 10.1109/TVCG.2006.143.

Jinyang Liu, Sheng Di, Kai Zhao, Sian Jin, Dingwen Tao, Xin Liang, Zizhong Chen, and Franck Cappello. Exploring autoencoder-based error-bounded compression for scientific data. In *2021 IEEE International Conference on Cluster Computing*, pp. 294–306, 2021. doi: 10.1109/Cluster48925.2021.00034.

Jinyang Liu, Sheng Di, Kai Zhao, Xin Liang, Zizhong Chen, and Franck Cappello. Dynamic quality metric oriented error bounded lossy compression for scientific datasets. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15, 2022. doi: 10.1109/SC41404.2022.00067.

Jinyang Liu, Sheng Di, Sian Jin, Kai Zhao, Xin Liang, Zizhong Chen, and Franck Cappello. Scientific error-bounded lossy compression with super-resolution neural networks. In *2023 IEEE International Conference on Big Data (BigData)*, pp. 229–236, 2023a. doi: 10.1109/BigData59044.2023.10386682.

Jinyang Liu, Sheng Di, Kai Zhao, Xin Liang, Zizhong Chen, and Franck Cappello. FAZ: A flexible auto-tuned modular error-bounded compression framework for scientific data. In *Proceedings of the 37th International Conference on Supercomputing*, pp. 1–13, 2023b. doi: 10.1145/3577193.3593721.

Jinyang Liu, Sheng Di, Kai Zhao, Xin Liang, Sian Jin, Zizhe Jian, Jiajun Huang, Shixun Wu, Zizhong Chen, and Franck Cappello. High-performance effective scientific error-bounded lossy compression with auto-tuned multi-component interpolation. *Proceedings of the ACM on Management of Data*, 2(1):1–27, 2024. doi: 10.1145/3639259.

Jinyang Liu, Pu Jiao, Kai Zhao, Xin Liang, Sheng Di, and Franck Cappello. QPET: A versatile and portable quantity-of-interest-preservation framework for error-bounded lossy compression. *Proc. VLDB Endow.*, 18(8):2440–2453, 2025. ISSN 2150-8097. doi: 10.14778/3742728.3742739.

Matthew McCormick. OME-Zarr Open SciVis Datasets, 2026. Version 2026.02.14.

NOAA National Centers for Environmental Information. Global forecast system (gfs). `https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast`, 2020. Accessed: 2026-09-23.

Alexander Novikov, Ngan Vu, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. AlphaEvolve: A coding agent for scientific and algorithmic discovery, 2025. URL `https://arxiv.org/abs/2506.13131`.

OpenAI. Introducing GPT-5.3-Codex. `https://openai.com/index/introducing-gpt-5-3-codex/`, February 2026.

Anne Ouyang, Simon Guo, Simran Arora, Alex L Zhang, William Hu, Christopher Re, and Azalia Mirhoseini. KernelBench: Can LLMs write efficient GPU kernels? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 47356–47415. PMLR, 2025. URL `https://proceedings.mlr.press/v267/ouyang25a.html`.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6.

Maxime Soler, Mélanie Plainchault, Bruno Conche, and Julien Tierny. Topologically controlled lossy compression. In *2018 IEEE Pacific Visualization Symposium*, pp. 46–55, 2018. doi: 10.1109/PacificVis.2018.00015.

Dingwen Tao, Sheng Di, Zizhong Chen, and Franck Cappello. Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization. In *2017 IEEE International Parallel and Distributed Processing Symposium*, pp. 1129–1139, 2017. doi: 10.1109/IPDPS.2017.115.

Jiannan Tian, Sheng Di, Kai Zhao, Cody Rivera, Megan Hickman Fulp, Robert Underwood, Sian Jin, Xin Liang, Jon Calhoun, Dingwen Tao, and Franck Cappello. cuSZ: An efficient GPU-based error-bounded lossy compression framework for scientific data. In *Proceedings of the 2020 International Conference on Parallel Architectures and Compilation Techniques*, pp. 3–15, 2020. doi: 10.1145/3410463.3414624.

Jiannan Tian, Sheng Di, Xiaodong Yu, Cody Rivera, Kai Zhao, Sian Jin, Yunhe Feng, Xin Liang, Dingwen Tao, and Franck Cappello. Optimizing error-bounded lossy compression for scientific data on GPUs. In *2021 IEEE International Conference on Cluster Computing*, pp. 283–293, 2021. doi: 10.1109/Cluster48925.2021.00047.

Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. SciCode: A research coding benchmark curated by scientists. In *Advances in Neural Information Processing Systems*, volume 37, pp. 30624–30650, 2024. doi: 10.52202/079017-0963.

Robert Underwood, Jon C. Calhoun, Sheng Di, Amy Apon, and Franck Cappello. OptZConfig: Efficient parallel optimization of lossy compression configuration. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):3505–3519, 2022. doi: 10.1109/TPDS.2022.3154096.

David H. Weinberg, Michael J. Mortonson, Daniel J. Eisenstein, Christopher Hirata, Adam G. Riess, and Eduardo Rozo. Observational probes of cosmic acceleration. *Physics Reports*, 530(2):87–255, 2013. doi: 10.1016/j.physrep.2013.05.001.

Mingze Xia, Sheng Di, Franck Cappello, Pu Jiao, Kai Zhao, Jinyang Liu, Xuan Wu, Xin Liang, and Hanqi Guo. Preserving topological feature with sign-of-determinant predicates in lossy compression: A case study of vector field critical points. In *2024 IEEE 40th International Conference on Data Engineering*, pp. 4979–4992, 2024. doi: 10.1109/ICDE60146.2024.00378.

Mingze Xia, Bei Wang, Yuxiao Li, Pu Jiao, Xin Liang, and Hanqi Guo. TspSZ: An efficient parallel error-bounded lossy compressor for topological skeleton preservation. In *2025 IEEE 41st International Conference on Data Engineering*, pp. 3682–3695, 2025. doi: 10.1109/ICDE65448.2025.00275.

Lin Yan, Xin Liang, Hanqi Guo, and Bei Wang. TopoSZ: Preserving topology in error-bounded lossy compression. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1302–1312, 2024. doi: 10.1109/TVCG.2023.3326920.

Kai Zhao, Sheng Di, Xin Liang, Sihuan Li, Dingwen Tao, Julie Bessac, Zizhong Chen, and Franck Cappello. SDRBench: Scientific data reduction benchmark for lossy compressors. In *International Workshop on Big Data Reduction (IWBDR), in conjunction with the 2020 IEEE International Conference on Big Data*, pp. 2716–2724, 2020a. doi: 10.1109/BigData50022.2020.9378449.

Kai Zhao, Sheng Di, Xin Liang, Sihuan Li, Dingwen Tao, Zizhong Chen, and Franck Cappello. Significantly improving lossy compression for HPC datasets with second-order prediction and parameter optimization. In *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, pp. 89–100, 2020b. doi: 10.1145/3369583.3392688.

Kai Zhao, Sheng Di, Maxim Dmitriev, Thierry-Laurent D. Tonellot, Zizhong Chen, and Franck Cappello. Optimizing error-bounded lossy compression for scientific data by dynamic spline interpolation. In *2021 IEEE 37th International Conference on Data Engineering*, pp. 1643–1654, 2021. doi: 10.1109/ICDE51399.2021.00145.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen Gong, Thong Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kaddour, Ming Xu, Zhihan Zhang, Prateek Yadav, Naman Jain, Alex Gu, Zhoujun Cheng, Jiawei Liu, Qian Liu, Zijian Wang, David Lo, Binyuan Hui, Niklas Muennighoff, Daniel Fried, Xiaoning Du, Harm de Vries, and Leandro von Werra. BigCodeBench: Benchmarking code generation with diverse function calls and complex instructions. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=YrycTjllL0.

A Compression Backends

CraftBench uses SZ3 (Liang et al., 2023b) and ZFP (Lindstrom, 2014) as base compressors. For each task, the base compressor is executed with the prescribed pointwise error bound to produce the base reconstruction. CraftBench then computes its feature distance and compressed size.

SZ3 supports pointwise error bounds and exposes configurable prediction, interpolation, and quantization components. ZFP operates on 4×4 blocks in two dimensions and $4 \times 4 \times 4$ blocks in three dimensions and supports fixed-accuracy, fixed-rate, fixed-precision, and expert modes. Fixed-accuracy directly specifies an error bound, whereas the other ZFP modes expose rate, precision, or lower-level compression parameters; their final reconstructions are therefore explicitly checked against the prescribed pointwise error bound.

For all adaptation modes, the compressed size includes the complete stored data required to reconstruct the final field. For an uncompressed float32 field of size B_{raw} and a compressed representation of size B, the compression ratio is $CR = B_{\mathrm{raw}}/B$. CraftBench uses FPZIP (Lindstrom & Isenburg, 2006) as the lossless reference. A generated adaptation satisfies the storage requirement only if its complete stored representation is smaller than the corresponding FPZIP representation.

B CraftBench Task Definitions

B.1 Task Catalog

CraftBench includes 83 feature-preservation tasks: 60 general-purpose features, 15 composite features that combine multiple features, and 8 domain-specific features derived from climate, cosmology, and turbulence analyses. The features cover a broad range of scientific information, including value distributions, differential features, local structures, boundaries, topological structures, spectra, distributed spatial statistics, global fidelity measures, and peak-related quantities. Each task is evaluated under three adaptation modes: error bounding (EB), configuration selection (CS), and error correction (EC). The three modes yield $83 \times 3 = 249$ instances per compressor backend (SZ3 and ZFP). We implemented the executable feature and distance functions from scientific definitions. We manually reviewed each implementation against its intended mathematical definition and verified on the paired benchmark data that it executed deterministically, returned finite outputs, and produced the expected output structure.

