Can we Aggressively Reduce Scientific Data Without Losing Science?

- Why lossy data reduction?
- SZ lossy compressor
- Z-checker: reduction error analysis
- Compression vs. Decimation

Franck Cappello

Argonne National Laboratory

University of Illinois at Urbana Champaign

With major contribution from: Sheng Di[#], Dinwen Tao[+]

[#]ANL, [+]U. California Riverside.

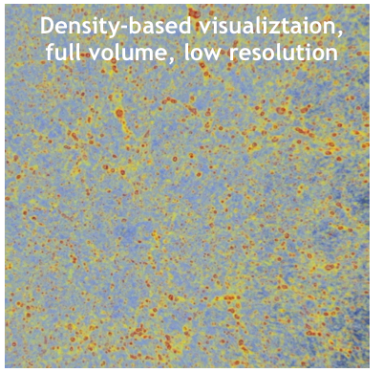
EXASCALE COMPUTING PROJECT

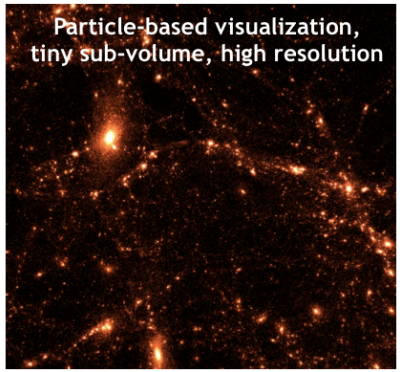
Argonne NATIONAL LABORATORY


Why do we need
to reduce scientific datasets?
Image from the Argonne Photon Source

Scientific Simulations and Experiments: the flood of data!

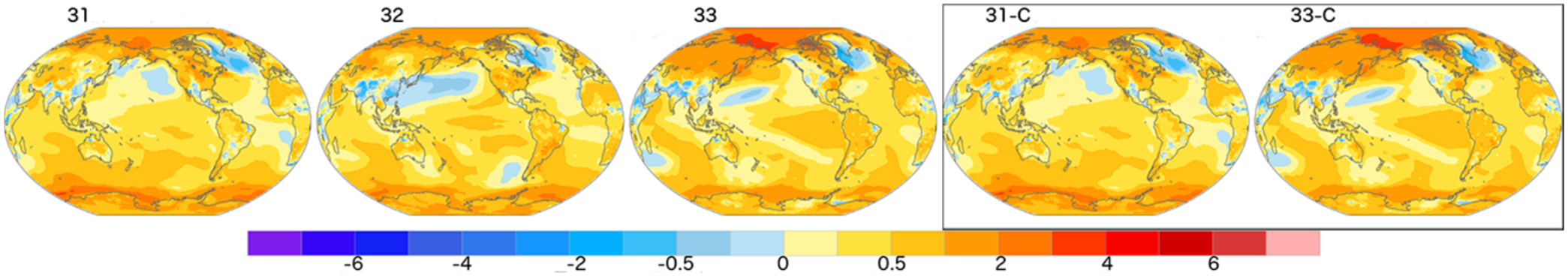
- Today's scientific research is using simulation or instruments and produces extremely large of datasets to process/analyze





- Feasibility problems:
  - Cosmology Simulation:
    - A total of >**20PB** of data when simulating trillion of particles
    - Petascale systems FS ~20PB
      (you will never have 20PB of scratch for one application)
    - On Blue Waters (1TB/s file system), it would take
      20 X 10^15 / 10^12 seconds (5h30) to store the data
  - Data reduction of a bout a factor of 10 is needed
    → currently drop 9 snapshots over 10 (also called decimation in time)

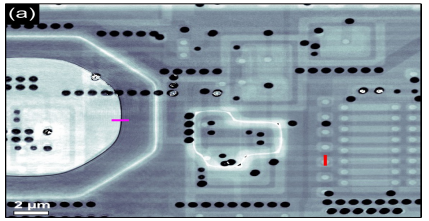
Data reduction is needed for many scientific simulations



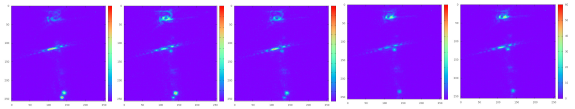
- High-resolution CESM simulation (1/10° ocean-ice, 1/4° atmosphere)
  → 1 TB of data generated per compute day
- IPCC Coupled Model Comparison Projects (CMIPs)
  - Phase 5 (2013) → 2.5 PB of output
  - Phase 6 (2018)→ >10 PB expected!
- The relative cost of storage is increasing...
  - Previous NCAR platform (2013): ~20% of hardware budget
  - Current NCAR platform (2017): ~50% of hardware budget
    → Work of Allison H. Baker, NCAR, on compression of climate datasets (HPDC2014)
    → Data reduction of about a factor of 10 is needed

Data reduction is also needed for experimental sciences

- APS-U (next-generation APS project at Argonne):
  - Brain Initiatives: in the order of **100PB** of storage: hundreds of specimens, each requiring 150TB of storage.
  - Data analysis is performed on ANL Mira (several miles)
  - Connection between APS and Mira: 100Gb/s
    - Would take ~10^17/10^10 = 10^7 seconds to transfer the data: 115 days to move the data
    - There is no 100PB buffer at the APS.



- APS-U users need to use filters to reduce the data size



→ Data reduction of about a factor of 100 is needed

When data is too big

It cannot be communicated, stored or processed plain

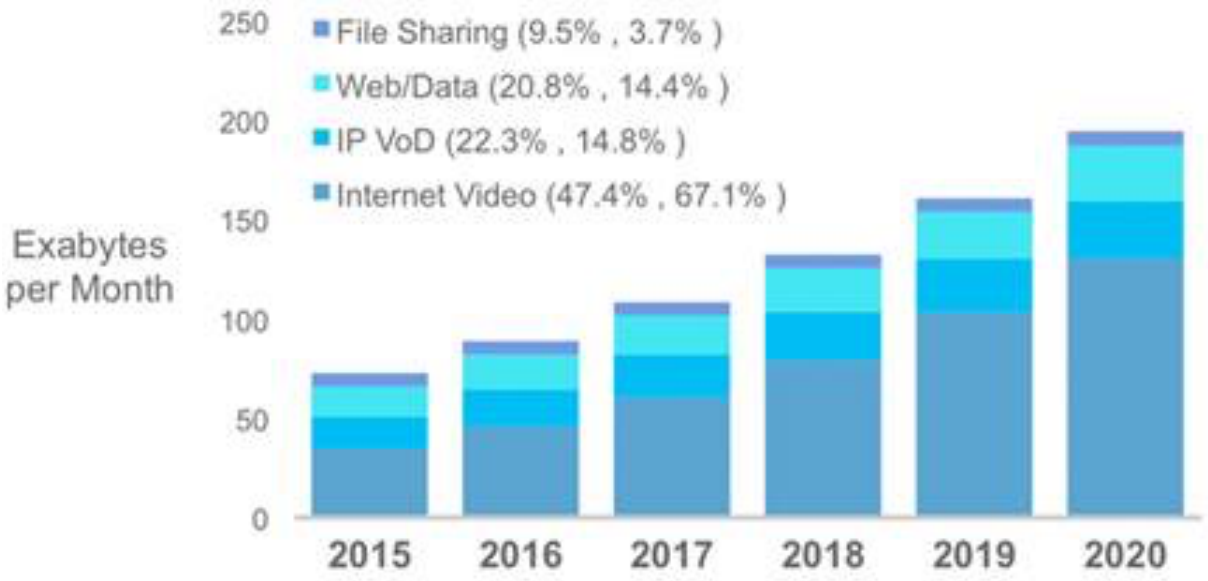
→ Data needs to live in reduced version

CISCO statistics:

- Annual global IP traffic: 2.3 ZB (10^21) in 2020
- IP video traffic will be 82 percent of all IP traffic

→ >>80% of the IP traffic will be compressed!

→ >>80% of the files exchanged in the Internet will exist (probably only) in compressed version

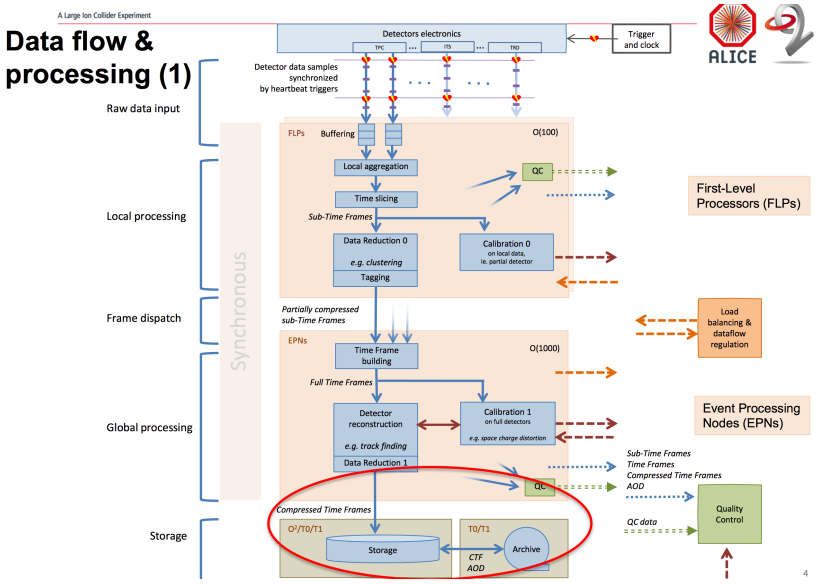


Data reduction techniques

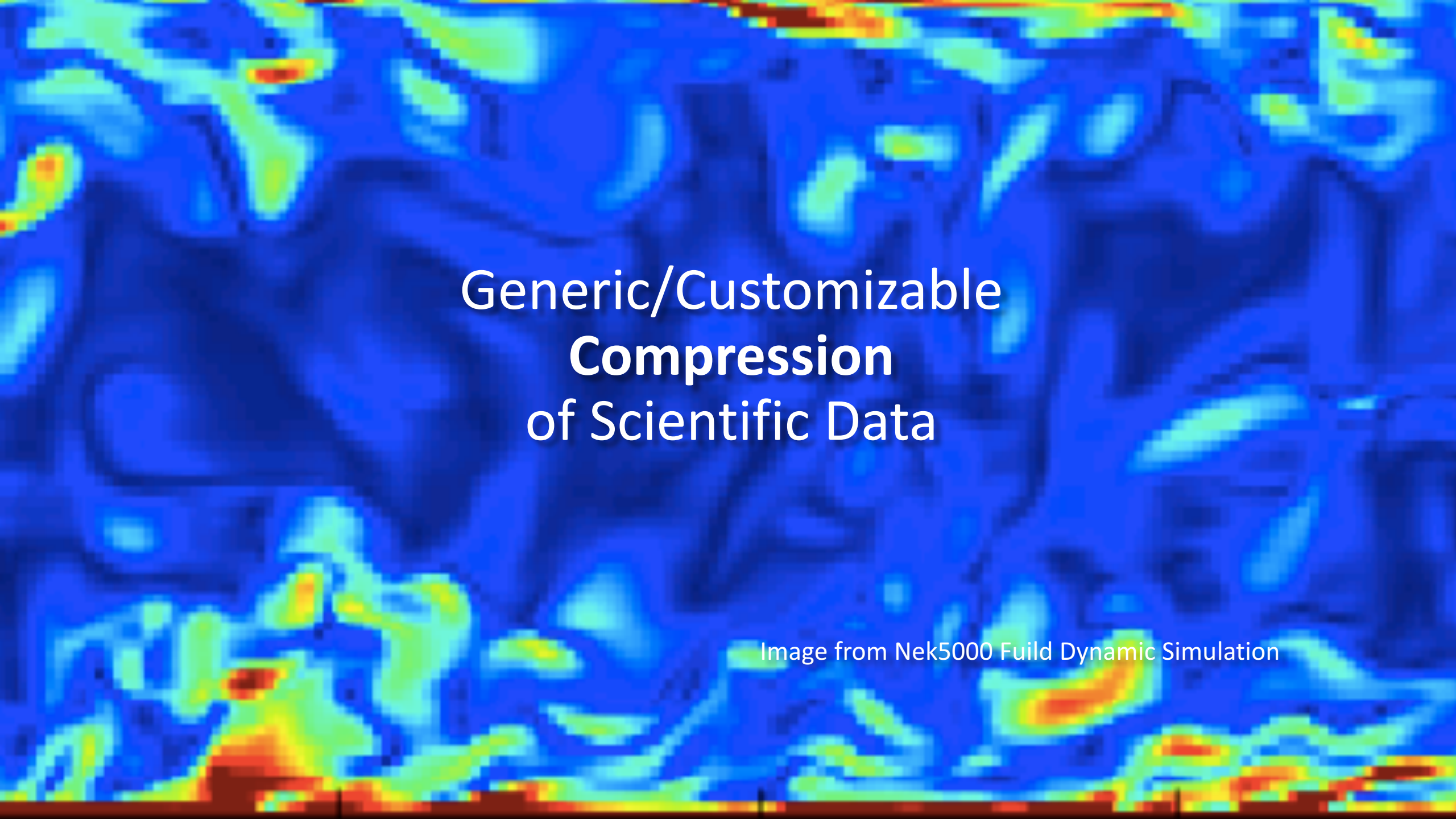
- Generic Decimation:
  - Just remove m datum over n. Typical example: remove 9 data over 10. 90% loss
  - In space (e.g. coarser mesh size) and/or in time (remove snapshots)

- Specific filtering:
  - E.G. ATLAS raw data per event: 1MB at 100Mhz (Run 2).
  - Run 3 (2021) will produce events at 60TB/s!!

