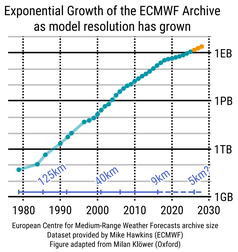
Towards Safe, Trusted, and Fearless Lossy Compression of Earth Science Data

EGU26-9673

Juniper Tyree[1,5], Daniel Köhler[1], Robert Underwood[2,3], Clément Bouvier[1,5], Tim Reichelt[4], Heikki J. Järvinen[1,5], and Milan Klöwer[4]

[1] INAR, University of Helsinki, [2] Argonne National Laboratory, [3] University of Chicago, [4] University of Oxford, [5] ESiWACE3

Earth System Data is growing Exponentially but data budgets are stagnating



Bigger models, ensembles, CMIP

Faster compute & AI surrogates

Storage is increasingly limiting climate model outputs [1]

Storage per FLOP is decreasing, as budgets stagnate, e.g. [2]

Lossless compression of floats is insufficient with only $\times 2-3$

Lossy Compression is Necessary but *Scary*

Compressors can be incorrect or surprising: Hallucinations, error bound violations, …

Compression may violate **variable-specific properties**: e.g. physical limits, multi-variable relationships, …

Compression may disrupt **downstream applications**: e.g. errors over derived **quantities of interest**

Not all compressors can support all use cases

Not all user can validate every compressor

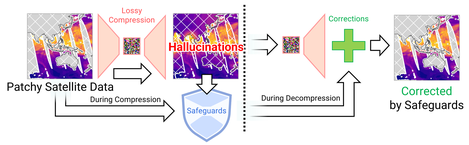
Compression Safeguards:

(1)**Declarative** (2)**Modular** (3)**User-Centric**
(4)**Guaranteed Safety** (5)**for *any* Compressor**

(1+3): User-declared Safety Requirements: "what must compression preserve?", e.g. topology via local ordering

(2+5): **Any compressor can be safeguarded**; data is decompressed & checked for violations mid-compression

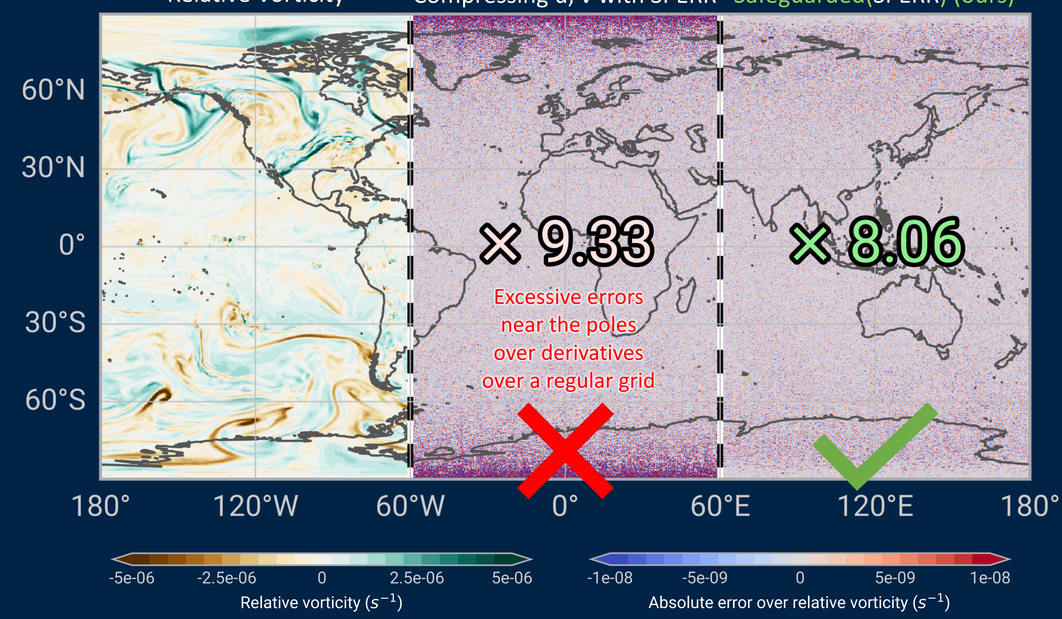
(4): **Safety is guaranteed** by producing quantized **corrections** if necessary, and **checking again** afterwards



Scared of Lossy Compression?

Use Compression Safeguards!