B.1.1 Scientific Feature and Distance Definitions

This section defines the feature functions and distance functions for the 83 CraftBench tasks. Let $f(x_i)$ and $\hat{f}(x_i)$ denote the original and decompressed values at data point x_i, for $i = 1, \ldots, N$, where N is the number of data points. The prescribed absolute pointwise error bound is denoted by e_0. Let F and $\hat{F}$ denote the feature values computed from the original and decompressed fields.

B.1.2 Numerical Fidelity and Value Distributions

The tasks in this category evaluate numerical fidelity, value distributions, and extreme-value behavior.

Pointwise numerical fidelity. The feature consists of the scalar values at all data points. The distance between the original and decompressed fields is the mean squared error:

$$D = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{f}(x_i) - f(x_i) \right)^2 .$$

Value distribution. The feature is the empirical distribution of scalar values, independent of their spatial locations. Let $f_{(i)}$ and $\hat{f}_{(i)}$ denote the original and decompressed values sorted in ascending order. The distance is the empirical 1-Wasserstein distance:

$$D = \frac{1}{N} \sum_{i=1}^{N} \left| \hat{f}(i) - f(i) \right| .$$

Value histogram. The feature is a normalized 32-bin histogram of the scalar values between the 0.5th and 99.5th percentiles. Values outside this range are omitted. The distance is the total-variation distance:

$$D = \frac{1}{2} \sum_j |p_j - \hat{p}_j|,$$

where p_j and $\hat{p}_j$ are the normalized bin frequencies of the original and decompressed fields.

Extreme values and locations. The feature consists of the ten largest scalar values and their corresponding grid locations, ordered by decreasing value. The distance combines normalized value differences and normalized spatial displacements:

$$D = \frac{1}{2} \left[\frac{1}{10} \sum_{j=1}^{10} \frac{|\hat{v}_j - v_j|}{s_v} \right] + \frac{1}{2} \left[\frac{1}{10} \sum_{j=1}^{10} \frac{\|\hat{p}_j - p_j\|_2}{s_p} \right],$$

where v_j and $\hat{v}_j$ are the jth largest original and decompressed values, p_j and $\hat{p}_j$ are their corresponding grid locations, s_v is the value normalization scale, and s_p is the diagonal length of the data domain.

Quantiles. The feature consists of the 5th, 25th, 50th, 75th, and 95th percentiles of the scalar values, providing a compact summary of the value distribution. The distance is the RMS error between the original and decompressed quantiles, normalized by the range of the original quantile values:

$$D = \frac{\sqrt{\frac{1}{5} \sum_{j=1}^{5} (\hat{q}_j - q_j)^2}}{\max_j q_j - \min_j q_j},$$

where q_j and $\hat{q}_j$ denote the corresponding quantiles of the original and decompressed fields.

Median, variability, and trimmed mean. The feature consists of the median, median absolute deviation (MAD), interquartile range (IQR), and the mean after removing the lowest and highest 10% of samples. These statistics summarize the center and variability of the value distribution while reducing the influence of extreme values.

Let m, A, Q, and t denote the median, MAD, IQR, and trimmed mean of the original field, and let $\hat{m}$, $\hat{A}$, $\hat{Q}$, and $\hat{t}$ denote the corresponding values of the decompressed field. Define

$$s = \max(A, Q, 10^{-12}).$$

The normalized component errors are

$$e_m = \frac{\hat{m} - m}{s}, \qquad e_A = \frac{\hat{A} - A}{A + s}, \qquad e_Q = \frac{\hat{Q} - Q}{Q + s}, \qquad e_t = \frac{\hat{t} - t}{s}.$$

The distance is

$$D = \sqrt{\frac{e_m^2 + e_A^2 + e_Q^2 + e_t^2}{4}},$$

where 10^{-12} is a small numerical constant used to avoid division by zero.

Skewness and kurtosis. The feature measures the asymmetry and tail behavior of the value distribution. Let

$$z_i = \frac{f(x_i) - \mu}{\sigma},$$

where μ and σ are the mean and standard deviation of the field. The feature is

$$F = \left(\frac{1}{N} \sum_{i=1}^{N} z_i^3,\ \frac{1}{N} \sum_{i=1}^{N} z_i^4 - 3 \right).$$

The first component is the skewness and the second is the excess kurtosis. A constant field returns $(0, 0)$. The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Tail frequency and severity. The feature characterizes how frequently large deviations from the mean occur and how far those deviations extend into the tails of the value distribution. It is intended to capture tail behavior that is not summarized by central quantiles or low-order moments alone.

Let $z_i = \frac{f(x_i)-\mu}{\sigma}$, where μ and σ are the mean and standard deviation of the field. For each $t \in \{1, 2, 3\}$, the tail fraction is defined as

$$r_t = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[|z_i| \geq t],$$

and the mean exceedance is defined as

$$e_t = \frac{1}{|\mathcal{I}_t|} \sum_{i \in \mathcal{I}_t} (|z_i| - t), \qquad \mathcal{I}_t = \{i : |z_i| \geq t\}.$$

The tail frequency and severity are defined as

$$F = (r_1, e_1, r_2, e_2, r_3, e_3).$$

The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Concentration of fluctuation energy. This function records the proportion of the total squared deviation (relative to the mean) accounted for by the largest 1%, 5%, 10%, and 25% of samples.

The distance is

$$D = \sqrt{\frac{1}{4} \sum_{j=1}^{4} (\hat{F}_j - F_j)^2}.$$

B.1.3 Derivatives and Spatial Statistics

The tasks in this category characterize local variation, curvature, and spatial correlation.

Gradient-magnitude statistics. The feature consists of the mean, standard deviation, and 90th percentile of the gradient magnitude, summarizing typical, variable, and local changes:

$$F = (\mu_g, \sigma_g, q_{90}(g)), \qquad g_i = \|\nabla f(x_i)\|_2.$$

The distance is

$$D = \sqrt{\frac{1}{3}\sum_{j=1}^{3}\left(\frac{\hat{F}_j - F_j}{|F_j| + 10^{-8}}\right)^2}.$$

Gradient-magnitude field. The feature consists of the gradient magnitude at every grid point and captures the spatial distribution of local variations in the field.

$$F_i = g_i = \|\nabla f(x_i)\|_2.$$

The distance is

$$D = \frac{\frac{1}{N}\sum_{i=1}^{N}|\hat{F}_i - F_i|}{\frac{1}{N}\sum_{i=1}^{N}|F_i| + 10^{-8}}.$$

The constant 10^{-8} is used to avoid division by zero.

Laplacian statistics. The feature summarizes local second-order variation using the mean, standard deviation, and a percentile of the Laplacian magnitude:

$$F = (\mathrm{mean}(L),\ \mathrm{std}(L),\ q_p(|L|)), \qquad L = \nabla^2 f,$$

where p is the prescribed percentile for the task. The distance is

$$D = \sqrt{\frac{1}{3}\sum_{j=1}^{3}\left(\frac{\hat{F}_j - F_j}{|F_j| + 10^{-8}}\right)^2}.$$

The constant 10^{-8} is used to avoid division by zero.

Laplacian field. The feature captures the spatial distribution of local second-order variation by recording the signed Laplacian at every grid point:

$$F_i = L_i = \nabla^2 f(x_i).$$

The distance is

$$D = \frac{\frac{1}{N}\sum_{i=1}^{N}|\hat{F}_i - F_i|}{\frac{1}{N}\sum_{i=1}^{N}|F_i| + 10^{-8}}.$$

The constant 10^{-8} is used to avoid division by zero.

Hessian-eigenvalue statistics. Let

$$h_i = \max_k |\lambda_k(H_i)|,$$

where $\lambda_k(H_i)$ are the eigenvalues of the Hessian at grid point x_i. The feature describes the strongest local second-order variation using

$$F = (\mathrm{mean}(h),\ \mathrm{std}(h),\ q_{90}(h))\,.$$

The distance is

$$D = \sqrt{\frac{1}{3}\sum_{j=1}^{3}\left(\frac{\hat{F}_j - F_j}{|F_j| + 10^{-8}}\right)^2}\,.$$

Blockwise variance. The feature describes the spatial distribution of local variability by partitioning the data into non-overlapping blocks and computing the variance within each block. For block k,

$$F_k = \mathrm{Var}(f_k).$$

The distance is

$$D = \frac{\mathrm{mean}\left(|\hat{F} - F|\right)}{\mathrm{mean}\left(|F|\right) + 10^{-8}}.$$

The constant 10^{-8} is used to avoid division by zero.

Second-order structure function. The feature measures the intensity of fluctuations at fixed spatial intervals. For a lag r, it calculates the average of the squared differences in the horizontal and vertical directions:

$$F = \frac{1}{2}\left[\mathrm{mean}\left((f_{i,j+r} - f_{i,j})^2\right) + \mathrm{mean}\left((f_{i+r,j} - f_{i,j})^2\right)\right].$$

The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 10^{-8}}.$$

The constant 10^{-8} is used to avoid division by zero.

Nearest-neighbor spatial autocorrelation. The feature measures the spatial correlation between neighbouring values. Let E contain each horizontally or vertically adjacent pair exactly once, and let $u_i = f(x_i) - \mu$, where μ is the mean of the field. The feature is defined as

$$F = \frac{N}{|E|}\frac{\sum_{(i,j)\in E} u_i u_j}{\sum_i u_i^2 + 10^{-30}}.$$

The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 10^{-8}}.$$

Gradient-orientation tensor. This feature captures the directional organization of local gradients without considering their magnitudes. For each grid point, we define

$$u_i = \frac{\nabla f(x_i)}{\|\nabla f(x_i)\|_2},$$

with $u_i = 0$ when the gradient magnitude is zero. The feature is defined as

$$F = \frac{1}{N}\sum_{i=1}^{N} u_i u_i^\top.$$

The distance is

$$D = \sqrt{\text{mean}\left[(\hat{F} - F)^2\right]}.$$

Hessian sign profile. This feature characterises the type of local curvature by recording the proportion of grid points with $0, \ldots, d$ positive eigenvalues of the Hessian matrix:

$$F = (p_0, p_1, \ldots, p_d),$$

where p_k is the proportion of points with exactly k positive eigenvalues of the Hessian matrix. An eigenvalue is counted as positive only if it exceeds $10^{-6}\max\left(\max_{i,k}|\lambda_k(H_i)|, 1\right)$.