  →Goal is to reduce data rate to storage to 20 GB/s (/1000).
  - O2 framework design: 463 FPGAs detector for readout and fast cluster finding, 100,000 cores to compress 1.1 TB/s streams.



- **Generic / customizable compression**


Generic/Customizable
Compression
of Scientific Data
Image from Nek5000 Fuild Dynamic Simulation

Data compression

- A form of data reduction
- ~40 years old (LZ77, 1977), ~70 years (Shannon's information theory, 1948)
- Systematically (almost) used for files (e.g. GZIP), digital photos (e.g. JPEG), movies (e.g. MP4), music (e.g. MP3)
- Very effective on images
- A priori compression algorithms are generic and applicable to many applications
  - In practice they are combined and optimized for specific usages

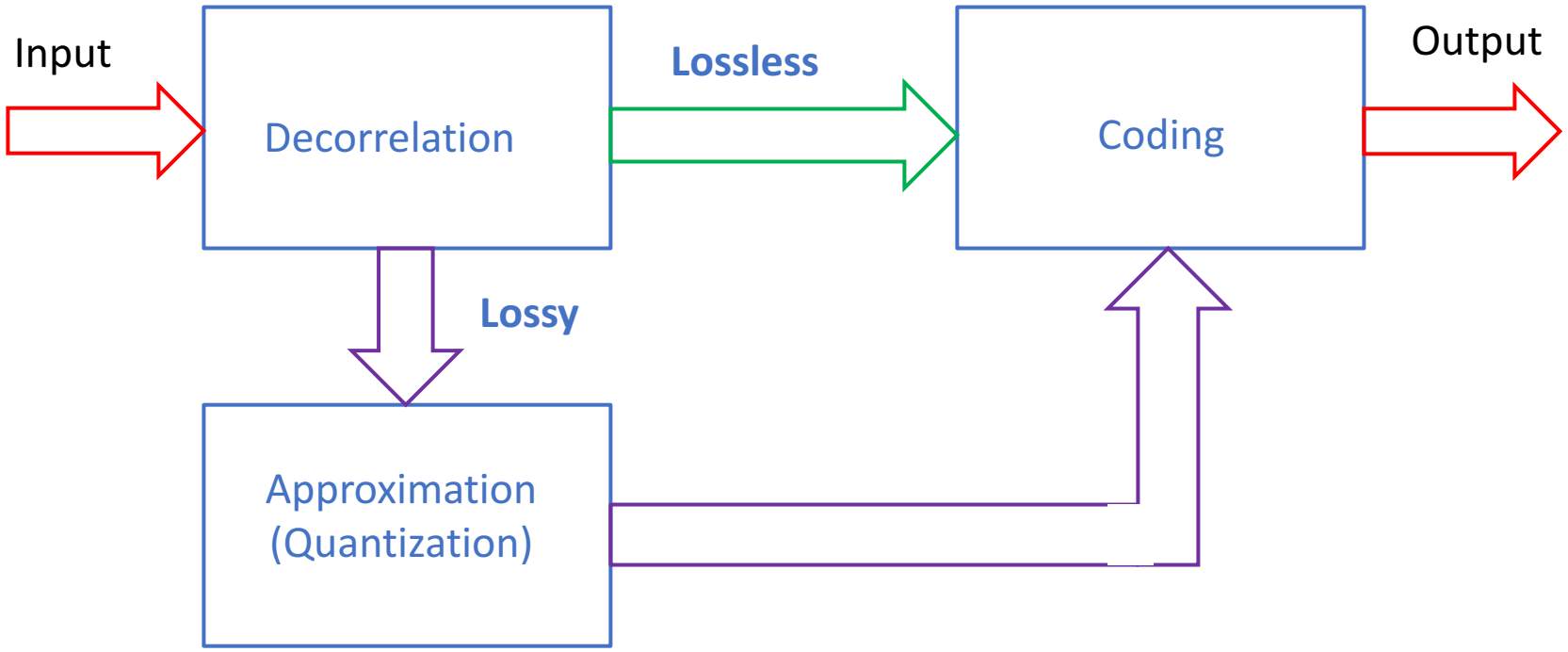
Compression

- **Lossless**:
  - The data is compressed and decompressed in such a way that the decompressed data is **identical** to the initial data (there is no alteration, deviation, distortion)
  - Best example: Gzip (will have a slide on it)

- **Lossy**:
  - The data is altered during compression: some piece of information is removed (lost): the original data cannot be retrieved.
  - If user can control the accuracy of the compressed data, then lossy compression is a trade-off between compression factor and loss of accuracy
  - Best example: JPEG (will have a slide on it)

General Compression Process for scientific data

- Two or three main stages (each stage may use multiple sub-stages)



What scientific data looks like?

- Floating point datasets

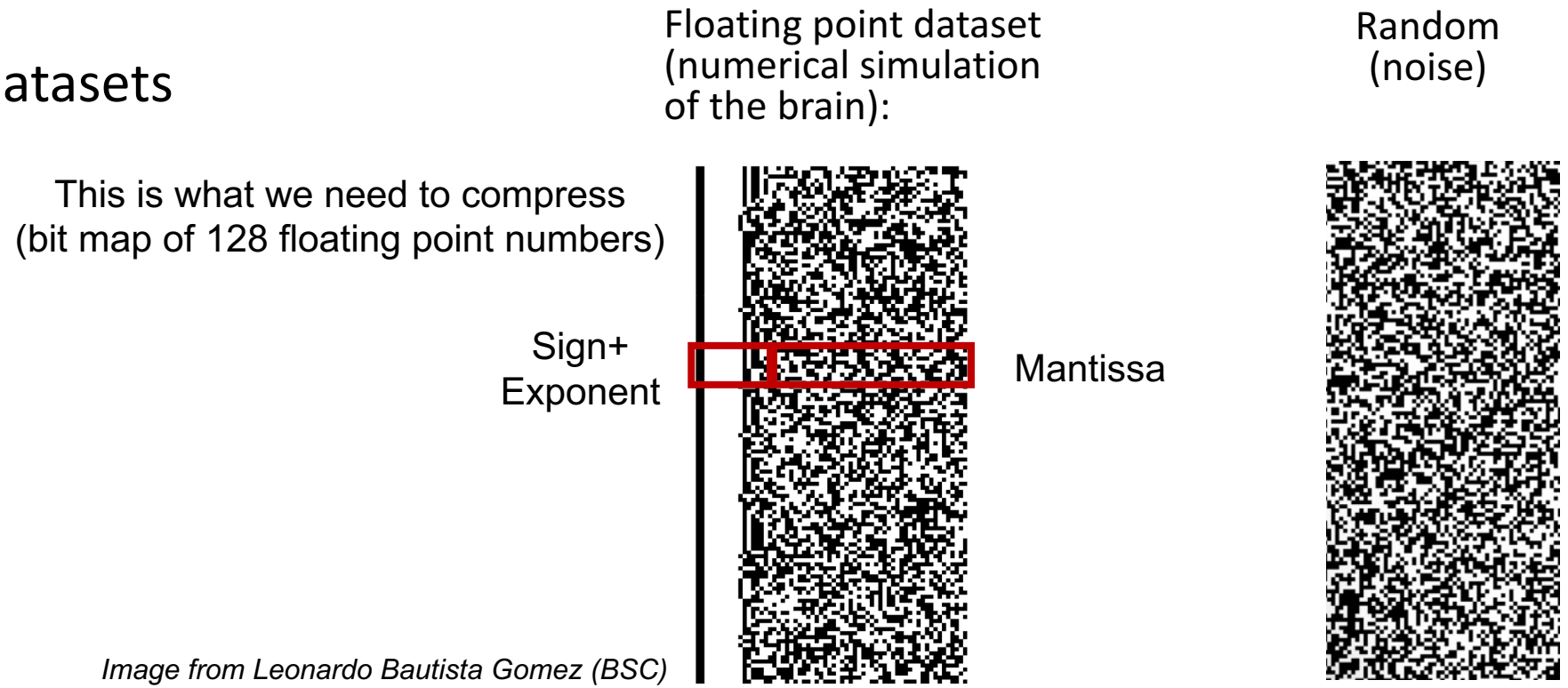


Image from Leonardo Bautista Gomez (BSC)

- Floating point datasets looks very random at first sight (except for the sign and the exponent)

Entropy of Datasets

Entropy of the string of symbols: $H = -\sum p(x) \log p(x)$

p(x) being the probability of occurrence of x in the string
log p(x): information quantity carried by observing x

Maximum compression factor of ~2 according to Shannon Entropy (datasets are 64 bits)

Note: 25.17 = Log_2(dataset size) 384 x 384 x 256
→ Every value is unique

From: Lindstrom, Peter, and Martin Isenburg. "Fast and efficient compression of floating-point data." *Visualization and Computer Graphics, IEEE Transactions on* 12.5 (2006): 1245-1250.

data set						
name	unique (%)	entropy (bits)	range (bits)	min	max	size (MB)
m2d density	3.89	3.49	21.83	8.7E-01	1.2E+00	19.6
m2d vorticity	99.20	22.25	31.05	-1.4E+02	2.5E+01	19.6
m3d density	7.67	5.16	23.60	1.0E+00	3.0E+00	364.5
m3d pressure	27.29	23.91	31.06	-3.7E+00	2.3E+03	364.5
m3d diffusivity	36.87	23.19	30.02	0.0E+00	6.8E+00	364.5
m3d viscocity	50.07	24.86	28.59	8.6E-15	2.9E+00	364.5
h3d temp	65.70	23.54	31.56	-7.7E+01	1.0E+35	95.4
h3d pressure	81.82	24.13	31.58	-3.4E+03	1.0E+35	95.4
h3d x velocity	84.18	24.18	31.55	-5.3E+01	1.0E+35	95.4
h3d y velocity	84.32	24.18	31.55	-4.6E+01	1.0E+35	95.4
h3d z velocity	86.82	24.24	31.54	-3.2E+00	1.0E+35	95.4
M3d density	40.14	18.84	52.59	1.0E+00	3.0E+00	288.0
M3d pressure	100.00	25.17	63.00	-2.2E+00	2.2E+00	288.0
M3d x velocity	100.00	25.17	63.00	-2.2E+00	2.3E+00	288.0
M3d y velocity	100.00	25.17	63.00	-2.1E+00	2.3E+00	288.0
M3d z velocity	100.00	25.17	63.00	-5.2E+00	9.0E+00	288.0
a3d x position	61.10	23.82	31.01	-4.8E-02	4.6E+02	107.7
a3d y position	45.90	23.32	26.99	3.7E-02	2.1E+03	107.7
a3d z position	61.68	23.84	27.48	9.1E-05	4.6E+02	107.7
a3d y velocity	64.65	23.87	30.96	-1.5E-01	1.4E-01	107.7
a3d temp	64.91	23.94	27.41	3.0E-03	7.1E+03	107.7
a3d energy	3.45	18.57	21.79	-3.6E+00	-2.7E+00	107.7
lucy	61.39	24.38	31.09	-6.1E+02	1.2E+03	160.5
david$_{1mm}$	25.23	17.08	31.11	-4.4E+03	1.8E+03	322.5
torso	84.72	18.48	31.08	-2.7E+02	5.8E+02	1.9
rbl	71.90	20.14	25.99	1.48E+00	3.6E+02	8.4

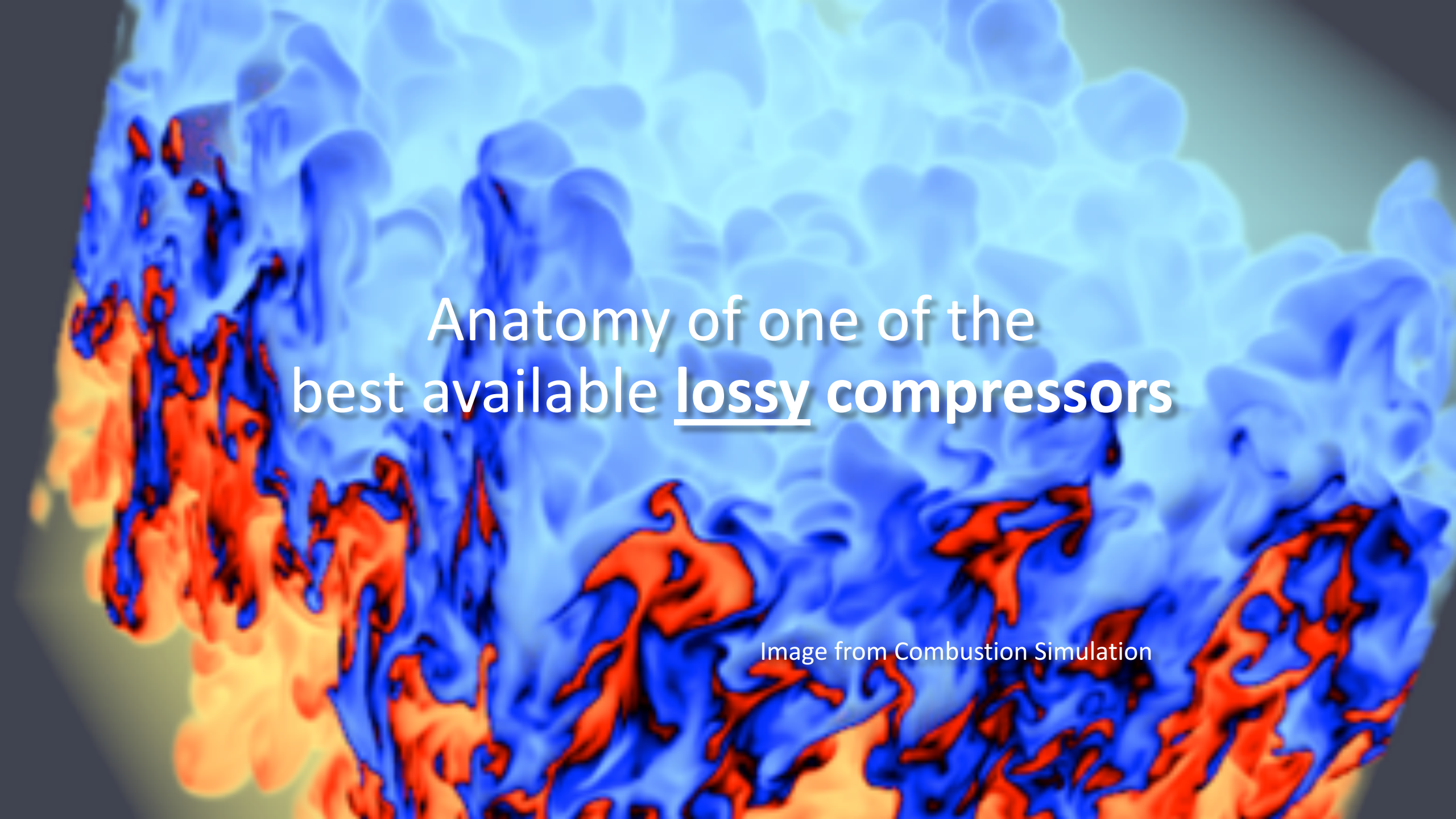
What advanced lossless compressors?