Sharing is encouraged



Example: Bound Absolute Error for Vorticity derived from u-v wind

Safeguarding Relative Vorticity as a Quantity of Interest requires several *Novel* Features

Multiple Variables v & u

Depends on Latitude ϕ coordinate (nD auxiliary data)

Relative Vorticity (QoI) → $\zeta = \left(\frac{\partial v}{\partial \lambda} - \frac{\partial (u \cdot \cos\phi)}{\partial \phi}\right) \cdot (R_e \cdot \cos\phi)^{-1}$

Periodic Longitude λ (Boundary Condition)

Derivatives (Finite Difference)

Earth Radius Constant (scalar auxiliary data)

Scan for more!

What can You Safeguard?

Simple: Missing Values, Isosurfaces

Regionally Varying Error Bounds over x and $f(x)$

Quantities of Interest $f(x)$ over rolling windows:

polynomials, x^y, log, sin, acos, tanh, $\lfloor x \rfloor$, not, all, any, where$(x > c, y, z)$, isnan(x), $A \cdot B$, finite differences, monotonicity, extra data …

Boundary Conditions across Data **Chunks**

And any **Composition** of the above, e.g. dSSIM

How can You Safeguard?

compression-safeguards: sans-compression

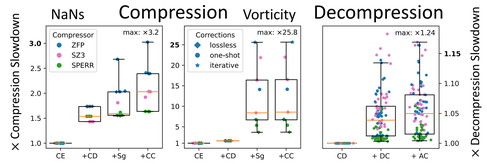
numcodecs-safeguards: metacompressor

xarray-safeguards: lazily across chunked data

Concise and consistent JSON-based config

How Costly are the Safeguards?

Compression: depends on safeguards, $\times 2-10k$
Decompression: typically, 5-10% overhead



CE: Compressor Encode, CD: Compressor Decode, **Sg: Safeguards**, CC: Compress Corrections, DC: Decompress Corrections, AC: Apply Corrections

QPET vs OptZConfig vs Safeguards

	QPET [3]	OptZConfig [4]	Safeguards (ours)
Method	Iteratively tune pointwise error bounds + Lossless corrections fallback	Iterative blackbox optimization of config + Fully-lossless compression fallback	Pointwise non-lossless corrections, Final layer of defense for end users
Compressor	SZ3 or SPERR (needs coupling)	**Arbitrary** (blackbox)	**Arbitrary** (blackbox)
Data Domain	Structured, pw-only unstructured	Structured, Extensible for unstructured	Structured, pointwise-only unstructured
Error Norm	Pointwise absolute, range-relative	**Many** through LibPressio, extensible	Pointwise absolute, relative, ratio
Quantities of Interest …	Any differentiable	Blackbox, extensible, requires global result	Any **piecewise-invertible** with pointwise output
… with output domain	Pointwise, block mean, multi-variable	Pointwise, extensible with custom C++ code	Pointwise, moving window stencil, stacked
… with auxiliary data	✗	✗	**Arbitrary** (coordinates, regions of interest, …)
… with boundary conditions	✗	Extensible with custom C++ code	**valid, const, edge, reflect, symmetric, wrap**
… with safe cross-chunk boundaries	✗	✗	✅
Combined Requirements	Pointwise all	Extensible with custom C++ code	**Pointwise any, all, where** (for pw & stencil reqs)
Strength vs Weakness	**High compression vs Limited features**	**All is Extensible vs Lossless fallback**	**Featureful & Composable vs Costly corrections**

Safeguards **do not replace** other compressors, and they can be used **with** QPET and OptZConfig

[1] Cappello, F., et al. (2025). What to Support When You're Compressing. SC '25 , Proceedings. doi:10.1145/3712285.3759856.
[2] Peters-von Gehlen, K., et al. (2025). Lossy Data Compression Exploration in an Online Laboratory and the Link to HPC Design Decisions. EGU25. doi:10.5281/zenodo.15403734.
[3] Liu, J., et al. (2025). QPET: A Versatile and Portable Quantity-of-Interest-Preservation Framework for Error-Bounded Lossy Compression. Proc. VLDB Endow. 18(8). doi:10.14778/3742728.3742729.
[4] R. Underwood, J. et al. (2023). OptZConfig: Efficient Parallel Optimization of Lossy Compression Configuration. IEEE Transactions on Parallel and Distributed Systems. 33(12). doi:10.1109/TPDS.2022.3154096.