The distance is

$$D = \max_j |\hat{F}_j - F_j|.$$

Spatial autocorrelation profile. The feature measures how spatial correlation varies with distance. For each lag value r, it calculates the normalised autocorrelation coefficients for valid point pairs along each coordinate direction, and then averages the results across all directions:

$$C_r = \frac{\text{mean}\left[(f(x_i) - \mu)(f(x_i + re_k) - \mu)\right]}{\sigma^2 + 10^{-30}}.$$

For $r \in \{1, 2, 4, 8\}$, the feature is

$$F = (C_1, C_2, C_4, C_8).$$

The distance is

$$D = \sqrt{\text{mean}\left[(\hat{F} - F)^2\right]}.$$

Semivariogram profile. This feature measures how spatial variability changes with distance. For each lag value $r \in \{1, 2, 4, 8\}$, it calculates the normalised semivariogram

$$F_r = \frac{\frac{1}{2}\,\text{mean}\left[\left(f(x_i + re_k) - f(x_i)\right)^2\right]}{\sigma^2 + 10^{-30}},$$

and performs an equally weighted average across all valid coordinate directions. The feature is defined as $F = (F_1, F_2, F_4, F_8)$.

The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Directional semivariogram profile. This feature measures spatial variability along each coordinate direction separately. For each axis k and lag $r \in \{1, 2, 4, 8\}$,

$$F_{k,r} = \frac{\frac{1}{2}\,\mathrm{mean}\left[(f(x_i + re_k) - f(x_i))^2\right]}{\mathrm{Var}(f) + 10^{-30}}.$$

The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Local-extrema density. The feature measures the prevalence of strong local maxima and minima. The feature is defined as

$$F = \left(\frac{N_{\max}}{N}, \frac{N_{\min}}{N}\right).$$

The distance is

$$D = \max_j |\hat{F}_j - F_j|.$$

Multiscale gradient energy. The feature measures gradient strength across multiple smoothing scales. For each $\sigma \in \{0, 1, 2, 4\}$,

$$E_\sigma = \frac{\sum_k \mathrm{mean}\left[(\partial_k (G_\sigma * f))^2\right]}{\mathrm{Var}(f) + 10^{-30}},$$

where $G_0 * f = f$. The feature is the vector of E_σ values across scales. The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Gradient coherence. The feature measures the degree of alignment of the gradient energy along the preferred direction. We define

$$C = \frac{1}{N} \sum_{i=1}^{N} \nabla f(x_i) \nabla f(x_i)^\top.$$

Let $\lambda_{\max}$ and $\lambda_{\min}$ denote the largest and smallest eigenvalues of C. The feature is defined as

$$F = \frac{\lambda_{\max} - \lambda_{\min}}{\sum_j \lambda_j + 10^{-30}}.$$

The distance is

$$D = |\hat{F} - F|.$$

Directional gradient-energy fractions. The feature measures the distribution of gradient energy across the various coordinate directions. For each axis k,

$$E_k = \text{mean}\left[(\partial_k f)^2\right], \qquad p_k = \frac{E_k}{\sum_j E_j}.$$

The feature is defined as $F = (p_1, \ldots, p_d)$. The distance is

$$D = \frac{1}{2} \sum_k |\hat{F}_k - F_k|.$$

Hessian-determinant statistics. The feature uses the determinant of the Hessian matrix, normalised by the field standard deviation, to characterise the local second-order structure. At each grid point,

$$h_i = \det\left(\frac{H_i}{\sigma + 10^{-30}}\right).$$

The feature is defined as

$$F = \left(\text{mean}(h),\ \text{std}(h),\ q_{95}(|h|)\right).$$

The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Local-mean field. This function captures the spatial distribution of local averages by applying a seven-unit-wide average filter and using reflective boundary conditions:

$$F = G_7 * f.$$

The distance is

$$D = \frac{\sqrt{\text{mean}\left[(\hat{F} - F)^2\right]}}{s},$$

where $s = \begin{cases} \max(F) - \min(F), & \text{if the range is not near zero,} \\ \max\left(\text{RMS}(F), 1\right), & \text{otherwise.} \end{cases}$

Local-entropy field. This feature reflects the spatial distribution of the complexity of local value distributions. The field is mapped to 16 intervals using the global 0.5th and 99.5th percentiles, and the normalised Shannon entropy is calculated within each block of full width 8:

$$H = -\sum_{b=1}^{16} p_b \log p_b, \qquad F = \frac{H}{\log 16}.$$

The distance is

$$D = \sqrt{\text{mean}\left[(\hat{F} - F)^2\right]}.$$

Higher-order structure-function profile. The feature measures the amplitude of higher-order increments across multiple spatial intervals. For order $p \in \{3, 4\}$ and lag $r \in \{1, 2, 4, 8\}$,

$$F_{p,r} = \text{mean}\left[\left|\frac{f(x_i + re_k) - f(x_i)}{\sigma + 10^{-30}}\right|^p\right],$$

where increments from all valid coordinate directions are combined.

The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Local autocorrelation map. The feature captures the spatial distribution of short-range correlations by dividing the field into non-overlapping blocks and calculating the first-order lagged autocorrelation within each block. For block B,

$$F_B = \text{mean}_k\left[\frac{\text{mean}\left[(f_i - \mu_B)(f_{i+e_k} - \mu_B)\right]}{\text{Var}(f_B) + 10^{-30}}\right],$$

where the mean is calculated with respect to the effective coordinate directions. The field is divided into four sectors along each coordinate axis. The distance is

$$D = \sqrt{\text{mean}\left[(\hat{F} - F)^2\right]}.$$

Multiscale Laplacian energy. The feature measures second-order variation across multiple smoothing scales. For $\sigma \in \{0, 1, 2, 4\}$,

$$F_\sigma = \frac{\text{mean}\left[\left(\nabla^2(G_\sigma * f)\right)^2\right]}{\text{Var}(f) + 10^{-30}},$$

where $G_0 * f = f$. The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Hessian-curvature occupancy. This feature measures the proportion of grid points exhibiting strongly negative or strongly positive Hessian curvature. Prior to calculating the Hessian eigenvalues, this field is standardised and smoothed using a Gaussian kernel. Let $\tau = 0.25\sqrt{\text{mean}_{i,k}\left[\lambda_k(H_i)^2\right]}$. The feature is defined as

$$F = \left(\frac{1}{N}\sum_i \mathbf{1}\left[\lambda_{\min}(H_i) < -\tau\right],\ \frac{1}{N}\sum_i \mathbf{1}\left[\lambda_{\max}(H_i) > \tau\right]\right).$$

The distance is

$$D = \max_j |\hat{F}_j - F_j|.$$

B.1.4 GEOMETRY AND TOPOLOGY

The tasks in this category are used to characterise threshold interfaces, connected regions, component geometries and topological structures based on persistence. Connected components are defined using grid points that are axially adjacent. The geometric quantities used here are approximations based on a regular grid.

Threshold-interface length proxy. The feature approximates the range of the median threshold boundary. We define $M_i = \mathbf{1}[f(x_i) \geq q_{50}(f)]$. The feature counts the number of horizontally and vertically adjacent pairs in M that have different membership levels. Each such pair of adjacent points indicates that the threshold boundary lies between these two grid points. The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 10^{-8}}.$$

The constant 10^{-8} is used to avoid division by zero.

Top-16 topological persistence lifetimes. This feature retains up to the sixteen longest super-level-set persistence lifetimes; let $F = (\ell_1, \ldots, \ell_m), \qquad \hat{F} = (\hat{\ell}_1, \ldots, \hat{\ell}_{\hat{m}})$ denote the sorted persistence lifetimes of the original and decompressed fields, with shorter vectors padded with zeros to the same length. The distance is

$$D = \frac{\sum_j |\hat{\ell}_j - \ell_j|}{\sum_j \ell_j + 10^{-8}}.$$

Birth and death coordinates are not retained, so this compares dominant persistence lifetimes rather than full persistence diagrams.