3 years ago

Scheme	Transformation Applied	Algorithm	Compression Ratio
FPC [8]	not used	it first predicts values sequentially using two predictors (FCM and DFCM), and subsequently selects the closer predicted value to the actual. Lastly, it XORs the selected predicted value with the actual value, and leading-zero compresses the result.	1.02x∼1.96x
ISOBAR [30]	divide byte-columns into compressible and incompressibles	apply zlib, bzlib2, (fpzip, FPC) on all compressible (after discarding noisy byte-columns). zlib is the main compression algorithm; others are for comparison purposes	1.12x∼1.48x
PRIMACY [31]	frequency based permutation of ID values	apply zlib on transformed data	1.13x∼2.16x
ALACRITY [19]	split floating-point values into sign, exponent, and significands	unique-value encoding of the most significant bytes (assuming high-order bytes (sign and exponents) are easy to compress); low-order bytes are compressed using ISOBAR	1.19x∼1.58x
CC [6]	XOR on Δ of neighboring data point in the same iteration	apply zero-filled run length encoding	up to 2.13x
IOFSL [36]	not used	integration of LZO, bzip2, zlib within the I/O forwarding layer	∼1.9x
Binary Masking [5]	bit masking (XOR)	apply zlib on bit masked data in order to partially decreases the entropy level	1.11x∼1.33x
MCRENGINE [18]	variable merging in the same group	apply parallel gzip on the merged variables across processes	up to 1.18x

Son, S. W., Chen, Z., Hendrix, W., Agrawal, A., Liao, W. K., & Choudhary, A. (2014). Data Compression for the Exascale Computing Era-Survey. *Supercomputing frontiers and innovations*, *1*(2), 76-88.

Compression limited to a factor of 2 in most cases


Anatomy of one of the
best available **lossy** **compressors**
Image from Combustion Simulation

ANL SZ Lossy compressor

Design considerations:

- Effective: compete with the best existing lossy compression algorithms
- Efficient, fast: only use linear complexity steps (no sorting)
- Strict respect of error bounds set by users
- Allow different types of error bounds: absolute, relative, relative to value range (request from users), PSNR
- Target all types of scientific data: 1D to nD, Structured, Unstructured, Particles, Instruments (images)
- Produce acceptable distortions (error autocorrelation, spectral, derivatives)

ECP EXASCALE COMPUTING PROJECT

SZ is supported by the DOE/NNSA **ECP EZ and ExaSky** Projects and the **NSF Aletheia** Project.

Argonne NATIONAL LABORATORY

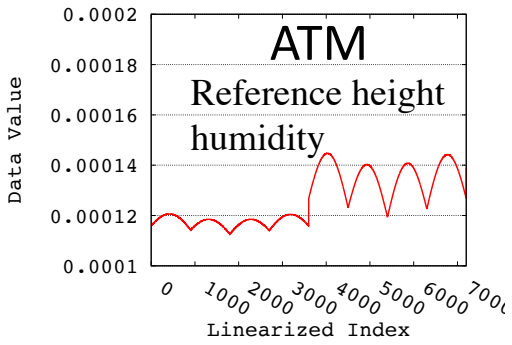
How do we compress
(leverage redundancy)

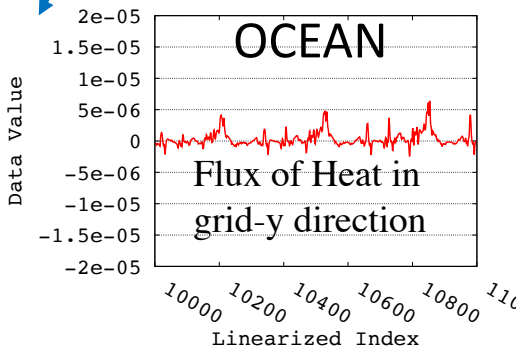
Similarities, autocorrelation, smoothness

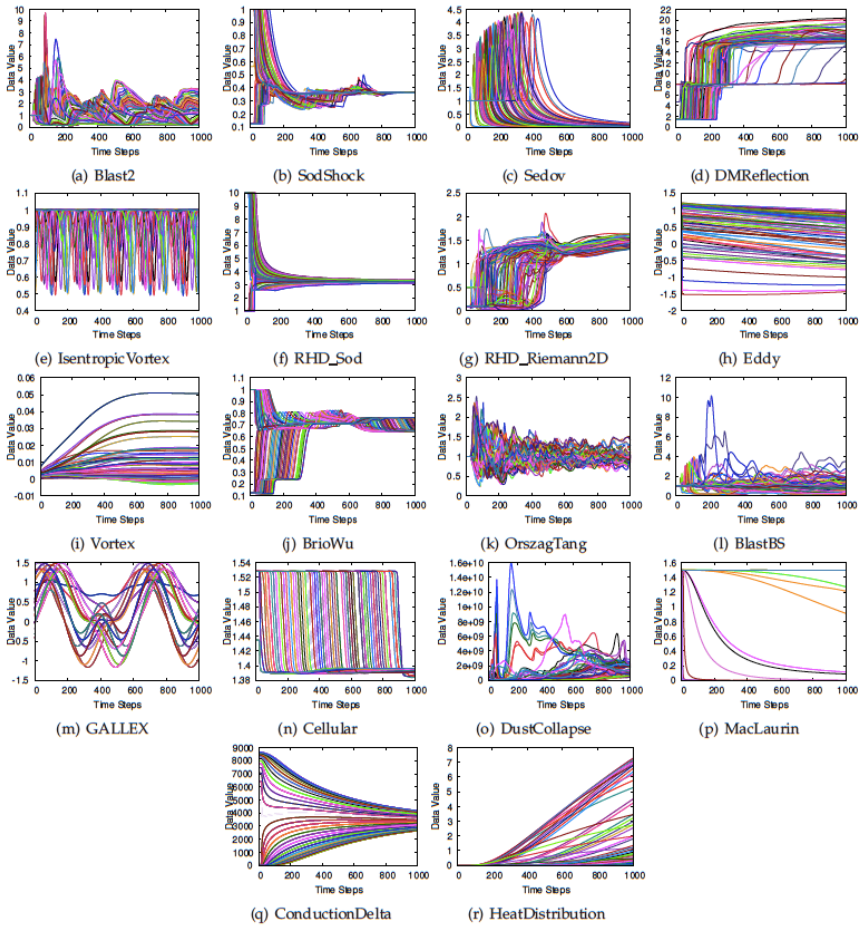
Smoothness → **Prediction**

Plotting the values as time series

Time/ Space

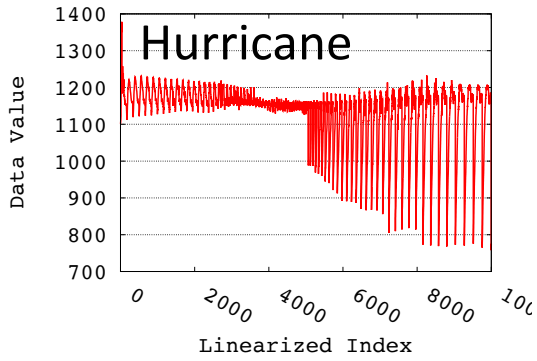


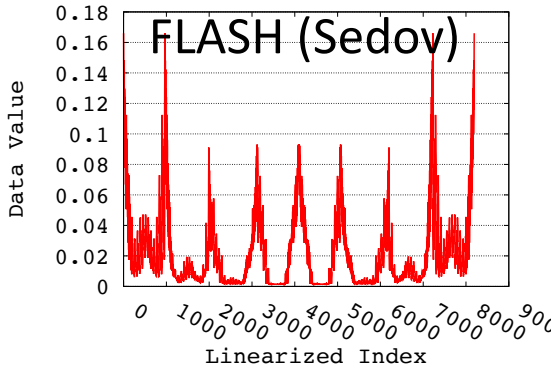


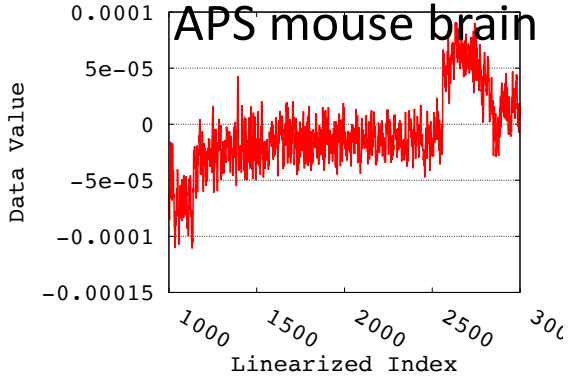


Not all datasets are smooth

Not smooth
→ **Deal with prediction error**



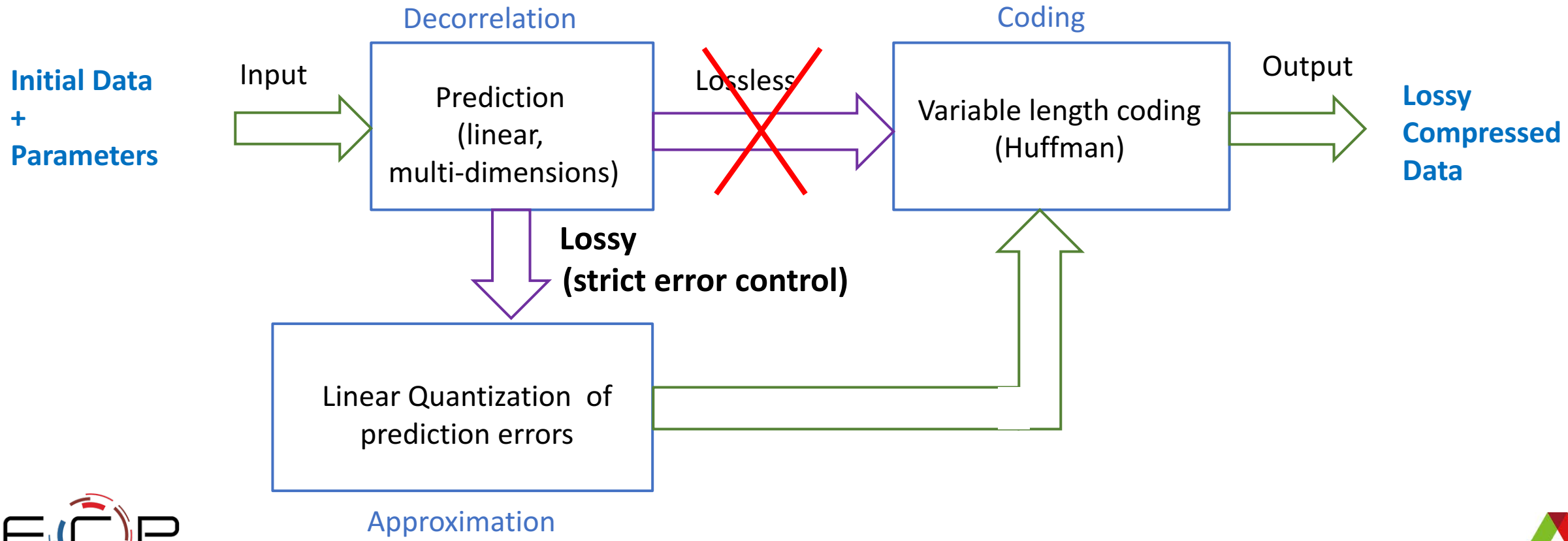




EXASCALE COMPUTING PROJECT

SZ Design Principles

- Prediction based lossy compressor
- Multi-stages



S. Di, F. Cappello, Fast Error-bounded Lossy HPC Data Compression with SZ, **IEEE IPDPS 2016**

Controlling the compression error

- SZ provides point wise error controls (input parameters)
- Three types of error (E: error, V: intial dataset, V': decompressed dataset)

  - Absolute error: (small value range), $E_a = |V - V'|$
  - Relative error: (large value range), $E_r = \frac{|V - V'|}{V}$
  - Relative to value range: $E_{vr} = \frac{|V - V'|}{Vmax\text{-}Vmin}$

