

Use-cases of lossy compression for floating-point data

Franck Cappello

Argonne
National Laboratory

- Why do we need lossy compression for scientific data?
- How lossy compressor works for scientific data?
- What is user primary request?
- **8 Use-cases**
- What's next?

F. Cappello, S. Di, S. Li,² X. Liang, A. Murat Gok, D. Tao, X.-C. Wu,⁶ Y. Alexeev, F. T. Chong,
Use-cases of lossy compression for floating-point data in scientific datasets,
IJHPCA, to be published, 2019

When the Scientific Data Becomes Too Big

- Today's scientific simulations produce extremely large of datasets, often too large to save, process and analyze

- **Cosmology Simulation (Space&Performance argument):**

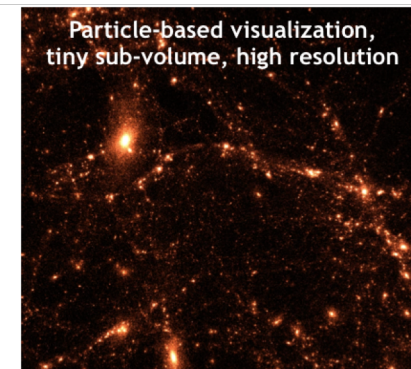
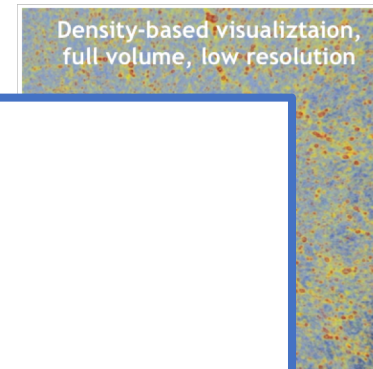
- A total of **>20PB** of data when simulating trillion of particles (500-1K snapshots)

- Petascale (you will r

- On current 20 X 10¹

→ To solve these problems,
We need significant data reduction to:

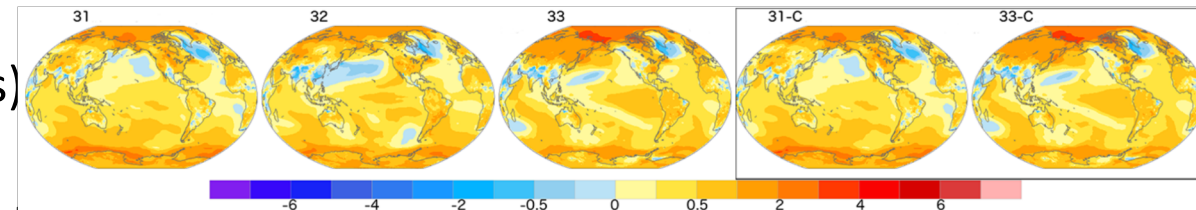
- Reduce storage size
- Reduce I/O overhead
- Reduce relative cost of storage in systems