Large excursion–set component count. This feature counts connected regions that are sufficiently large and exceed a certain high threshold. We define $T = \mu + k_\sigma$. The feature is defined as

$$F = \#\left\{C : C \subset \{x_i : f(x_i) > T\},\ |C| \geq s_{\min}\right\}.$$

The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 1}.$$

Euler characteristic. This feature summarises the topological structure of regions following thresholding by comparing the number of connected components with the number of enclosed holes. We define

$$M = \{x_i : f(x_i) > q_p(f)\},$$

where $q_p(f)$ is the prescribed percentile threshold. Let $C(M)$ denote the number of connected components in M. The feature is defined as

$$F = C(M) - \max\left(C(\Omega \setminus M) - 1, 0\right).$$

The feature measures the balance between connected regions and holes in a thresholded field. The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 1}.$$

Persistent–extrema count. The feature counts extrema whose persistence is larger than a fixed fraction of the field value range. We define the persistence threshold as

$$\tau = \tau_{\text{frac}} \left(\max_i f(x_i) - \min_i f(x_i) \right).$$

If pers_j denotes the persistence of pair j, then the feature is defined as

$$F = \#\left\{ j : \text{pers}_j > \tau \right\}.$$

The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 1}.$$

Topological merge level. The feature records the scalar value at which a dominant topological component merges with an older component. It is the death value of the pair of components ranked second in terms of persistence; if only one pair of components exists, it is the 'death value' of that pair. The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 10^{-8}}.$$

Excursion–set component size distribution. The feature describes the distribution of connected component sizes greater than the threshold $\mu + 2\sigma$, where μ and σ are the mean and standard deviation of the field. Each connected component C_j is represented by its size proportion

$$s_j = \frac{|C_j|}{N}.$$

The feature is the normalized histogram of s_j in eight logarithmically spaced bins between $1/N$ and 1. The distance is

$$D = \frac{1}{2} \sum_j |\hat{F}_j - F_j|.$$

Excursion–set component position distribution. The feature describes the spatial distribution of connected components of the excursion set above the threshold

$$T = \mu + 2\sigma,$$

where μ and σ are the mean and standard deviation of the field. The centroid of mass of each component is represented by its distance from the centroid of the domain, normalised relative to the distance from the center to the vertices. The distance is

$$D = \frac{1}{2} \sum_j |\hat{F}_j - F_j|.$$

Interface-crossing density. This feature is used to measure the frequency with which threshold boundaries cross between adjacent grid points. We define

$$T = \mu + 0.5\sigma,$$

where μ and σ are the mean and standard deviation of the field. The feature represents the proportion of adjacent grid point pairs along the axis where one value is greater than T, and the other is less than T. The distance is

$$D = |\hat{F} - F|.$$

Boundary-distance profile. This feature measures the distance between grid points and the boundaries of the threshold regions. We define $M = \{x_i : f(x_i) \geq \mu + 0.5\sigma\}$, where μ and σ are the mean and standard deviation of the field values. The inner boundary of M is obtained by removing the result of binary erosion applied to it.

The distance from each grid point to this boundary is normalized using the grid diagonal. This feature is defined as

$$F = (q_{25}, q_{50}, q_{75}, q_{95}),$$

where these four values are the 25th, 50th, 75th and 95th percentiles of the normalized boundary distances, respectively. The distance is given by

$$D = \sqrt{\frac{1}{4}\sum_{j=1}^{4}(\hat{F}_j - F_j)^2}.$$

Excursion–set component profile. The feature measures how the number of connected regions changes with the threshold. For

$$t \in \{-1, 0, 1, 2\},$$

we define the excursion set

$$M_t = \{x_i : f(x_i) \geq \mu + t\sigma\},$$

where μ and σ are the mean and standard deviation of the field. The feature records

$$F_t = \frac{C(M_t)}{N},$$

where $C(M_t)$ is the number of connected components in M_t. The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Persistent extrema locations. The feature records the locations and normalized persistence lifetimes of up to five most persistent extrema. The distance uses an optimal assignment between extrema in the original and decompressed fields. The cost of a matched pair is

$$c = 0.75\, d_{\text{pos}} + 0.25\, d_{\text{pers}},$$

where d_{pos} is the normalized positional distance and d_{pers} is the absolute difference in normalized persistence. We use the minimum total assignment cost as the final distance, with each unmatched extremum assigned a cost of 1, normalized by the larger number of retained extrema.

Persistence entropy. The feature measures how persistence is distributed among topological features whose lifetimes exceed $0.01R$, where $R = \max_i f(x_i) - \min_i f(x_i)$. If the retained lifetimes are $\ell_1, \ldots, \ell_m$, define

$$p_j = \frac{\ell_j}{\sum_{k=1}^{m} \ell_k}.$$

The feature is

$$F = -\frac{\sum_{j=1}^{m} p_j \log p_j}{\log m},$$

with $F = 0$ when $m \leq 1$. The distance is

$$D = |\hat{F} - F|.$$

Component compactness distribution. The feature measures the geometric compactness of connected components above a high threshold. Let $M = \{x_i : f(x_i) > \mu + 2\sigma\}$. For each connected component C_j of M, we define the grid-based compactness metrics as:

$$c_j = \frac{|C_j|^{(d-1)/d}}{|\partial C_j| + 10^{-30}},$$

where $|\partial C_j|$ is the number of boundary grid points of the component. The feature consists of the 25th, 50th, 75th, and 90th percentiles of the component compactness values. If no component is present, the feature is zero. The distance is

$$D = \sqrt{\text{mean}\left[(\hat{F} - F)^2\right]}.$$

Topological persistence–size relation. The feature measures the relationship between the size of a topological component and its persistence. For each persistence pair, the persistence lifetime is normalized by the field value range, and the size of the component is divided by N.

The normalized persistence range is divided into the intervals

$$[0, 0.01), \quad [0.01, 0.05), \quad [0.05, 0.1), \quad [0.1, 1.01).$$

For each interval, the feature records the mean normalized size of the persistence pairs whose lifetimes fall in that interval. The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Boundary curvature statistics. The feature measures the curvature distribution along the boundaries of thresholded regions. We define the excursion set as $M = \{x_i : z_i \geq 0.5\}$. A signed distance field is then computed from the boundary of M, and its normalized gradient is used to estimate boundary curvature. The feature consists of the 25th, 50th, 75th, and 95th percentiles of the absolute boundary curvature, normalised along the grid diagonals. The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Multi-threshold excursion-set overlap. The feature measures the spatial overlap between excursion sets of the original and decompressed fields at multiple standardized thresholds. For each $t \in \{0, 1, 2\}$, the feature is the binary excursion-set mask

$$F_t = \mathbf{1}[z_i > t].$$

The distance measures how much the original and decompressed excursion sets fail to overlap across the three thresholds:

$$D = \frac{1}{3} \sum_{t \in \{0,1,2\}} \left(1 - \frac{|\hat{M}_t \cap M_t|}{|\hat{M}_t \cup M_t|} \right).$$

B.1.5 Spectral and Multiscale Features

The tasks in this category describe the distribution of field variations in terms of frequency, direction and spatial scale. We define$X_k = \mathcal{F}(f - \mu)_k, \qquad P_k = |X_k|^2, \qquad p_k = \frac{P_k}{\sum_j P_j}$, where μ is the mean of the field. For frequency f_k in cycles per grid spacing, the normalized radial frequency is $r_k = \frac{2\|f_k\|_2}{\sqrt{d}}$, where d is the dimensionality of the field. A zero-energy field returns spectral features with a value of zero.

Spectral-band energy fractions. The feature measures the distribution of spectral energy across four radial frequency bands. Using the normalized spectral energy p_k, define $F_j = \sum_{k:r_k \in B_j} p_k$, where the band boundaries are$0,\ 0.1,\ 0.25,\ 0.5,\ 1.01$. The distance is

$$D = \frac{1}{2} \sum_j |\hat{F}_j - F_j|.$$

Spectral entropy. The feature measures the range of spectral energy distribution across different frequencies:

$$F = -\frac{\sum_{k:p_k>0} p_k \log p_k}{\log N}.$$

The lower the value, the more the spectral energy is concentrated at a smaller number of frequencies; the higher the value, the broader the spectral distribution. The distance is

$$D = |\hat{F} - F|.$$

Spectral centroid and bandwidth. The feature describes the center and spread of spectral energy over radial frequency. The spectral centroid is $c = \sum_k r_k p_k$, and the spectral bandwidth is $w = \left[\sum_k (r_k - c)^2 p_k\right]^{1/2}$. The feature is defined as

$$F = (c, w).$$

The distance is

$$D = \sqrt{\frac{1}{2} \sum_{j=1}^{2} (\hat{F}_j - F_j)^2}.$$

Multiscale residual energy. The feature measures the amount of residual variation remaining after Gaussian smoothing has been applied at multiple scales. For each smoothing scale $s \in \{1, 2, 4, 8\}$,

$$F_s = \frac{\text{mean}\left[(f - G_s * f)^2\right]}{\text{Var}(f) + 10^{-30}}.$$

The feature is the vector of residual energies across the four scales. The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

Radial spectral-energy profile. The feature measures the distribution of spectral energy across the radial frequency. The normalised radial frequency range $[0, 1]$ is divided into 16 equally spaced bins, and this feature records the proportion of total spectral energy accounted for by each bin:

$$F_j = \sum_{k: r_k \in B_j} p_k, \qquad j = 1, \ldots, 16.$$

The feature represents the proportion of energy in each radial frequency band, rather than the average power within each bin. The distance is

$$D = \frac{1}{2} \sum_{j=1}^{16} |\hat{F}_j - F_j|.$$

Spectral anisotropy tensor. The feature measures the distribution of spectral energy across different directions in the frequency domain and is defined as

$$F = \frac{\sum_k P_k \, f_k f_k^\top}{\sum_k P_k \, \|f_k\|_2^2}.$$

The tensor is flattened into a feature vector. The distance is

$$D = \sqrt{\text{mean}\left[(\hat{F} - F)^2\right]}.$$

Coarse-scale variance profile. The feature measures how large-scale variation changes with spatial scale. For each block width $s \in \{2, 4, 8, 16\}$, the field is partitioned into non-overlapping blocks, and the mean value of each block is computed. The feature at scale s is

$$F_s = \frac{\text{Var}\left(\{\mu_{B_j}^{(s)}\}_j\right)}{\text{Var}(f) + 10^{-30}},$$

where $\mu_{B_j}^{(s)}$ is the mean of block B_j at scale s. The distance is

$$D = \sqrt{\frac{1}{4} \sum_{j=1}^{4} (\hat{F}_j - F_j)^2}.$$

Haar-style scale-energy fractions. The feature measures the distribution of field variations across four scales, ranging from coarse to fine. At each scale, a differential operation is performed on pairs of neighbouring points along each coordinate direction, and the result is divided by $\sqrt{2}$; then the mean square detail energy is calculated across all directions.