SZ Prediction Stage

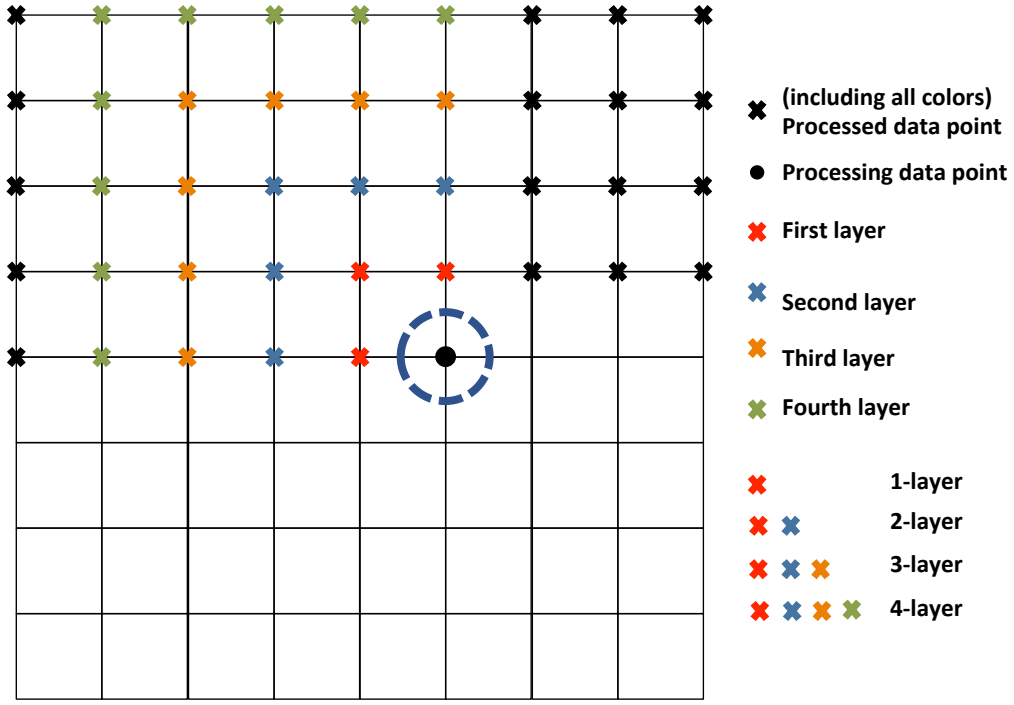
5 steps

Example for 2D:

$$f(i_0, j_0) = \sum_{0 \le k_1, k_2 \le n}^{(k_1,k_2) \neq (0,0)} (-1)^{k_1+k_2+1} C_n^{k_1} C_n^{k_2} V(i_0 - k_1, j_0 - k_2)$$

1) Multi-dimensional Multi-layer **Prediction** (extension of Lorenzo)

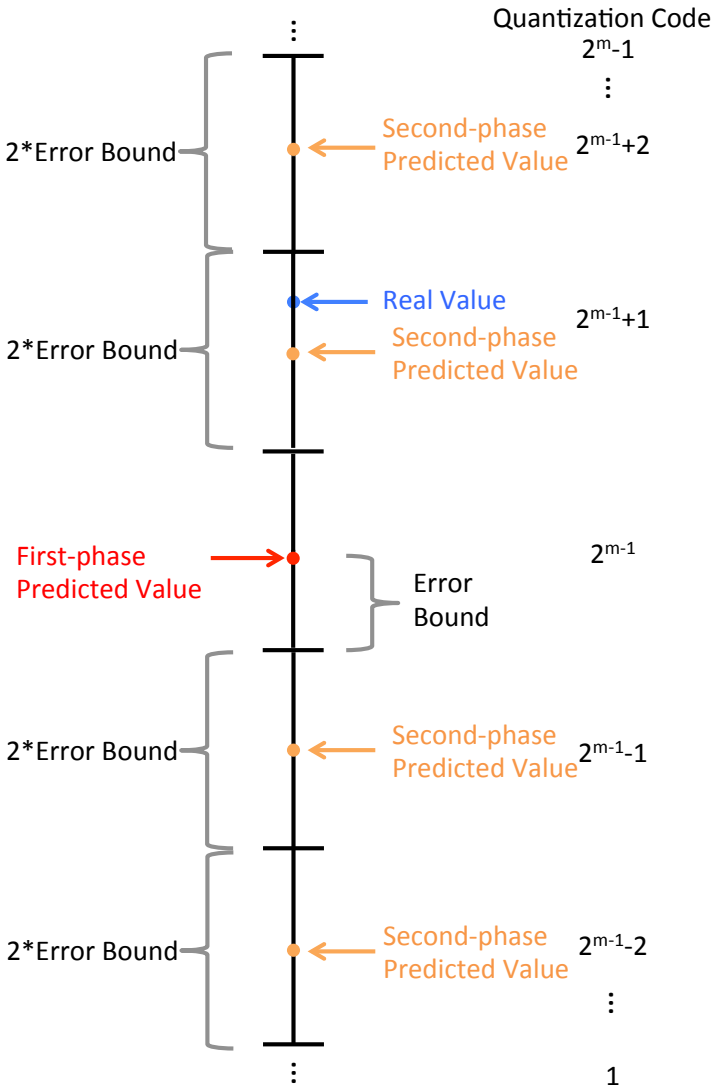
Input floating Point data



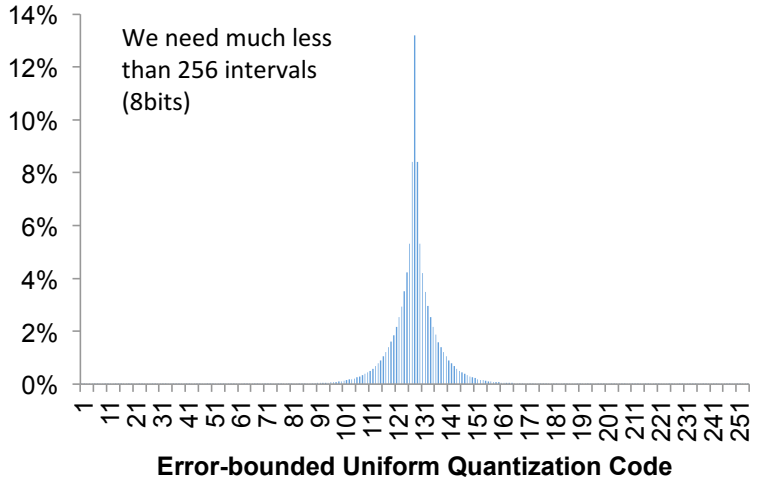
Produces floating point Numbers (predictions)

SZ Quantization Stage

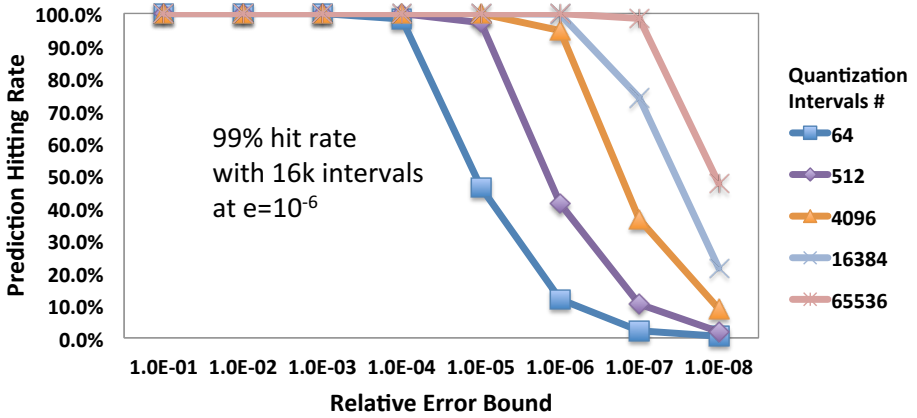
2) **Linear Quantization of prediction error (map data into quantization bins, #bins defined by users or SZ)**



Example for err: 10^{-4}, for ATM



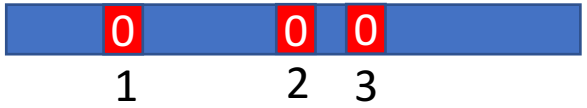
Example: hit rate for ATM



- Transforms each predicted data into 1 integer value (with loss)
- If data is out of scale, keep it in a separate array

Initial data (FP numbers)

Array of quantization (integers)



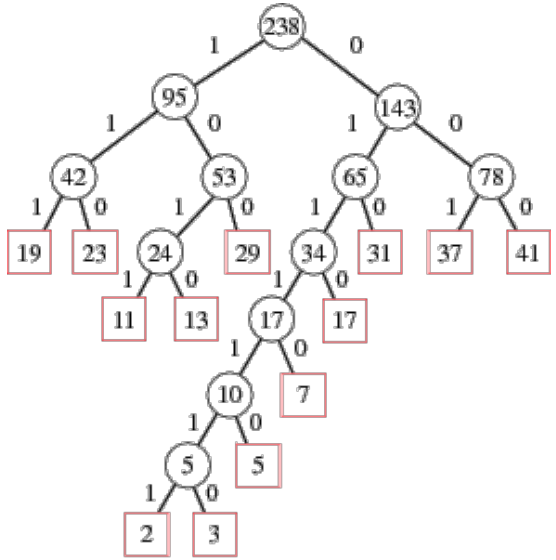
Array of unpredictable data (FP)

Data are ordered in the arrays in their initial order

SZ Coding Stage

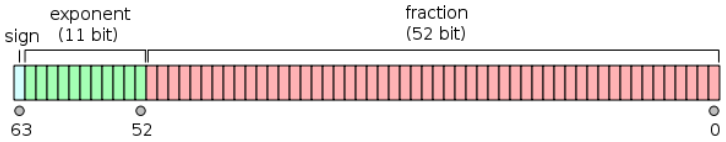
3) **Variable length coding** (Huffman)

We built the Huffman tree using a symbol size corresponding to the number of bits needed to code the bin numbers



➡ Reduce the number of bits needed to represent values

4) Unpredictable data analysis



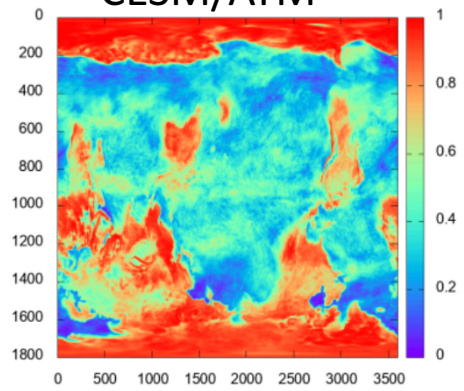
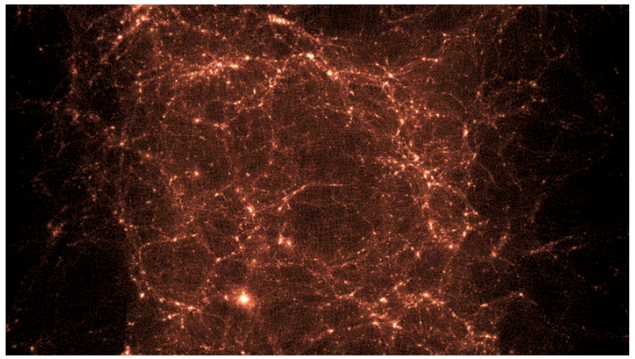
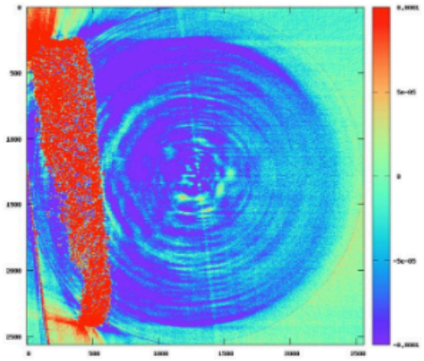
➡ Reduce the number of bits needed to represent unpredictable data (<1%)

5) Optional GZIP (L77 + Huffman coding): improve the compression by about 10%, but also slowdown
Both on result of the Huffman coding and the array of unpredictable data

Lossy Compression Performance

Argonne SZ 1.4 lossy compressor

All these apps need at least a factor of 10 compression

Mesh data	Particle data	Instrument data
CESM/ATM 	HACC 	APS 
Compression performance	Compression performance	Compression performance
Compression rate: 111 MB/s	Compression rate: 62 MB/s	Compression rate: 110 MB/s
Decompression rate: 202 MB/s	Decompression rate: 182 MB/s	Decompression rate: 152 MB/s
Compression factor: **14.5**	Compression factor: **10.4**	Compression factor: **13.5**
Point Wise Error Bound: 10^{-3}	Point Wise Error Bound: 10^{-3}	Point Wise Error Bound: 10^{-2}
GZIP compression factor: 1.3	GZIP compression factor: 1.2	GZIP compression factor: 1.1

S. D., F. Cappello, Optimization of Error-Bounded Lossy Compression for Hard-to-Compress HPC Data, **IEEE TPDS,** to appear.

Surprisingly Good Compression Performance

Quantum Chemistry

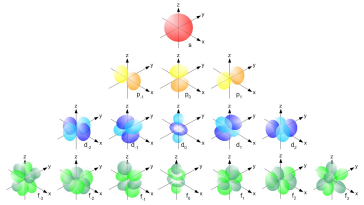
All properties of the quantum mechanical system are determined by the wave functions which are obtained by solving the Schrödinger Equation.