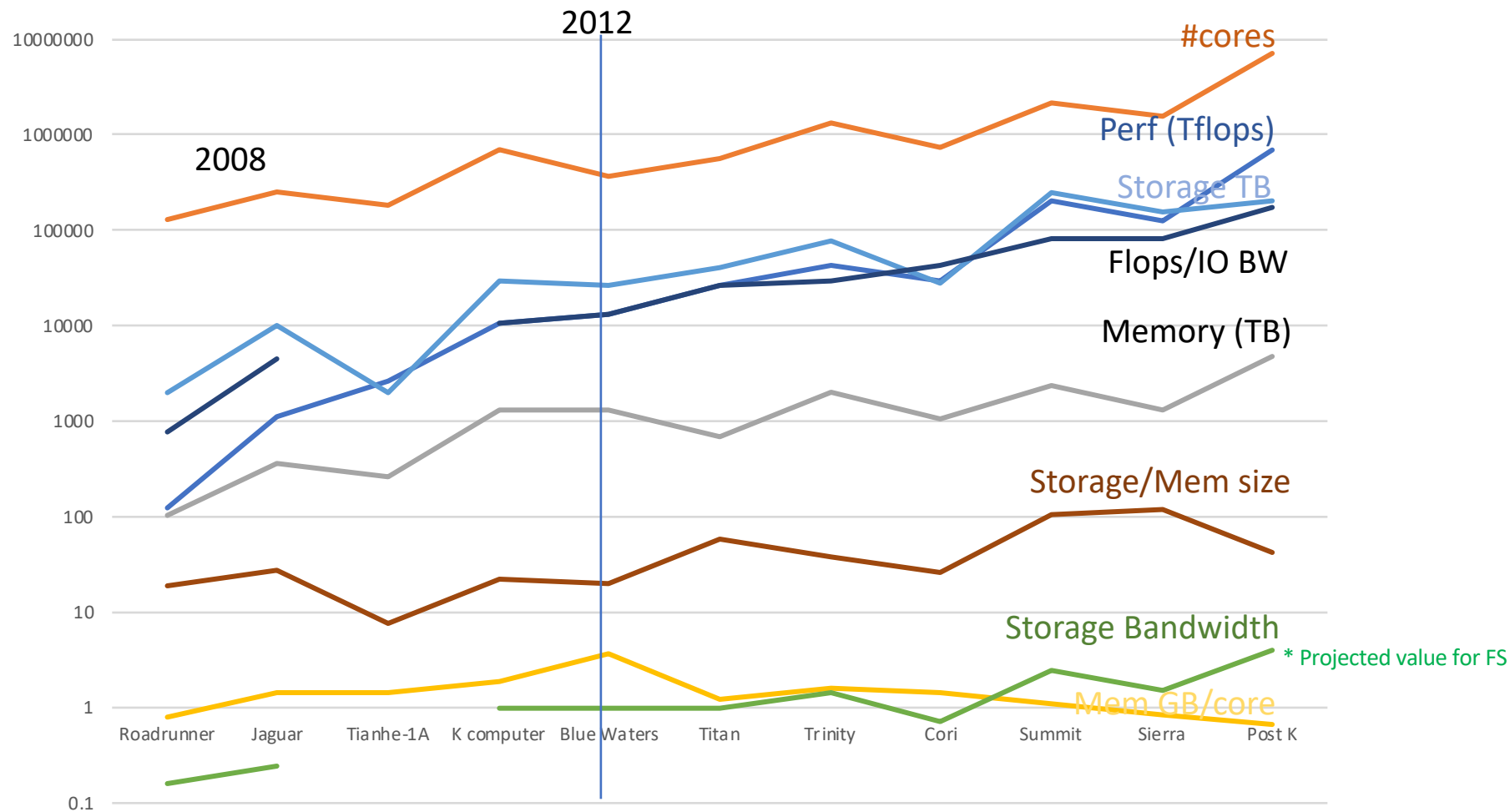
- **Climate Simulation (Cost argument):**

- IPCC Coupled Model Comparison Projects (CMIPs)
- The relative cost of storage is increasing...