Let E_ℓ denote the detail energy at level ℓ. The feature is the normalized energy vector

$$F_\ell = \frac{E_\ell}{\sum_{j=1}^{4} E_j + 10^{-30}}, \qquad \ell = 1, \ldots, 4.$$

The distance is

$$D = \frac{1}{2} \sum_{\ell=1}^{4} |\hat{F}_\ell - F_\ell|.$$

Spectral phase coherence. This feature is used to measure the consistency of the phase difference between adjacent frequencies along each frequency grid axis. For axis j, it is defined as

$$F_j = \frac{\left| \sum_k |X_k| \, |X_{k+e_j}| \, \exp\left[\mathrm{i} \left(\arg X_k - \arg X_{k+e_j}\right)\right] \right|}{\sum_k |X_k| \, |X_{k+e_j}|}.$$

The distance is

$$D = \sqrt{\frac{1}{d} \sum_{j=1}^{d} (\hat{F}_j - F_j)^2}.$$

Local spectral-energy map. The feature measures the spatial and frequency distribution of spectral energy. Each spatial axis is divided into four blocks, thereby forming 4^d non-overlapping local regions. Within each block, the mean is removed, and a Hann window is applied before computing the Fourier power spectrum to reduce spectral leakage (Donnelly & Rust, 2005).

The local spectral energy is summed into three normalized radial frequency bands with boundaries $0,\ 0.2,\ 0.45,\ 1.01$. For each block b and frequency band j, the feature is the fraction of the block's spectral energy in that band:

$$F_{b,j} = \frac{\sum_{k : r_k \in B_j} P_{b,k}}{\sum_k P_{b,k}}.$$

The distance is

$$D = \frac{\|\hat{F} - F\|_2}{1 + \|F\|_2}.$$

B.1.6 Composite Features

The tasks in this category combine multiple features to evaluate preservation of several field properties simultaneously. The first three tasks use task-specific distance functions. For the remaining 12 tasks, the feature distance is computed separately for every single feature. Let F_j and $\hat{F}_j$ denote the jth feature computed from the original and decompressed fields. The composite distance is the largest normalized feature error:

$$D = \max_j \frac{\sqrt{\mathrm{mean}\left[(\hat{F}_j - F_j)^2\right]}}{\max\left(\sqrt{\mathrm{mean}(F_j^2)}, 1\right)}.$$

Numerical fidelity and gradients. The feature combines the field values with the gradient-magnitude map. Let

$$E_f = \frac{\sqrt{\text{mean}\left[(\hat{f} - f)^2\right]}}{\max f - \min f},$$

and

$$E_g = \frac{\text{mean}\,|\hat{G} - G|}{\text{mean}\,|G| + 10^{-8}},$$

where G is the gradient-magnitude map. The distance is

$$D = \max\left(\frac{E_f}{0.01}, \frac{E_g}{0.10}\right).$$

Distribution and connectivity. The feature combines an adaptive 32-bin histogram with the number of connected components containing at least 16 points above $\mu + 2\sigma$. Let

$$E_h = \frac{1}{2}\sum_j |\hat{H}_j - H_j|,$$

and

$$E_c = \frac{|\hat{C} - C|}{|C| + 1}.$$

The distance is

$$D = \max\left(\frac{E_h}{0.05}, \frac{E_c}{0.10}\right).$$

Extreme values and spectral slope. This feature combines the ten largest field values and their positions with the log–log slope of radially averaged spectral power over normalized radial frequencies from 0.05 to 0.30. Let

$$E_{\text{val}} = \frac{\text{mean}\,|\hat{V} - V|}{\max|V| + 10^{-8}},$$

and

$$E_{\text{pos}} = \frac{\text{mean}_i\,\|\hat{p}_i - p_i\|_2}{\|\text{shape}\|_2}.$$

The extrema error is

$$E_{\text{ext}} = \frac{1}{2}E_{\text{val}} + \frac{1}{2}E_{\text{pos}},$$

and the spectral-slope error is

$$E_{\text{slope}} = \frac{|\hat{s} - s|}{|s| + 10^{-8}}.$$

The distance is

$$D = \max\left(\frac{E_{\text{ext}}}{0.10}, \frac{E_{\text{slope}}}{0.10}\right).$$

Standardized quantiles and gradients. The feature combines the 5th, 25th, 50th, 75th, and 95th percentiles of the standardized field z with the mean, standard deviation, and 90th percentile of $\|\nabla z\|_2$.

Tail frequency and connectivity. The feature combines the fractions of points with $z_i > 1$, $z_i > 2$, and $z_i > 3$ with three measurements of the set $\{z_i > 1\}$: the number of connected components divided by N, the largest-component size divided by N, and the occupied fraction.

Interface density and boundary gradients. For the thresholded set $\{z_i > 0.5\}$, the feature combines the average boundary crossing rate and the average boundary gradient magnitude in each axis direction, divided by the global average gradient magnitude, plus 10^{-30}. Boundary points use periodic neighbor comparisons, while crossing fractions use nonperiodic adjacent pairs.

Component sizes and spatial arrangement. For connected components of $\{z_i > 1\}$, the feature combines an eight-bin histogram of

$$\frac{\log(1 + |C_j|)}{\log(1 + N)}$$

with the axis-wise mean and standard deviation of component centroids in normalized grid coordinates and the three connectivity measurements defined above.

Quantiles and spectral energy. The feature combines the standardized-field quantiles at $p = 5, 25, 50, 75, 95$ with the four spectral-band energy fractions.

Local and global standardized variance. The feature combines a width-seven moving-window variance map of the standardized field z, using reflected boundaries, with the global variance $\mathrm{Var}(z)$. For nonconstant fields, the global variance is approximately one.

Persistence and extrema arrangement. The feature combines persistence statistics with the spatial arrangement of extrema. Small persistence pairs are removed using the threshold $10^{-12}\max(R, 1)$, where R is the field value range. The feature includes the 25th, 50th, 75th, and maximum normalized persistence lifetimes. The feature also measures the average location and spatial spread of the retained extrema, with more persistent extrema given larger weights. It also includes the number of retained persistence pairs divided by N.

Multiscale residuals and connectivity. For $s = 1, 2, 4, 8$, we define $E_s = \text{mean}\left[(z - G_s * z)^2\right]$. The feature combines $E_s/(1 + E_s)$ with component counts divided by N for the thresholded sets $\{z_i > t\}$ at $t = -1, 0, 1, 2$.

Distribution and spatial autocorrelation. The feature combines a normalized 32-bin histogram of z over $[-4, 4]$ with nearest-neighbor Moran's I over all available grid axes. Values outside $[-4, 4]$ are omitted from the histogram.

Curvature and interface geometry. The feature combines the fractions of points having $0, \ldots, d$ Hessian eigenvalues of z greater than 10^{-8} with the interface-density and relative boundary-gradient measurements defined above.

Distributional moments and tails. The feature combines the standardized third and fourth moments,

$$\big(\mathrm{mean}(z^3), \mathrm{mean}(z^4) - 3\big)\,,$$

with the fractions of points satisfying $|z_i| > 1$, $|z_i| > 2$, and $|z_i| > 3$.

Spectral energy and spatial increments. The feature combines the four spectral-band energy fractions with half the mean squared standardized increments at periodic lags $1, 2, 4, 8$, averaged equally over coordinate directions.

B.1.7 Domain-Specific Features

The tasks in this category use measurements defined for specific scientific variables. They include turbulence features based on enstrophy and intermittency (Elsinga et al., 2023), climate features based on zonal-mean air temperature (Kalnay et al., 1996), and cosmology features based on density contrast and power spectra (Weinberg et al., 2013).

Enstrophy distribution. The feature describes the distribution of enstrophy values. Enstrophy is commonly used to characterize small-scale rotational activity and intense events in turbulence (Elsinga et al., 2023). For nonnegative enstrophy

$$e_i = \max(f(x_i), 0),$$

the values $\log_{10}(e_i + 10^{-30})$ are summarized by a normalized histogram over the 0.5th–99.5th percentile range. The distance is

$$D = \frac{1}{2} \sum_j |\hat{F}_j - F_j|.$$

Intense-enstrophy occupancy. The feature measures the spatial distribution density of samples with high enstrophy values. The field is divided into discrete spatial blocks, and for each block, the number of samples whose enstrophy exceeds a prescribed threshold. The distance is

$$D = \frac{\mathrm{mean}\,|\hat{F} - F|}{\mathrm{mean}\,|F| + 10^{-8}}.$$

Zonal-mean temperature profile. The feature measures the variation in zonal-mean air temperature with latitude. The zonal mean is calculated by averaging the values in the longitude direction at each latitude (Kalnay et al., 1996). First, the air temperature is averaged in the longitude direction, and then averaged across consecutive latitude bands. The distance is

$$D = \sqrt{\mathrm{mean}_j \left[\left(\frac{\hat{F}_j - F_j}{|F_j| + 10^{-8}} \right)^2 \right]}.$$

Density-spectrum slope. The feature measures the slope of the density power spectrum over a prescribed radial-frequency range. In cosmology, the density contrast and its power spectrum are metrics for describing matter fluctuations across different spatial scales (Weinberg et al., 2013). The density contrast is defined as

$$\delta = \frac{f}{\mu + 10^{-30}} - 1.$$

Fourier power is averaged within radial frequency bins, and a linear fit is applied to log power versus log radial frequency. The feature is the fitted slope F. The distance is

$$D = \frac{|\hat{F} - F|}{|F| + 10^{-8}}.$$

Mass distribution across density levels. The feature measures the distribution of total mass across different density levels. Non-negative densities are normalised by their mean to form an overdensity:

$$\rho_i = \frac{\max(f(x_i), 0)}{\mathrm{mean}(\max(f, 0)) + 10^{-30}}.$$

Density values are divided into log-interval overdensity bins, with each feature entry recording the proportion of total mass within a single bin. The distance is

$$D = \frac{1}{2} \sum_j |\hat{F}_j - F_j|.$$

High-temperature exceedance fraction. This feature measures the fraction of air-temperature samples above a prescribed absolute temperature threshold T:

$$F = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[f(x_i) > T].$$

Each grid point has an equal weight; therefore, this is a proportional sample rather than a geographical area weighted by latitude and area. The distance is

$$D = |\hat{F} - F|.$$

Enstrophy intermittency. The feature describes the intensity of anomalously high enstrophy and its spatial concentration. Strong and spatially localised enstrophy is a typical manifestation of small-scale turbulence intermittency (Elsinga et al., 2023). Let $M = \{e_i > \mu_e + k\sigma_e\}$, where k is the prescribed threshold parameter. The feature contains

$$\frac{\mathrm{mean}(e^2)}{\mu_e^2 + 10^{-30}}, \qquad \frac{\mathrm{mean}_{i \in M}(e_i)}{\mu_e + 10^{-30}}, \qquad \frac{|M|}{N}, \qquad \frac{C(M)}{N},$$

where $C(M)$ is the number of connected high-enstrophy regions. These quantities measure intermittency, the relative strength of intense regions, their occupied fraction, and their component density. The distance is

$$D = \sqrt{\frac{1}{4} \sum_{j=1}^{4} (\hat{F}'_j - F'_j)^2},$$

where $F' = (\log(1 + F_1), \log(1 + F_2), F_3, F_4)$.