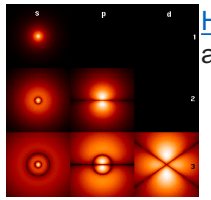
Computing two-electron repulsion integrals is the most time-consuming step in solving Schrödinger Equation.

Typically, integrals cannot fit into memory and iterative solutions requires recalculating integrals every iteration.

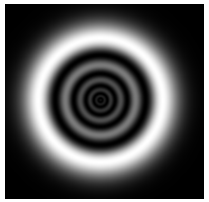
Much better strategy: compute election repulsion integrals once, compress and store them in memory, and read ERIs from memory every time we need them.

→ But a factor of 10 compression is needed (at least)
→ Designed a new predictor using pattern matching

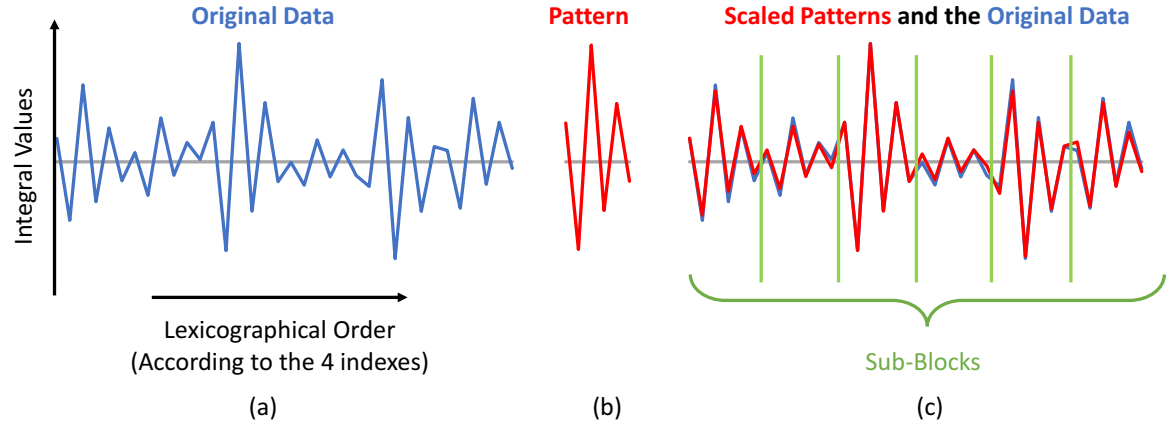


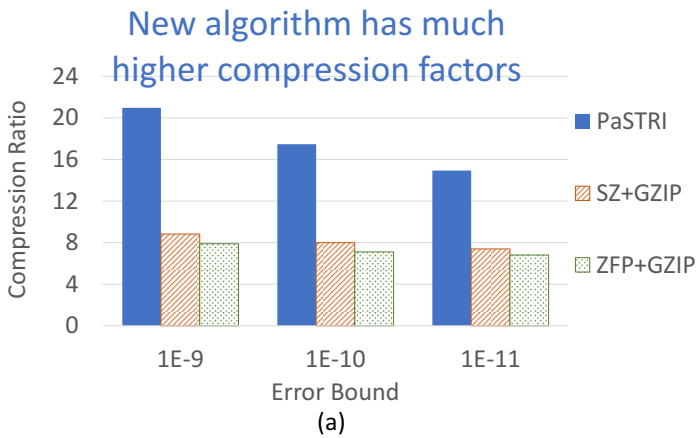


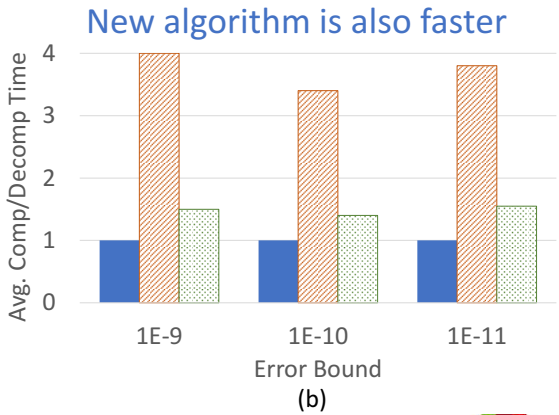
Hydrogen-like atomic orbitals



Computed hydrogen atom orbital for the 6s orbital







EXASCALE COMPUTING PROJECT

A. Murat Gok, D. Tao, S. Di, V. Mironov, Y. Alexeev, F. Cappello, PaSTRI: A Novel Data Compression Algorithm for Two-Electron Integrals in Quantum Chemistry, **IEEE/ACM SC17**

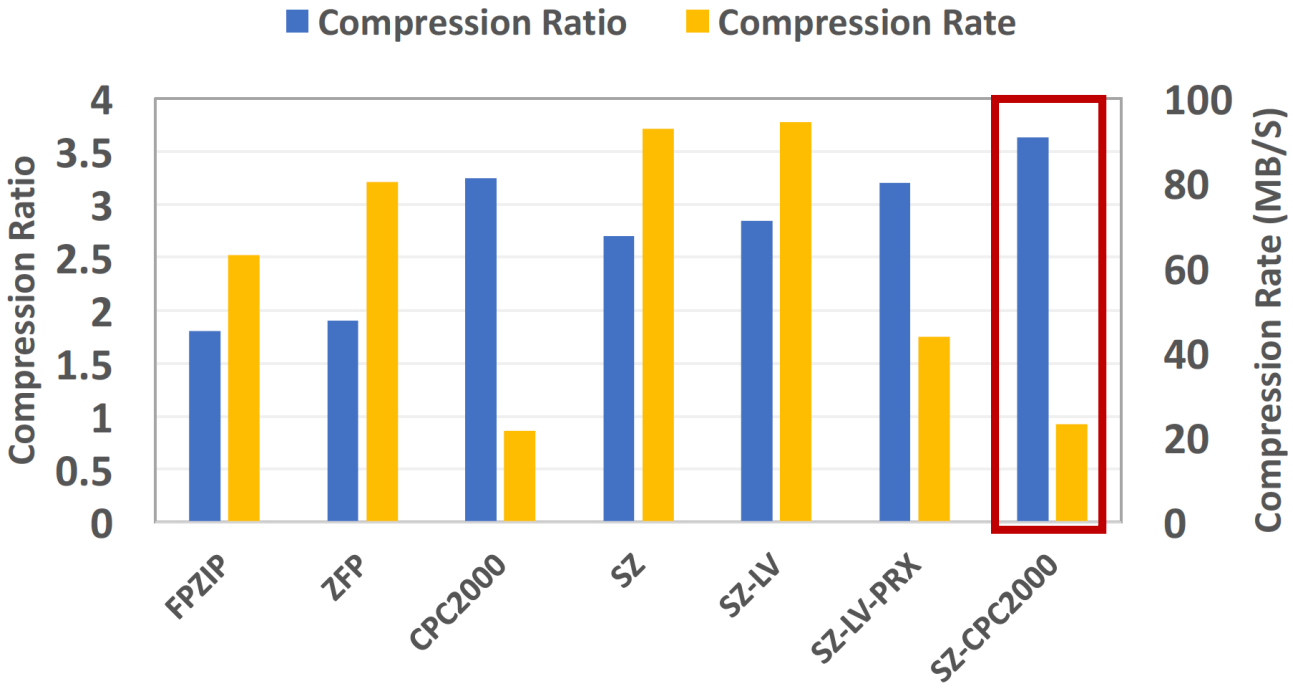
Argonne NATIONAL LABORATORY

Some datasets are very hard to compress e.g. Particle datasets

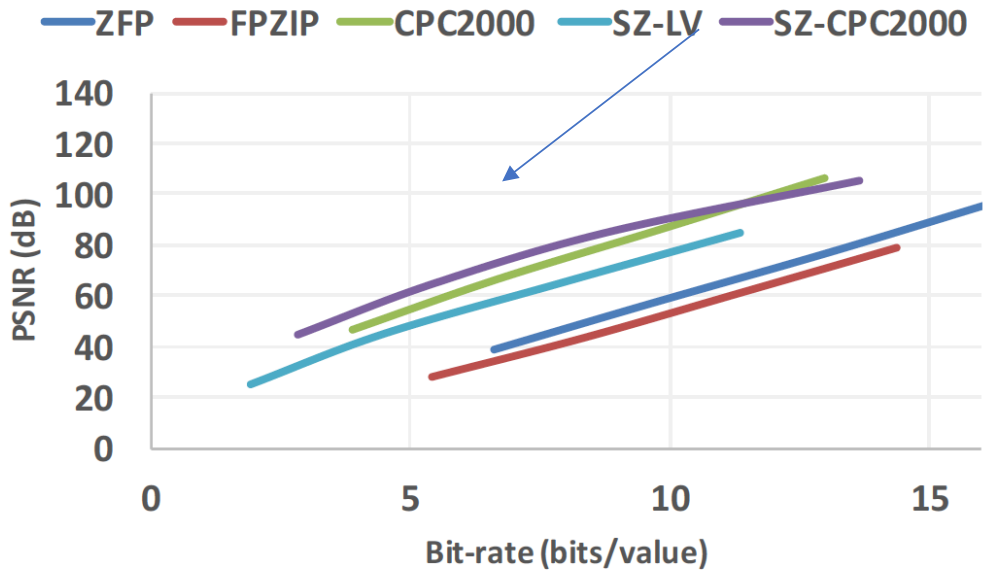
AMDF Molecular Dynamics code (Accelerated Molecular Dynamics Family)

A compression factor of 10 is needed

Compression ratio/rate (AMDF), REB: 10^{-4}



Rate distortion



(b) AMDF

Similar result with HACC (cosmology)

→ We need to find a new way to compress these particles datasets

D. Tao, S. Di, Z. Chen, and F. Cappello, In-Depth Exploration of Single-Snapshot Lossy Compression Techniques for N-Body Simulations, Submitted

Some datasets are much easier to compress

In some situation, SZ 1.4 compression factor can be >100.
And the difference is barely noticeable.

E.G. Miranda dataset*

Miranda is a radiation hydrodynamics code designed for large-eddy simulation of multicomponent flows with turbulent mixing.

Example of Rayleigh-Taylor instability simulation involving two fluids of different density.

→ Compression factor of 110 with acceptable distortions

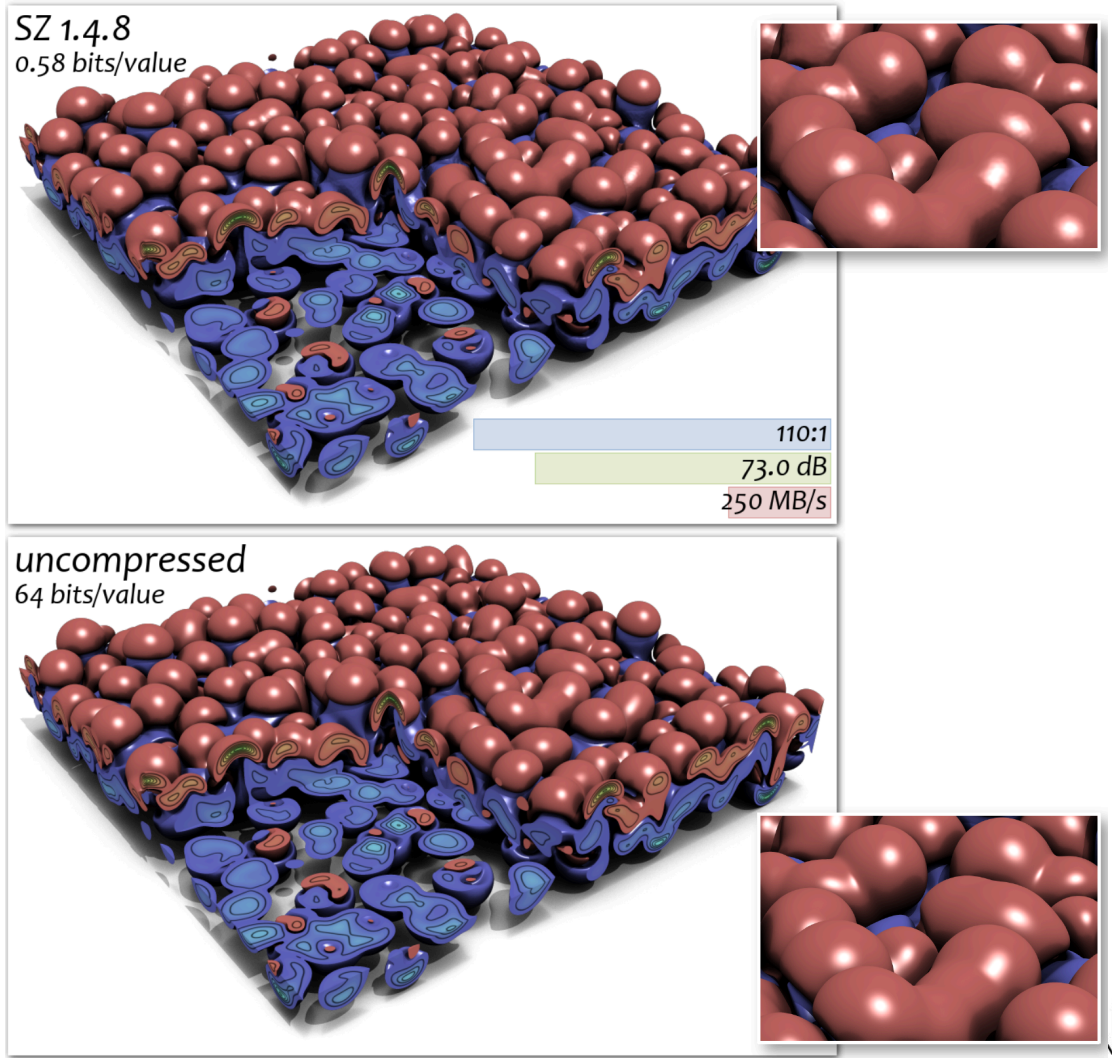


*Figure from Peter Lindstrom, LLNL


Can we Aggressively
Reduce Scientific Data
Without Losing Science?
Image from HACC Cosmology Simulation

How Lossy Compression Distorts Scientific Data?