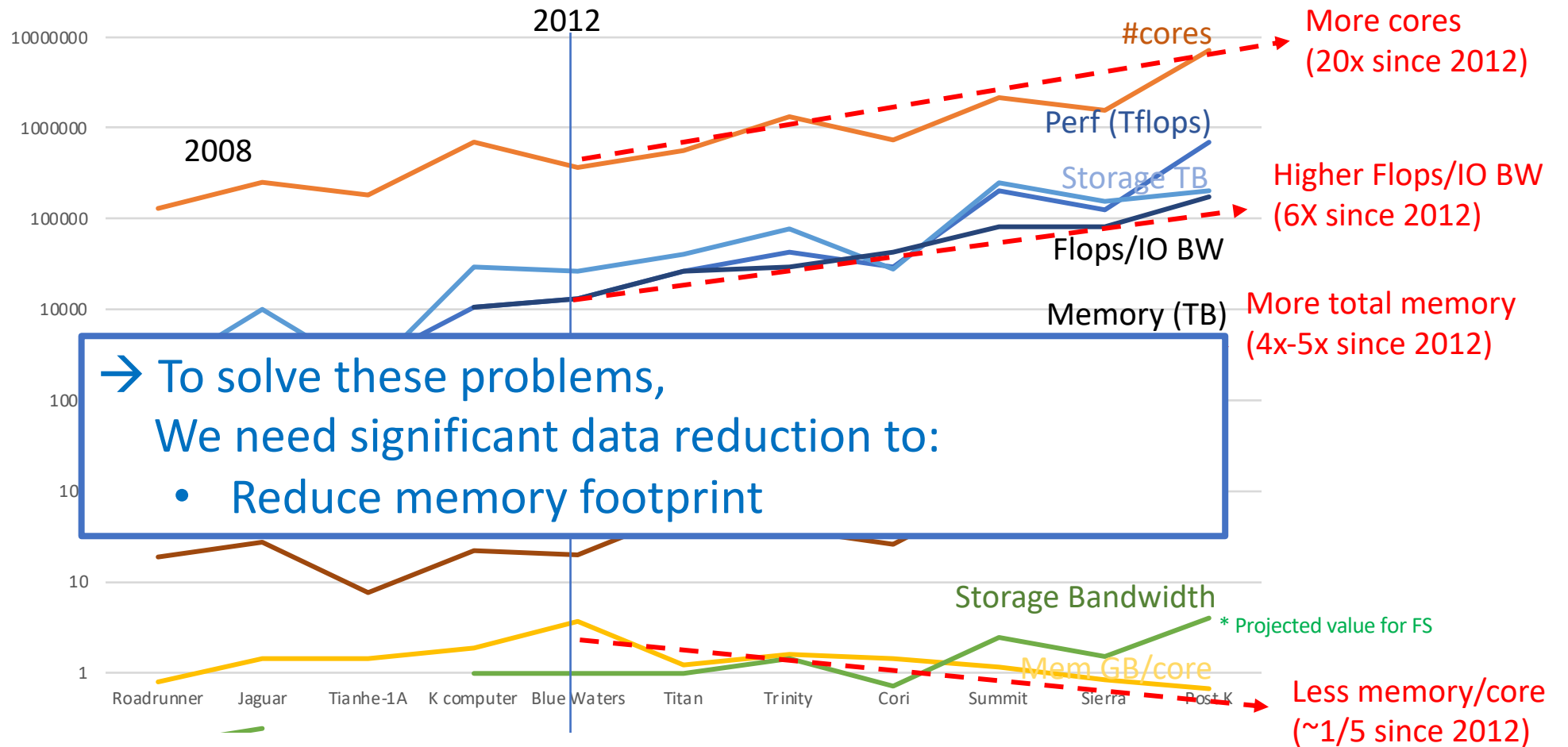
- Previous NCAR platform (2013): ~20% of hardware budget
- **Current NCAR platform (2017): ~50% of hardware budget**



Architectural trends worsen the situation



Architectural trends worsen the situation



- If we can exploit all the memory → **run larger problems that will generate more I/Os**
- +Higher flops/IO BW → **need to compute more for each I/O byte** (x6 in 7 years)
- **Less memory per core** makes weak scaling more difficult WRT previous generations

- If reducing reduce the Flops/communication byte ratio possible
- → unable to exploit the full scale because required problem size will not fit in memory

Ok we need to reduce the data, but how?