Coarse temperature-anomaly field. The feature measures large-scale longitudinal temperature anomalies. At each latitude, the zonal mean is first subtracted from the temperature field. The resulting anomaly field is then averaged within spatial blocks, including partial blocks at the field boundaries. The distance is

$$D = \frac{\sqrt{\mathrm{mean}\left[(\hat{F} - F)^2\right]}}{s},$$

where

$$s = \begin{cases} \max(F) - \min(F), & \text{if } \max(F) - \min(F) > 10^{-30}, \\ \max\left(\sqrt{\text{mean}(F^2)}, 1\right), & \text{otherwise.} \end{cases}$$

C Datasets and Benchmark Composition

CraftBench does not evaluate each task on the full source dataset. Instead, we construct fixed evaluation fields from the source data using predefined slices or crops. This controlled field construction enables systematic evaluation across the full set of feature tasks, compression backends, adaptation modes, and models, while ensuring reproducible comparisons on identical inputs. Each benchmark task is evaluated on one of these fixed fields. Because the 2D evaluation field is smaller than the 3D fields, we additionally examine a larger 2D field in Appendix F.9.

We use scientific scalar data from three application domains: turbulence, climate, and cosmology. The evaluation fields are derived from TACC turbulence enstrophy from the OME-Zarr Open SciVis collection (McCormick, 2026), NCEP/NCAR air temperature (Kalnay et al., 1996), and Nyx density from SDRBench (Zhao et al., 2020a). The benchmark contains 74 2-D feature tasks and 9 3-D feature tasks, each evaluated on a field matching the dimensionality required by its feature definition. Table 4 reports the corresponding data sources, source dimensions, and evaluation-field dimensions.

All evaluation fields are represented as float32 arrays. We preprocess the data as needed, including cropping, slicing, axis reordering, or precision conversion to construct the fixed evaluation fields. General-purpose tasks can be applied to any compatible scalar field, whereas domain-specific tasks are applied only to variables with the required scientific meaning and units. Multiple tasks may share the same field while evaluating different scientific features.

Table 4: Scientific data and fixed evaluation fields used in CraftBench. Source shape refers to the source array from which each evaluation field is constructed.

Domain	Source	Variable	Source Shape	Eval. Shape	Tasks
Climate	NCEP/NCAR	Air temperature (K)	$366\times73\times144$	73×144	74
Cosmology	Nyx	Density	512^3	128^3	6
Turbulence	TACC	Enstrophy	256^3	128^3	3
Total					83

D Evaluation Protocol

This section describes how the model-generated adaptations are executed, validated, and scored in CraftBench, including the three adaptation modes, the base reconstruction and success criteria, the handling of invalid outputs, and how the final evaluation metrics are computed.

D.1 Adaptation Modes and Execution

In the EB mode, the generated Python function takes the original field and the prescribed error bound e_0 as input and returns a pointwise error-bound map with the same shape as the field, where each entry satisfies $0 \le e_i \le e_0$. SZ3 applies these bounds pointwise, whereas ZFP assigns each native block the minimum bound among all data points in the same block.

For storage calculation in SZ3 EB mode, the pointwise bounds are represented by a codebook of at most 16 IEEE-754 `float64` values. When more than 16 levels are needed, CraftBench selects evenly spaced order statistics of the requested bounds. Each point is assigned the largest codebook value not exceeding its requested bound, so quantization never increases the requested error bound. Each data point stores the index of its assigned level using $\lceil \log_2 L \rceil$ bits, packed contiguously in row-major order, where L is the number of distinct levels; a constant error map requires no assignment bits. The codebook and packed assignments are compressed with Zstd together with the SZ3 compressed data. All associated metadata and storage overhead are included in B_c.

For ZFP EB mode, each block first takes the minimum pointwise bound and then applies the same quantization rule to the full sequence of block bounds. Blocks assigned the same level are compressed in batches. The stored representation includes the codebook, one `uint8` level code per block, the compressed batch streams, and length metadata; all associated overhead is included in B_c.

In the CS mode, the generated adaptation selects one of the supported compressor configurations described in Appendix A. Every resulting reconstruction is evaluated against the prescribed pointwise error bound e_0; a configuration that violates e_0 fails the pointwise-error criterion.

In the EC mode, the generated adaptation compares the original field with the base reconstruction and produces correction data. During decompression, the stored corrections are applied to the base reconstruction. The corrected field must satisfy the prescribed pointwise error bound. The correction data are stored using the smaller of the raw and losslessly Zstd-compressed representations, and all associated storage overhead is included in B_c.

D.2 Prompt Templates

This section summarizes the task specification and required output format used in the prompts. The full prompts additionally contain compressor-specific instructions, runtime configurations, and execution constraints.

D.2.1 Shared Task Specification

All three adaptation modes receive the same dataset metadata, executable feature and distance functions, pointwise error bound, base feature distance d_0, target distance $0.5d_0$, storage requirement, and uniform-reference criterion. The prompt states the dominance rule but does not provide the measured reference values.

````
You are solving one instance of a feature-preserving scientific data
compression benchmark.

DATA SOURCE:
<dataset, source, variable, units, preprocessing, shape, and data type>

TASK CODE:
```python
<EXECUTABLE_FEATURE_CODE>
```

SCORING (lower is better):
```python
<EXECUTABLE_DISTANCE_CODE>
```

POINTWISE ERROR BOUND:
For the final decompressed data, |reconstructed[i] - original[i]| <= e0
at every data point, where e0 = <E0>.

FEATURE-PRESERVATION TARGET:
The baseline feature distance is <BASELINE_DISTANCE>. The generated solution
must reduce it by at least 50%, to at most <TARGET_DISTANCE>.

OFFICIAL SUCCESS REQUIRES ALL FOUR CONDITIONS:
1. feature_distance <= 0.5 * frozen_d0.
2. max_abs_error <= e0.
3. The candidate is not dominated by the fixed 25-point observed uniform-EB
   sweep under the measured feature-distance / total-stored-bytes comparison
   defined below.
4. total_candidate_bytes < lossless_reference_bytes.

STORAGE AND EFFECTIVENESS DETAILS:
The candidate is compared with measured reference reconstructions produced
for the same task and backend using one uniform global error bound. The fixed
reference sweep contains 25 factors
2^(-k/4), k=0,...,24, covering e0 through e0/64. The candidate is dominated
if a measured reference has both task-feature distance no greater than the
candidate distance and total stored bytes no greater than the candidate
bytes, with at least one strict inequality. A passing candidate must not be
dominated. Only measured points are used. Do not interpolate between measured
reference points.
The measured per-instance reference distances and byte counts are not
disclosed.
````

```
STORAGE SANITY GUARD:
The complete candidate representation must also be strictly smaller than the
fixed lossless reference (FPZIP 1.3.0 full-float32) for the original field. This
lossless byte comparison is a hard storage guard, while the observed
uniform-EB sweep remains the effectiveness comparison. The per-instance
lossless byte count is not disclosed; other lossless-compression statistics
are diagnostics only.
```

D.2.2 ERROR BOUNDING (EB)

The model generates one absolute error bound for each data point. For ZFP, CraftBench assigns each native block the minimum generated bound among the points in that block.

```
YOUR TASK:

Define exactly

    def compute_error_bound_map(field, e0):
        ...
        return eb_map

The returned array must have the same shape as field and contain the absolute
error bound for every data point. Every value must be finite and satisfy

    0 <= eb_map[i] <= e0.

A zero requests exact reconstruction at that point. Return actual error
bounds, not importance scores.

ZFP ONLY:
ZFP operates on 4x4 blocks in 2-D and 4x4x4 blocks in 3-D. CraftBench assigns
each block the minimum generated error bound among the points in that block.
```

D.2.3 CONFIGURATION SELECTION (CS)

The model selects a supported configuration of the specified base compressor. The instance supplies the pointwise error bound e_0, and every resulting field is checked against the same pointwise error bound.