- Lossy compressors using different algorithms are likely to distort the dataset differently

- Very few studies (mostly on climate datasets) analyze compression quality

- We need to compare the different compressors scientifically (fairly)
- We need to share common metrics quantifying the lossy compression error

→ We need a methodology (metrics and tools) to assess the nature of the error introduced by lossy compressors.

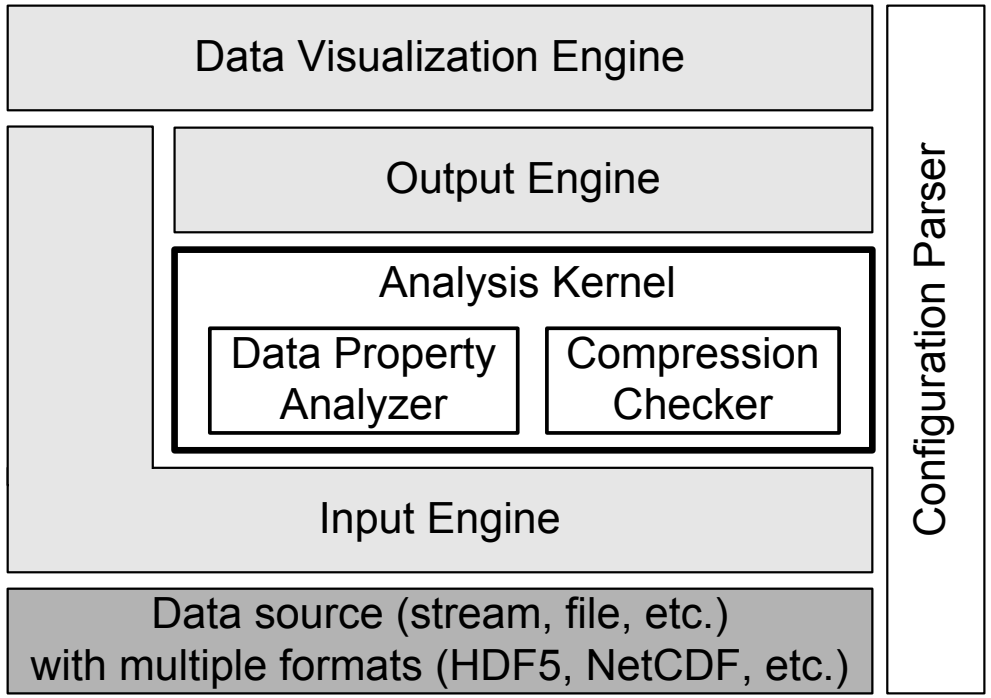
ANL Z-checker (ECP CODAR)

Analysis Kernels:

1. *Initial Data Property Analyzer*
2. *Compression Error Analyzer*

Incremental design:

- Metrics and interface defined with users
  Integrates analysis functions in C and R



2 modes:

- With SZ and ZFP integrated (1 input file), compression is run several times for RD
- 2 input files (does not provide rate-distortion analysis)

Currently accepts only data sources in binary format (big/little endian), 32 or 64 bits.

Z-Checker is supported by the DOE/NNSA **ECP CODAR** Project.

I. Foster, M. Ainsworth, B. Allen, J. Bessac, F. Cappello et al. Computing Just What You Need: Online Data Analysis and Reduction at Extreme Scales, **Europar 2017**

EXASCALE COMPUTING PROJECT

Argonne NATIONAL LABORATORY

Z-checker Example: Climate and Severe Weather

- Experimental data (single-floating point)
  - Climate: ATM: 2D datasets from climate/atmosphere simulations (pseudo 3D)
  - Weather: hurricane: 3D datasets from Hurricane Isabel simulation

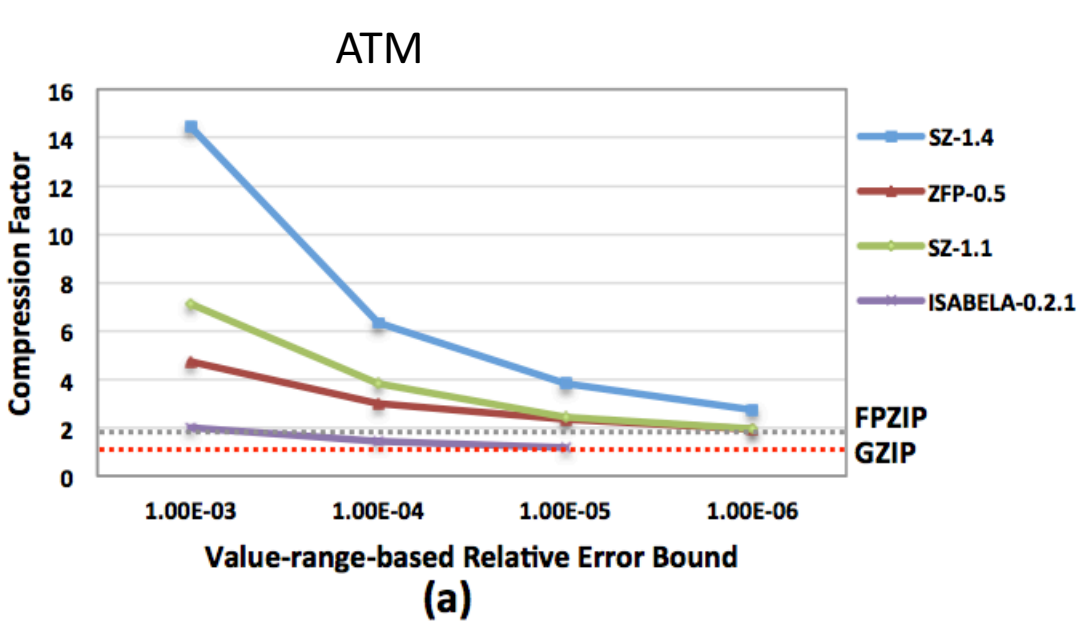
	Data Source	Dimension Size	Data Size	File Number
ATM	Climate simulation	1800 × 3600	2.6 TB	11400
Hurricane	Hurricane simulation	100 × 500 × 500	1.2 GB	624

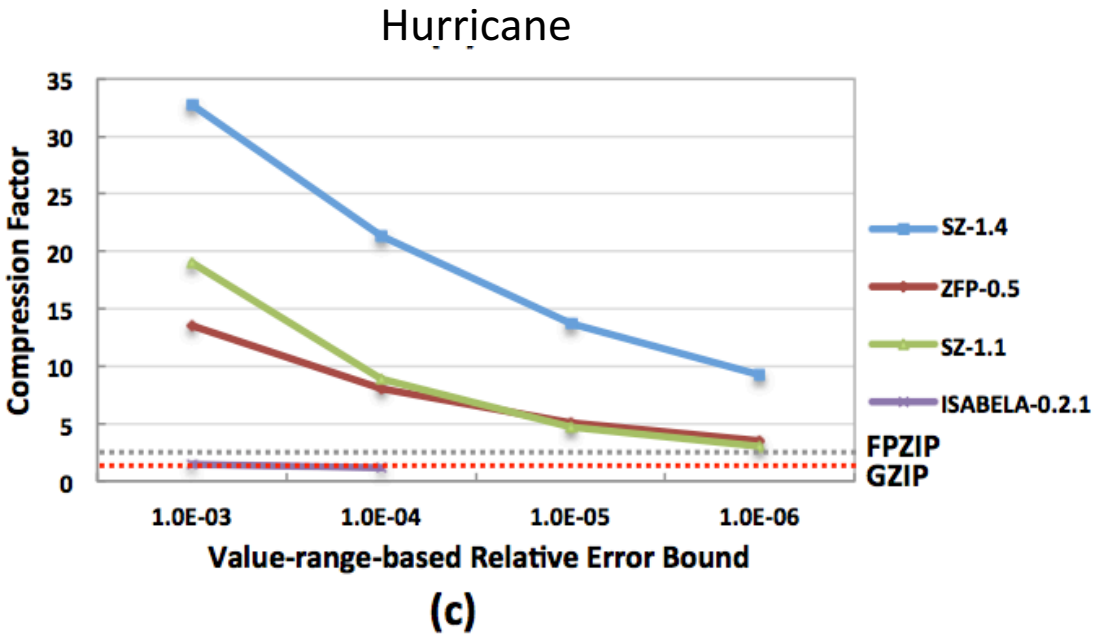
Notes: Tested many different datasets:

- Particle datasets (Cosmology, Molecular Dynamics)
- Quantum Chemistry
- Datasets from Instrument (Argonne APS, Berkeley Linac Coherent Light Source)

→ The following slides apply also to these datasets

Z-checker: Compression ratios (factors)





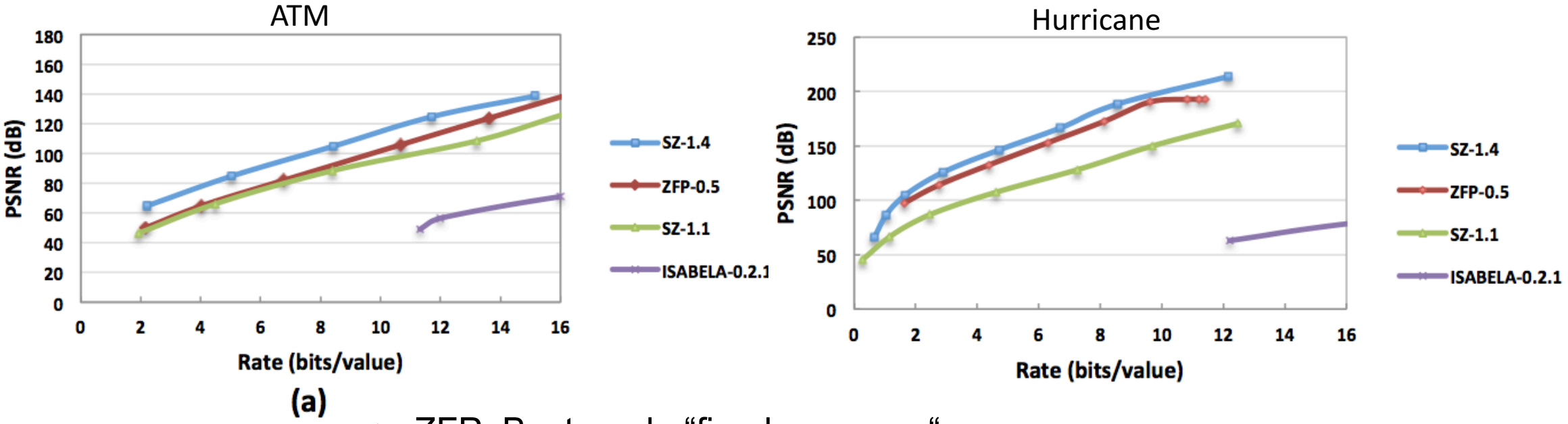
- Value-range-based (VRB) relative error bound:
  (ratio of absolute error bound to data value range)

- E.g., VRB relative error bound = 1E-3
  - ATM: SZ-1.4: CF=14 → **2x** of ZFP
  - Hurricane: SZ-1.4: CF=33 → **2.5x** of ZFP

Z-checker: Rate Distortion

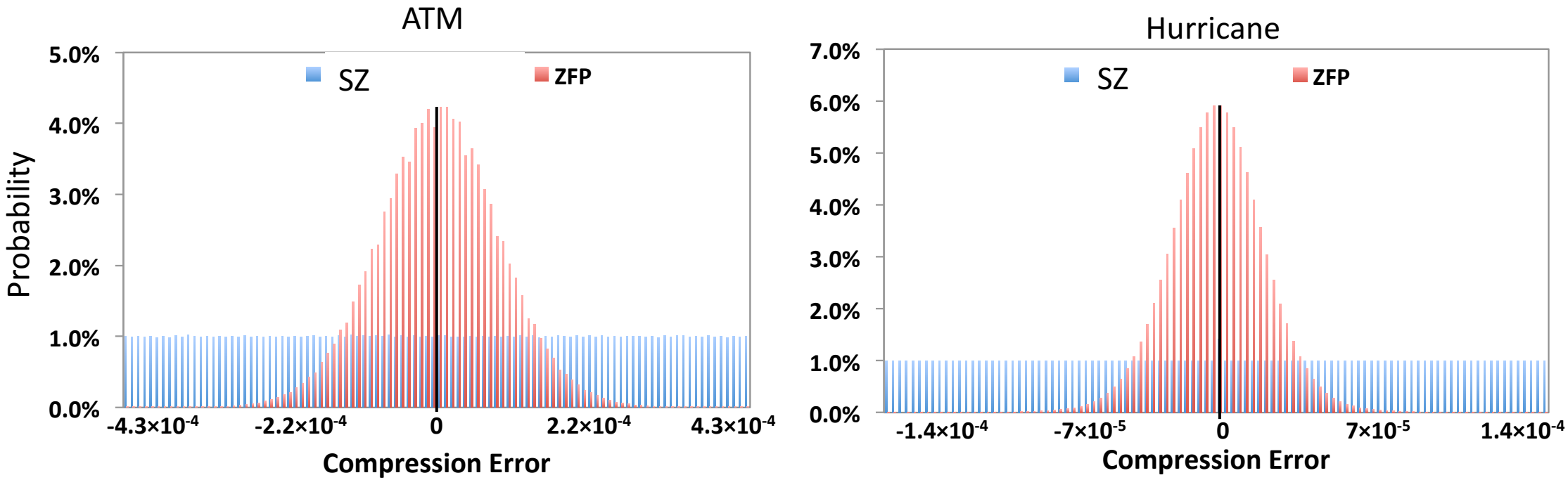
$$R_X = x_{max} - x_{min}$$

$$psnr = 20 \cdot log_{10}(\frac{R_X}{rmse}) \qquad rmse = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(e_{abs_i})^2}$$



- ZFP: Best mode "fixed-accuracy"
- E.g., bit-rate = 8 bits/value (CR = 4)
  - SZ-1.4: 14 dB higher than ZFP on ATM
  - SZ-1.4: 11 dB higher than ZFP on Hurricane
- PSNR is logarithmic scale
- PSNR → RMSE: 14 dB ~ 5x, 11 dB ~ 3.5x

Z-checker: Distribution of Compression Error



SZ: Uniform

ZFP: Gaussian

Distribution plot informs about the nature of the noise that the lossy compressor adds to the dataset. It also shows how the errors are distributed around 0. SZ and ZFP provide symmetric errors: Good. We need to know what type of distribution is better for users: Uniform or Gaussian

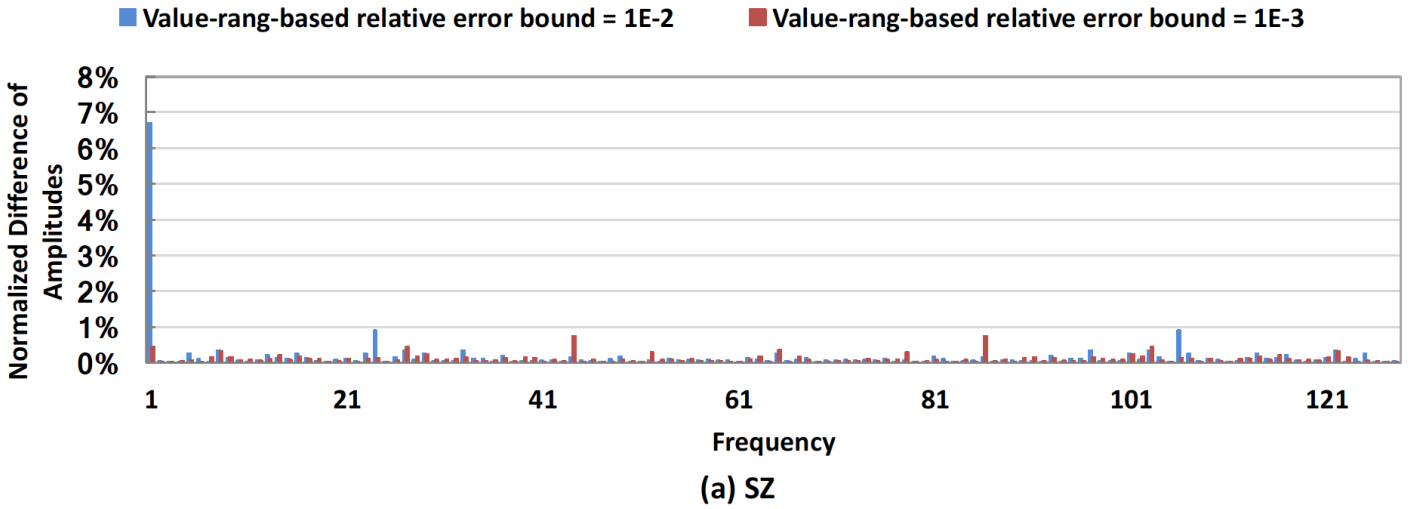
Z-checker: Spectral Alteration

We want to know how the compression affects the different components of the signal (frequency domain)

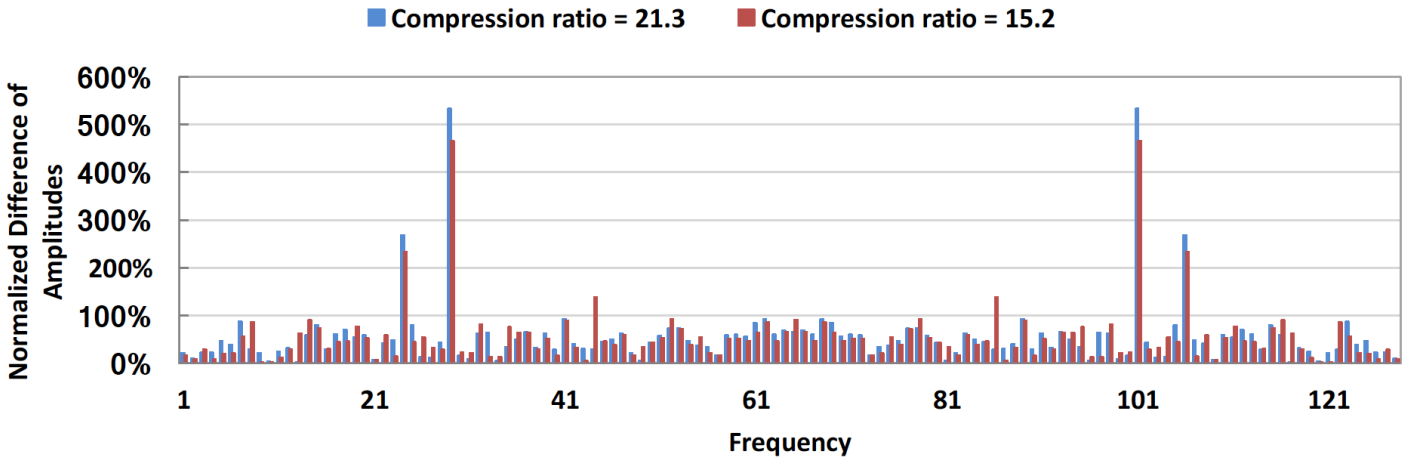
Compute the Fourier Spectrums for
- the initial dataset
- the decompressed dataset

Plot the amplitude difference for each frequency (normalized from the initial dataset)

In general, we want to preserve as much as possible the spectral properties of the signal



(a) SZ



(b) JPEG2000

EXASCALE COMPUTING PROJECT

Z-checker: Pearson Correlation of Initial and Decompressed Datasets

Considering a dataset $\{x_1,\ldots,x_n\}$ containing n values and another dataset $\{y_1,\ldots,y_n\}$ containing n values

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

covariance

Std var x,y

where:

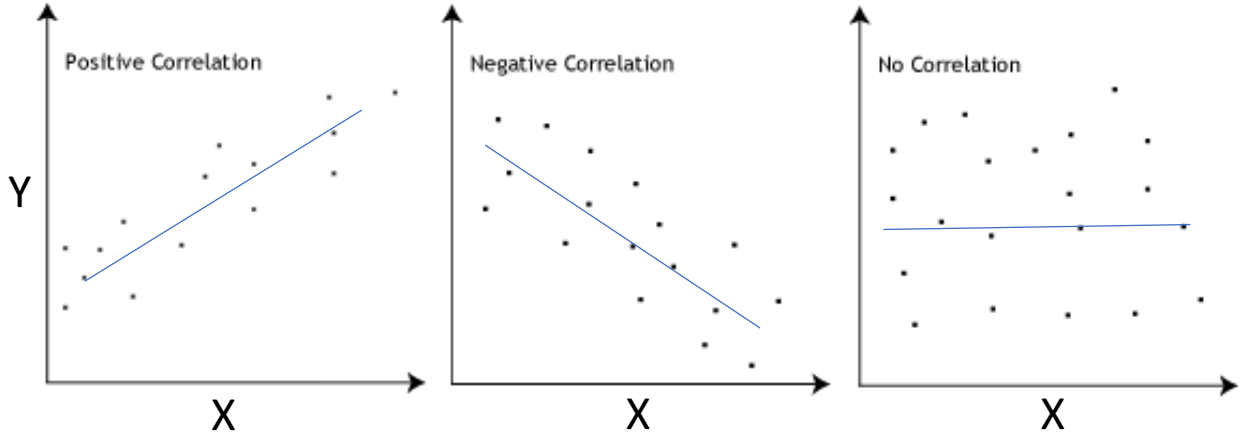
- n, x_i, y_i are defined as above
- $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ (the sample mean); and analogously for $\bar{y}$

Pearson correlation coefficient measures the linear correlation between two variables *X* and *Y*.

r = +1≤ Value ≤ −1,

- 1: total positive linear correlation,
- 0: no linear correlation,
- −1: total negative linear correlation

If X is the initial dataset and Y the reconstructed dataset,
We sort the values in X and use X ordering for Y to plot the graph



We want a maximum correlation (~1) between the initial and decompressed datasets

Maximum e_{rel}	ATM		
	SZ-1.4	*ZFP*	*SZ-1.1*
3.3×10^{-3}	0.99998	0.9996	0.99998
4.3×10^{-4}	$\geq 1-10^{-6}$	$\geq 1-10^{-7}$	$\geq 1-10^{-6}$
2.6×10^{-5}	$\geq 1-10^{-8}$	$\geq 1-10^{-9}$	$\geq 1-10^{-9}$
3.4×10^{-6}	$\geq 1-10^{-10}$	$\geq 1-10^{-11}$	$\geq 1-10^{-11}$
4.1×10^{-7}	$\geq 1-10^{-12}$	$\geq 1-10^{-13}$	$\geq 1-10^{-13}$

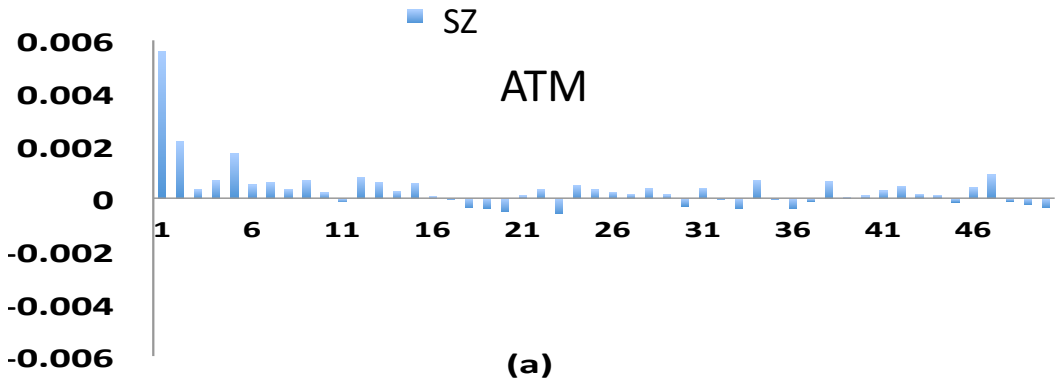
Maximum e_{rel}	Hurricane		
	SZ-1.4	*ZFP*	*SZ-1.1*
2.4×10^{-3}	0.998	0.99995	0.998
1.8×10^{-4}	$\geq 1-10^{-5}$	$\geq 1-10^{-6}$	$\geq 1-10^{-5}$
2.5×10^{-5}	$\geq 1-10^{-6}$	$\geq 1-10^{-8}$	$\geq 1-10^{-5}$
2.6×10^{-6}	$\geq 1-10^{-8}$	$\geq 1-10^{-9}$	$\geq 1-10^{-7}$
2.9×10^{-7}	$\geq 1-10^{-10}$	$\geq 1-10^{-11}$	$\geq 1-10^{-11}$

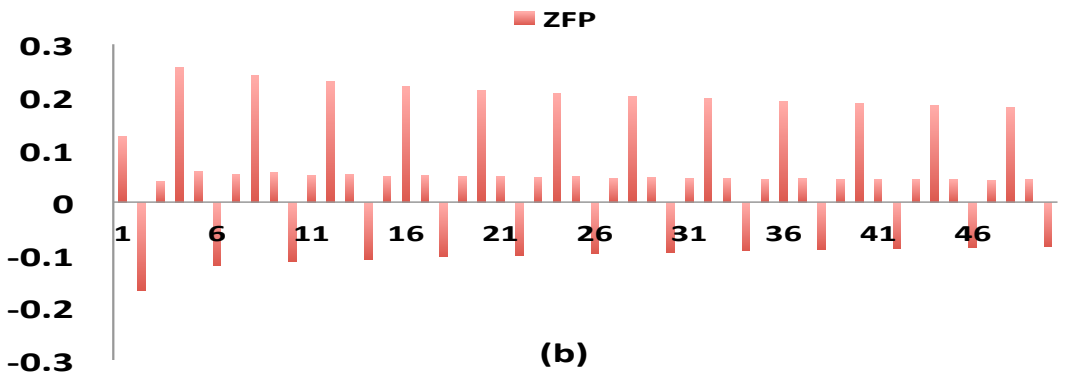
Z-checker: Autocorrelation of the Compression Error