SZ3 configuration schema.

```
{
  "framework": "sz3",
  "compression": {
    "algorithm": "lorenzo_regression" | "interpolation_lorenzo" |
                 "interpolation" | "no_prediction",
    "predictor": {
      "lorenzo": BOOLEAN,
      "lorenzo_order": 1 | 2,
      "regression": BOOLEAN,
      "regression_order": 1 | 2,
      "interpolation": "linear" | "cubic",
      "interpolation_direction": 0 | 1 | 2
    },
    "quantizer": {
      "type": "linear",
      "bins": 256 | 65536,
      "error_bound_mode": "absolute",
      "error_bound": "e0"
    },
    "encoder": {"type": "huffman"},
    "lossless": {"type": "zstd"}
  },
  "decompression": {"mode": "from_stream"},
  "spatial_adaptation": null,
  "correction": null
}
```

ZFP configuration schema.

```
{
  "framework": "zfp",
  "compression": <ONE OF THE FOLLOWING>,
  "decompression": {"mode": "from_stream"},
```

```
  "spatial_adaptation": null,
  "correction": null
}

Fixed accuracy:
{"mode":"fixed_accuracy", "tolerance":"e0",
 "execution": "serial", "self_contained_header": true}

Fixed rate:
{"mode":"fixed_rate", "rate":NUMBER_1_TO_64,
 "execution":"serial"|"omp", "self_contained_header":true}

Fixed precision:
{"mode":"fixed_precision", "precision":INTEGER_1_TO_32,
 "execution":"serial"|"omp", "self_contained_header":true}

Expert:
{"mode": "expert", "minbits": NONNEGATIVE_INTEGER,
 "maxbits":NONNEGATIVE_INTEGER, "maxprec":INTEGER_0_TO_32,
 "minexp": INTEGER_MINUS_149_TO_127,
 "execution":"serial"|"omp", "self_contained_header":true}

For expert mode, maxbits=0 means unlimited; otherwise, maxbits must be at least minbits.

Reversible mode is not allowed. The final decompressed data must satisfy e0.
```

D.2.4 Error Correction (EC)

The model generates a correction strategy that compares the original field with the baseline decompressed field and generates corrections to better preserve the target features. The corrected field must remain within the prescribed pointwise error bound, and the stored correction data is included in the compressed size.

```
C++ CORRECTION INTERFACE:

// Encode side: field is the original data and recon is the base
// decompressed data. Serialize the derived edits into aux.
void make_aux(const float* field, const float* recon, const float* eb,
              int nx, int ny, int nz, std::vector<uint8_t>& aux);

// Decode side: apply the stored edits to recon in place. The original data
// are not available to this function.
void correct(float* recon, const uint8_t* aux, size_t aux_len,
             const float* eb, int nx, int ny, int nz);

CORE REQUIREMENTS:
- make_aux and correct must be fully defined in one C++ translation unit.
- correct may use only recon, aux, eb, and the field dimensions.
- The corrected candidate must satisfy all four official-success conditions
  in the shared task header, including the fixed measured 25-point uniform-EB
  non-dominance comparison.
- The measured per-instance uniform-reference distances and byte counts are
  not disclosed.
```

D.3 Evaluation Procedure and Success Criteria

Each task is evaluated using a single model output. If the output cannot be parsed directly, we deterministically recover the required code or configuration from the same output without invoking the model again. The resulting adaptation is executed once. Compilation errors, runtime errors, invalid outputs, and failed evaluation criteria do not trigger another model call.

Let d_0 denote the feature distance of the base reconstruction and let d_c denote that obtained from the generated adaptation. A generated adaptation succeeds only if all of the following four conditions are satisfied.

First, it must reduce the target feature distance by at least 50%: $d_c \leq 0.5d_0$. Second, the final reconstruction must satisfy the prescribed pointwise error bound: $\|x - \hat{x}_c\|_\infty \leq e_0$. Third, the generated adaptation must not be dominated by any measured point in the uniform global error-bound sweep. The sweep uses $e_k = e_0 2^{-k/4}, \qquad k = 0, \ldots, 24$. A measured reference dominates the generated adaptation if its feature distance and compressed size are both no larger, with at least

one strictly smaller. Fourth, the complete stored representation of the generated adaptation must be smaller than the fixed lossless reference: $B_c < B_{\text{lossless}}$.

The uniform-reference comparison prevents an adaptation from being considered successful when it is dominated by one of the measured uniform-tightening references. The lossless comparison serves as a storage sanity bound by requiring the complete adapted representation to remain smaller than the fixed lossless reference.

E Experimental Details

Table 5: Generation settings. "—" indicates that the parameter was not applicable or was not explicitly configured.

Model	Reasoning	Temperature	Max output tokens
Gemini-3.8-Flash	medium	—	32768
Claude-Sonnet-5	medium (adaptive)	—	32768
Kimi-K2.5	—	0.0	32768
GPT-5.3-Codex	medium	1.0	32768
Qwen3-Coder-Next	—	0.0	32768

Table 5 reports the model names and generation settings used in the evaluation. Parameters not explicitly supplied in the request use the corresponding provider defaults. For models whose evaluated interfaces expose a configurable reasoning level, we use medium reasoning effort, providing a consistent setting across models while balancing reasoning capability with generation cost and latency. Models without a configurable reasoning level use the default behavior provided by their interfaces. GPT-5.3-Codex uses temperature 1.0, the only explicit temperature value supported by the deployed API endpoint. No top-p, top-k, or seed value is explicitly supplied for any of the five models. The model outputs were generated in September 2026. For endpoints that do not expose a separate fixed version identifier, no additional version information is available.

Benchmark programs are compiled and executed on CPU jobs requesting four CPU cores and 16 GiB of memory, with no GPU requested. Generated EB executions use a 60-second limit, while EC compilation and each correction stage use separate 30-second limits. Timed-out programs are recorded as failures and are not retried. CraftBench uses SZ3 and ZFP as the lossy-compression backends and FPZIP 1.3.0 as the fixed lossless reference.

F Additional Results

This section provides additional analyses of the benchmark results, including qualitative examples of generated adaptations, detailed breakdowns of outcomes and compression behavior, and further analyses of the evaluation criteria.

F.1 Examples of Generated Adaptations

To illustrate the adaptations produced by the evaluated models, Figure 5 shows three adaptations for the same `excursion_set_overlap_profile` task on `ncep_air_2020` using ZFP.

The base reconstruction has a feature distance of $d_0 = 7.73 \times 10^{-3}$. The EB adaptation generated by Gemini-3.8-Flash in Figure 5(a) reduces the feature distance to 9.56×10^{-4}, with $B_c/B_0 = 1.47$ and $B_c/B_{\text{lossless}} = 0.43$. The correction generated by Gemini-3.8-Flash in Figure 5(b) reduces the feature distance to 1.52×10^{-3} while modifying only 63 of 10,512 values (0.60%), with $B_c/B_0 = 1.06$ and $B_c/B_{\text{lossless}} = 0.31$. Both adaptations satisfy all four success criteria. In contrast, the correction generated by GPT-5.3-Codex in Figure 5(c) reduces the feature distance to zero but modifies 10,432 of 10,512 values (99.24%), with $B_c/B_0 = 4.54$ and $B_c/B_{\text{lossless}} = 1.33$. It therefore fails C4 because its total compressed size exceeds the lossless reference.

Listing 1 shows parts of the generated programs corresponding to the three adaptations in Figure 5. For the EB adaptation in Figure 5(a), Gemini-3.8-Flash identifies the thresholds related to the task

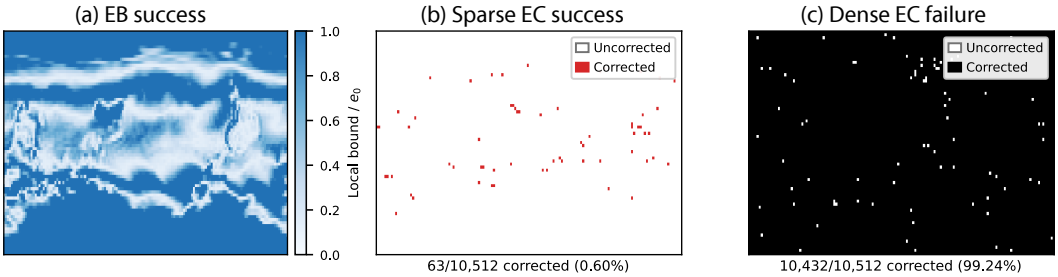


Figure 5: Examples of generated adaptations for the `excursion_set_overlap_profile` task on `ncep_air_2020` using ZFP. (a) Local error bounds from a successful EB adaptation. (b) Corrected locations for a successful Gemini-3.8-Flash correction, where red indicates corrected values; 63 of 10,512 values (0.60%) are corrected. (c) Corrected locations for an unsuccessful GPT-5.3-Codex correction, where black indicates corrected values; 10,432 of 10,512 values (99.24%) are corrected.

and assigns tighter error bounds to values closer to them. For the successful correction in Figure 5(b), Gemini-3.8-Flash identifies points whose threshold classifications differ between the original field and the base reconstruction. It selects points that are misclassified at more thresholds first and stores their original values for correction. In contrast, for the unsuccessful correction in Figure 5(c), GPT-5.3-Codex stores the entire original field as the correction data and uses it to replace the base reconstructed field. This reduces the feature distance to zero but makes the resulting compressed representation larger than the lossless reference.

Listing 1: Selected code from the generated programs corresponding to the three adaptations in Figure 5.

```
# (a) Gemini-3.8-Flash: EB success
def compute_error_bound_map(field, e0):
    work = np.asarray(field, dtype=np.float64)
    mean, std = float(work.mean()), float(work.std())

    thresholds = [mean, mean + std, mean + 2.0 * std]

    min_dist = np.full_like(work, fill_value=np.inf)
    for t in thresholds:
        dist = np.abs(work - t)
        min_dist = np.minimum(min_dist, dist)

    # ...
    scale = np.clip(min_dist / (2.0 * e0), 0.0, 1.0)
    eb_map = (e0 / 8.0) + (e0 - e0 / 8.0) * (scale**1.5)

    return np.clip(eb_map, 1e-6, e0).astype(np.float64)


/* (b) Gemini-3.8-Flash: correction success */
std::vector<Flip> flips;
for (size_t i = 0; i < n; ++i) {
    // ...
    int count = (orig_m0 != recon_m0) +
                (orig_m1 != recon_m1) +
                (orig_m2 != recon_m2);
    if (count > 0) {
        Flip f;
        f.idx = static_cast<uint16_t>(i);
        f.flip_count = count;
        f.diff = std::abs(field[i] - recon[i]);
        flips.push_back(f);
    }
}

std::sort(flips.begin(), flips.end(), [](const Flip& a, const Flip& b) {
    if (a.flip_count != b.flip_count)
        return a.flip_count > b.flip_count;
    return a.diff > b.diff;
});

size_t max_fixes = std::min<size_t>(flips.size(), 80);
// ...
```

```
for (size_t k = 0; k < max_fixes; ++k) {
    uint16_t idx = flips[k].idx;
    float val = field[idx];
    std::memcpy(aux.data() + pos, &idx, sizeof(idx));
    pos += sizeof(idx);
    std::memcpy(aux.data() + pos, &val, sizeof(val));
    pos += sizeof(val);
}
// ...
if (idx < n) {
    recon[idx] = val;
}


/* (c) GPT-5.3-Codex: correction failure */
void make_aux(const float* field, const float* recon, const float* eb,
              int nx, int ny, std::vector<uint8_t>& aux) {
    // ...
    aux.clear();
    append_u32(aux, n);
    for (uint32_t i = 0; i < n; ++i) {
        append_f32(aux, field[i]);
    }
}

// ...
uint32_t n = std::min(nstored, nmax);
for (uint32_t i = 0; i < n; ++i) {
    float v = 0.0f;
    if (!read_f32(aux, aux_len, pos, v)) return;
    recon[i] = v;
}
```

F.2 Outcome and Failure Breakdown

Table 6 summarizes failures of the four success criteria among valid evaluations, and Table 7 separates successful instances from instances that execute successfully but fail one or more evaluation criteria, and from instances that do not produce a valid result for evaluation. Most unsuccessful instances belong to the second group rather than failing during execution.

Table 6: Failures of the four success criteria among valid evaluations.

Criterion	Failures	Fail only this criterion
Feature-distance reduction (C1)	1438	552
Pointwise error (C2)	41	1
Uniform reference (C3)	1417	509
Lossless storage (C4)	60	26

Outcome	Instances	Fraction
Success	282	11.4%
Valid but unsuccessful	1997	80.7%
Invalid or non-evaluable	196	7.9%
Total	2475	100.0%

Table 7: Overall outcomes across all benchmark instances.

Table 8 further breaks down the 196 instances that do not produce a valid result for evaluation. Invalid configurations are the most common cause of these failures, accounting for 86 instances, primarily because some generated configurations violate the required output contract.

F.3 Compression Efficiency

Table 9 reports the median ratio between the complete stored representation of a successful adaptation and the fixed lossless representation. The median remains below 0.5 for every backend and adaptation mode.

Failure type	Instances
Invalid configuration	86
Compilation error	34
Invalid runtime output	25
Runtime crash / manual review	19
Invalid model output	17
Native memory-safety failure	8
Timeout	5
C++ exception	1
Bad output / manual review	1
Total	196

Table 8: Breakdown of instances that fail to produce a valid result for evaluation.

Backend	Mode	Successes	Median B_c/B_{lossless}
SZ3	EB	15	0.239
SZ3	CS	16	0.057
SZ3	EC	56	0.116
ZFP	EB	82	0.395
ZFP	CS	48	0.416
ZFP	EC	65	0.308

Table 9: Storage size of successful adaptations relative to the fixed lossless reference.

F.4 Correction Sparsity and Storage Overhead

Table 10 compares the number of modified values and the size of the stored correction data. Successful corrections modify a median of 189 values and require 663 bytes of correction data, compared with 512 values and 2057 bytes for unsuccessful corrections.

Status	Count	Median modified points	Median correction bytes
Successful	121	189	663
Unsuccessful	704	512	2057

Table 10: Sparsity and storage cost of the EC mode adaptations. Counts include all valid correction instances; medians are computed from available measurements.

Among the 93 EC mode adaptations that fail only the uniform-reference criterion, correction data account for a median of 72.6% of B_c, and the adaptations modify a median of 2102 points. Storage overhead is even more pronounced for the 25 correction instances that satisfy the feature-distance reduction, pointwise-error, and uniform-reference criteria but fail the lossless-storage requirement. The remaining C4-only failure is an SZ3 EB adaptation with $B_c = 23{,}728 > B_{\text{lossless}} = 22{,}276$ bytes. Among these 25 correction instances, 23 are generated by GPT-5.3-Codex, and the adaptations modify 58.1%–99.99% of the field. The median stored correction size is 23,105 bytes, with a maximum of 13,783,971 bytes. Their median B_c/B_{lossless} is 1.331, with a maximum of 2.431.

F.5 Results under Different Feature-Distance Reduction Thresholds

We use a 50% reduction as a moderate target for requiring a clear improvement over the base reconstruction. To examine how the required feature-distance reduction affects the results, we recompute the success rates using reductions of 25%, 50%, and 75%, while keeping the other three success criteria unchanged. The overall success rate decreases from 13.1% at 25% reduction to 11.4% at 50% and 8.3% at 75%; Table 11 reports the corresponding task-group rates at each threshold.

Table 11: Success rates (%) under different required reductions in feature distance.

Task group	25%	50%	75%
Numerical and distributional	13.0	11.5	7.8
Differential and spatial	9.3	7.9	5.6
Geometric and topological	18.6	17.0	12.9
Spectral and multiscale	10.0	7.7	5.3
Composite	10.9	8.7	5.8
Domain-specific	21.3	20.0	16.3
Overall	13.1	11.4	8.3

F.6 SZ3 Configuration Achievability

We enumerate every legal SZ3 configuration for each task. The configuration space contains 480 configurations for each 2D task and 720 configurations for each 3D task. Across the 74 two-dimensional and 9 three-dimensional tasks, we perform 42,000 evaluations in total.

As reported in Section 5.5, only 10 of the 83 tasks have at least one SZ3 configuration that satisfies all four success criteria. Across these tasks, 16 of the 50 generated configurations are successful (32.0%), and 8 of the 10 tasks have at least one successful adaptation.

The low SZ3 configuration success rate reflects both the limited number of tasks for which a successful configuration exists and the models' inability to select such a configuration for every achievable task.

F.7 Fixed-Configuration Comparison

Section 5.3 shows that Kimi-K2.5 repeatedly selects the same ZFP expert-mode configuration among its successful outputs. We therefore perform a post hoc comparison using its most frequent successful configuration, `minbits=0`, `maxbits=0`, `maxprec=16`, and `minexp=-20`, unchanged for all 82 ZFP configuration instances, and evaluate it using the same C1–C4 criteria.

The fixed configuration satisfies C1 on 65/82 instances, C2 on 82/82, C3 on 38/82, and C4 on 82/82, and satisfies all four criteria on 34/82 instances (41.5%). In comparison, Kimi-K2.5 succeeds on 32/82 instances (39.0%). The two approaches succeed on 31 common instances; one instance succeeds only with the Kimi-K2.5-generated configuration, three succeed only with the fixed configuration, and 47 succeed with neither.

F.8 Extended Uniform-Reference Analysis

The C3 criterion measures each candidate on 25 measured uniform-error-bound references from e_0 to $e_0/64$. To analyze whether a tighter bound beyond this range could dominate a successful candidate, we perform an additional analysis using uniform references over a wider range.

Among the 282 successful adaptations, 96 achieve a feature distance below the $e_0/64$ reference. In 65 of these cases, the candidate is already no larger than the $e_0/64$ reference in compressed size. We extend the uniform-reference range through $e_0/4096$ for the remaining 31 cases. 10 of these 31 cases become dominated by an additional uniform reference, while 21 remain undominated. Since the analysis considers only a finite set of uniform error bounds, domination by an unmeasured bound remains possible.

The result shows that the measured success rate depends on the reference range. Because adding further uniform references can only turn a current C3 pass into a failure, the reported 282 successes constitute an upper bound under more exhaustive uniform-reference evaluation, while the 509 observed C3-only failures constitute a lower bound on the number of dominated adaptations. A finite sweep cannot exhaustively cover all possible uniform error bounds.

F.9 Additional 2D Field Analysis

To investigate whether the relatively small 2D evaluation field affects the observed success rates, particularly through the storage overhead of error maps and correction data, we additionally evaluate

Table 12: Success rates on the original NCEP field and the larger GFS field for the same matched 2D adaptations.

Mode	Matched	NCEP	GFS
EB	740	76 (10.3%)	88 (11.9%)
CS	740	54 (7.3%)	105 (14.2%)
EC	576	79 (13.7%)	73 (12.7%)
Overall	2056	209 (10.2%)	266 (12.9%)

the generated 2D adaptations on a 721×1440 NOAA Global Forecast System (GFS) 0.25° 2-m temperature field (NOAA National Centers for Environmental Information, 2020), without additional model calls.

We match adaptations between NCEP and GFS by model, backend, mode, and task. A total of 164 correction adaptations use LLM-generated 16-bit flattened spatial indices that are valid for the original field shape but cannot represent the larger GFS grid, so we exclude them. This leaves 2,056 matched adaptations; outputs that were already invalid in the original benchmark are counted as failures. Thus, the correction results apply only to the matched subset. For the GFS evaluation, we recompute d_0, the $0.5d_0$ target, all 25 uniform-reference distances and compressed sizes, and B_{lossless} on the GFS field.

Across the matched adaptations, 209 (10.2%) succeed on NCEP, and 266 (12.9%) succeed on GFS. As shown in Table 12, success rates for EB increase from 10.3% to 11.9%, those for CS from 7.3% to 14.2%, while those for EC decrease from 13.7% to 12.7%. The higher overall success rate shows that field size can affect the measured outcomes, but the effect differs across adaptation modes: the largest increase occurs in configuration, while correction slightly decreases.