Compression introduces autocorrelation that was not there before

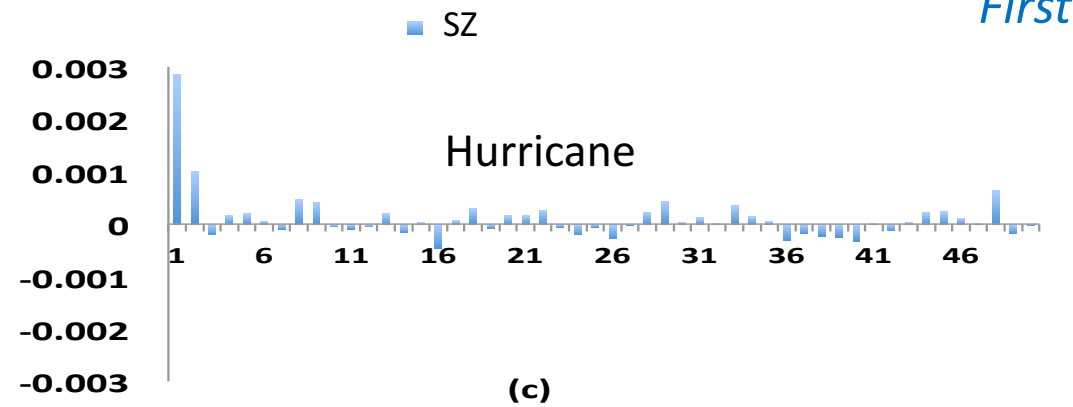
We want as little (0) autocorrelation of the compression error as possible

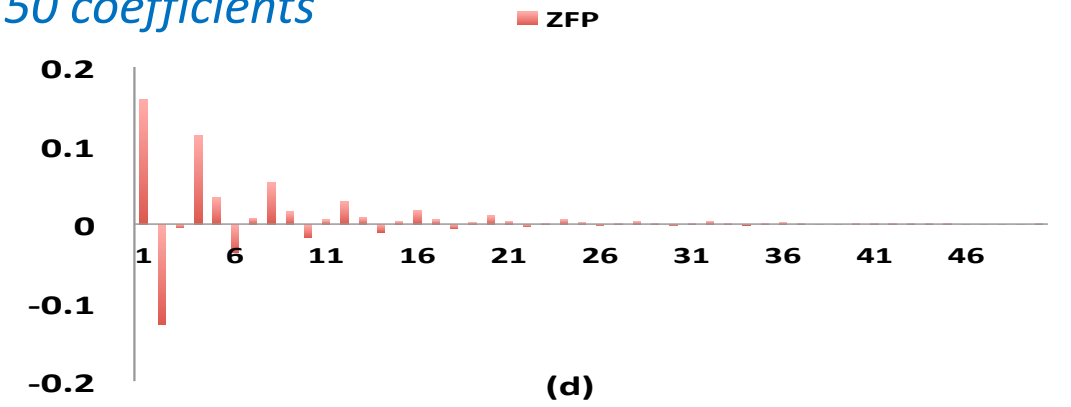
$$R(\tau) = \frac{\mathrm{E}[(X_t - \mu)(X_{t+\tau} - \mu)]}{\sigma^2}$$





First 50 coefficients





SZ introduce a much lower and shorter autocorrelation compared to ZFP

Z-checker: Preservation of Derivatives

Derivatives (original versus decompressed) for the ATM dataset using SZ 1.4

Compute the derivatives on each point of the original and Decompressed dataset.

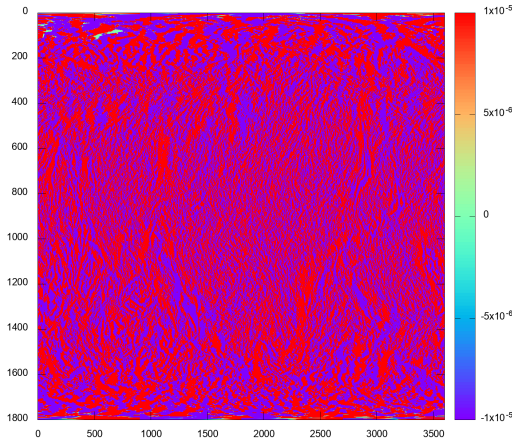
Plot the result.

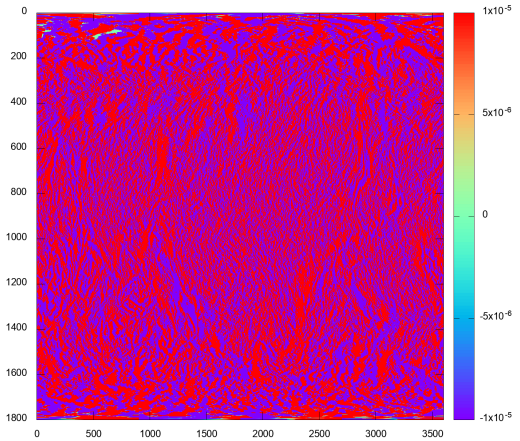
Note that we could have plotted the difference instead of the two figures

→ SZ does not visually alter the first and second derivatives

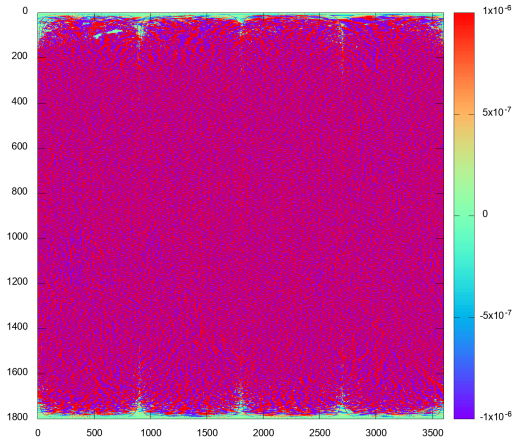
Original

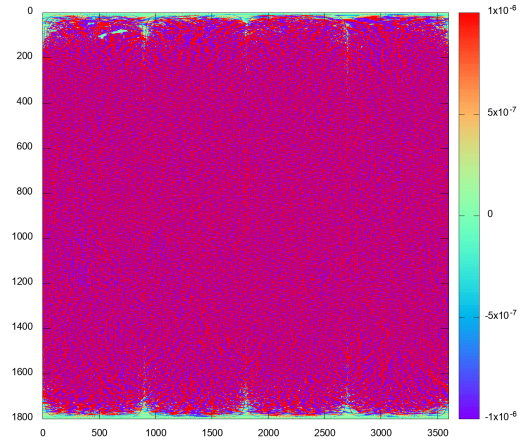
Decompressed





First derivatives





Second derivatives

OK, So What?
Lossy Compression vs. Decimation

Image from Halo finder in Cosmology Simulation

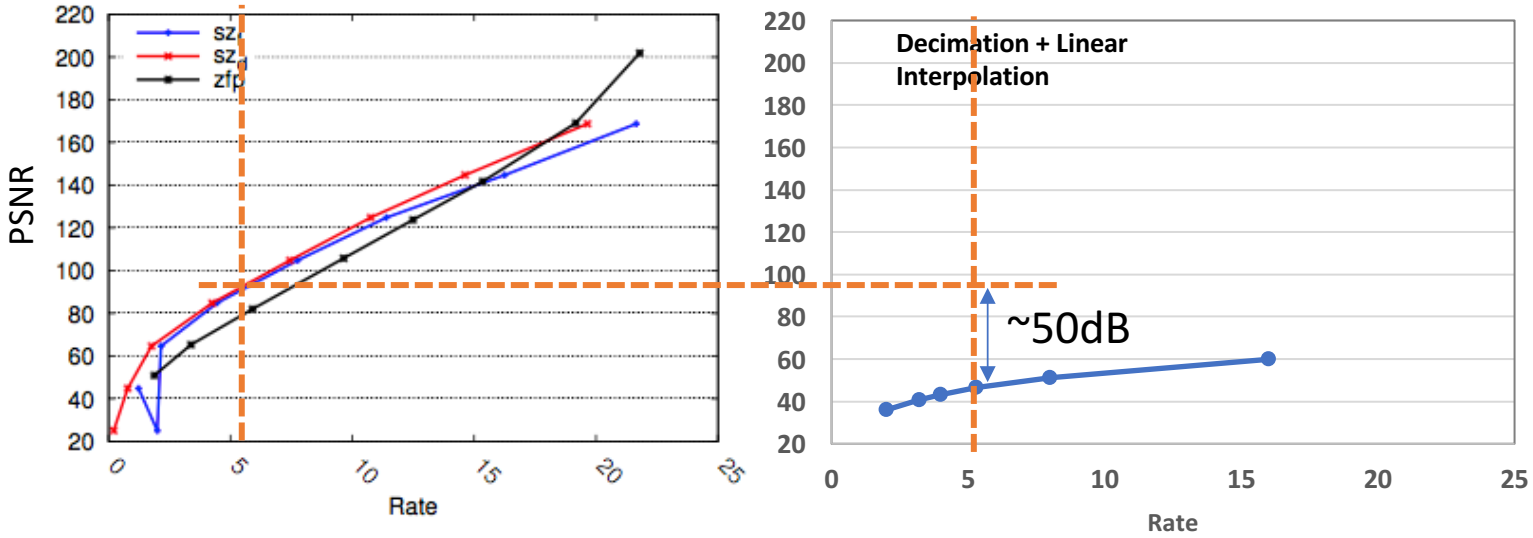
Decimation vs. Compression (Z-checker analysis)

Lossless Spatial Decimation + Linear Interpolation → The most common reduction technique
Lossy compression with SZ 1.4

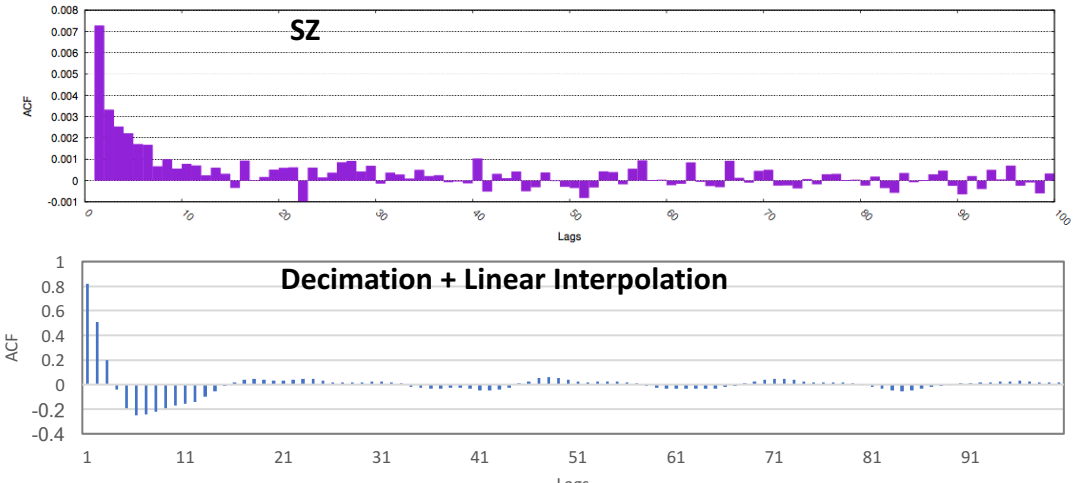
ATM dataset: ATM CLDLOW
Decimation/compression factor: 6

→ Much better rate-distortion for SZ
→ Much lower autocorrelation for SZ
→ Much lower spectral alteration for SZ
→ **Compression outperforms decimation**

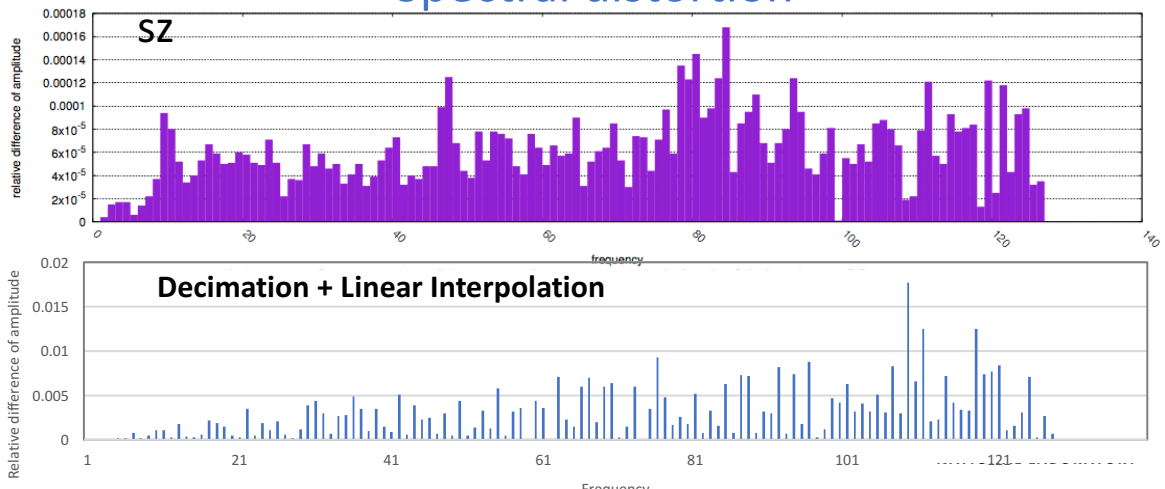
Rate distortion



Autocorrelation of the reduction error



Spectral distortion



Links:

SZ: https://github.com/disheng222/SZ

Z-checker: https://github.com/CODARcode/Z-checker

SC17 Tutorial on lossy compression for scientific data:
http://sc17.supercomputing.org/presentation/?id=tut106&sess=sess205

Soon: a repository with reference datasets, compressors, configurations, results

Conclusion

- **Aggressive data reduction is needed** and will be critical to advance science
- **All IT domains use lossy compression** when data becomes too large
- **Lossy compressors are available** for scientific data (simulation and instruments)
- They produce **excellent results:** performance, speed, distortions
- The current popular technique of **decimation is far inferior to lossy compression**

- As we progress in the use of lossy compression in scientific application, we should **continue developing the evaluation methodology** (metrics, tools, benchmarks, etc.)

By the way, compression is also an art