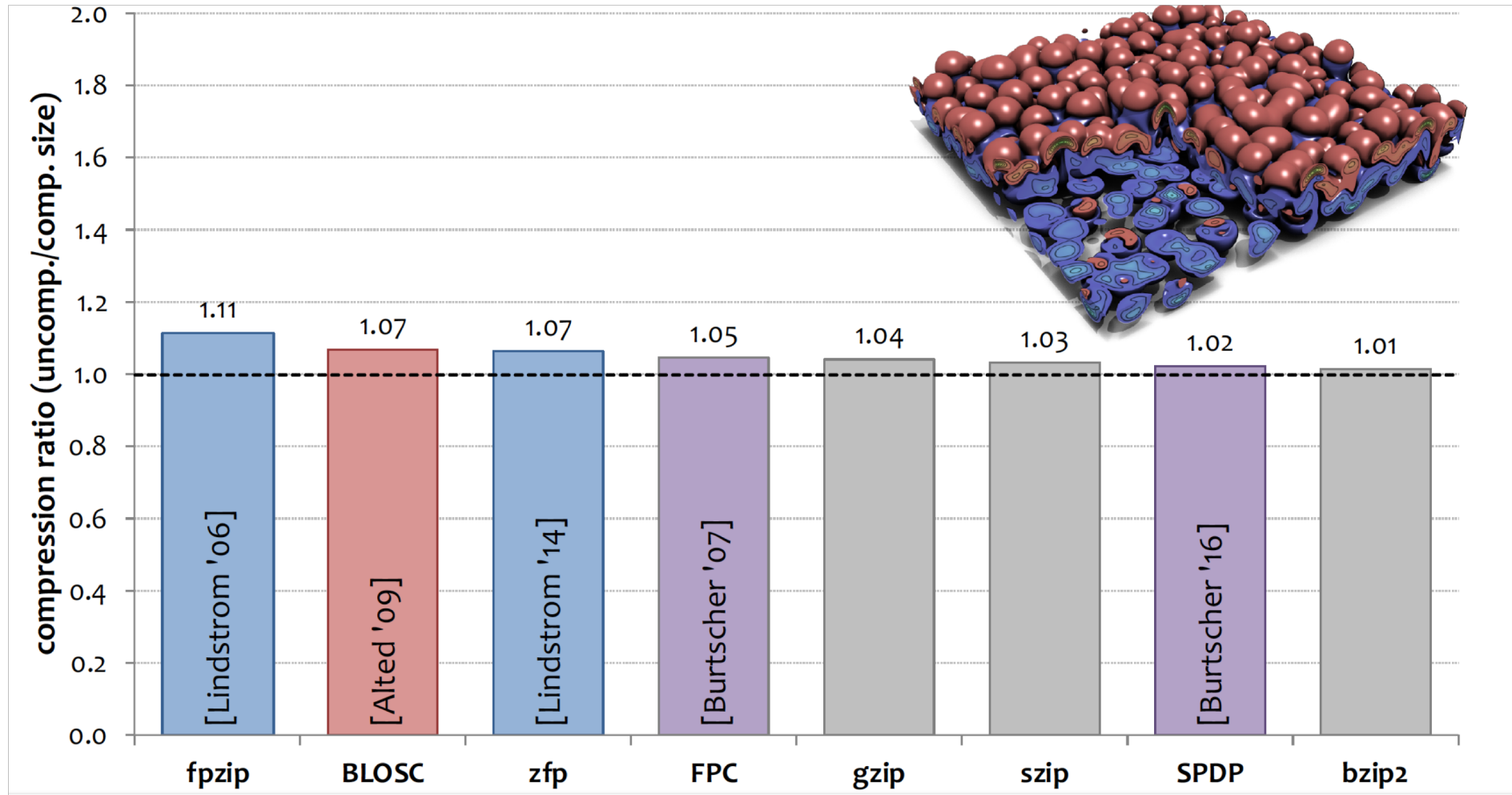
- **Lossless compression:**

- 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)
- Popular example: Gzip

- **Lossy reduction:**

- **The data is altered** during reduction: some piece of information is removed (lost): the original data cannot be retrieved.
- **Noise is added** to the original data. Knowing the nature of the noise is critical for users of the reduced data.
- If user can control the accuracy of the reduced data, then **lossy reduction is a trade-off between reduction ratio and loss of accuracy**
- Popular example: JPEG

Why lossless compression does not help?



What about decimation in time (lossy reduction)?

N body simulation (large scale phenomena: cosmology, small scale phenomena: MD)

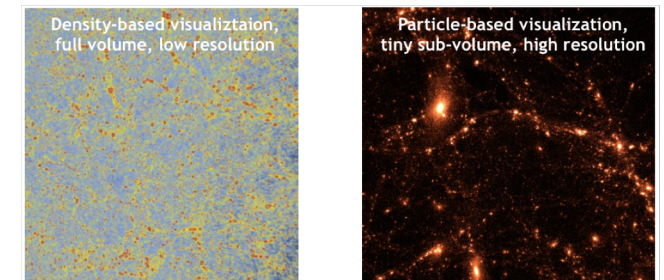
Cosmology: HACC code simulates the formation of the universe (halo, galaxies, etc.)

Molecular Dynamic: EXAALT code simulates behavior of atoms in a nano-crystalline sample of copper under the influence of a strong electric field.

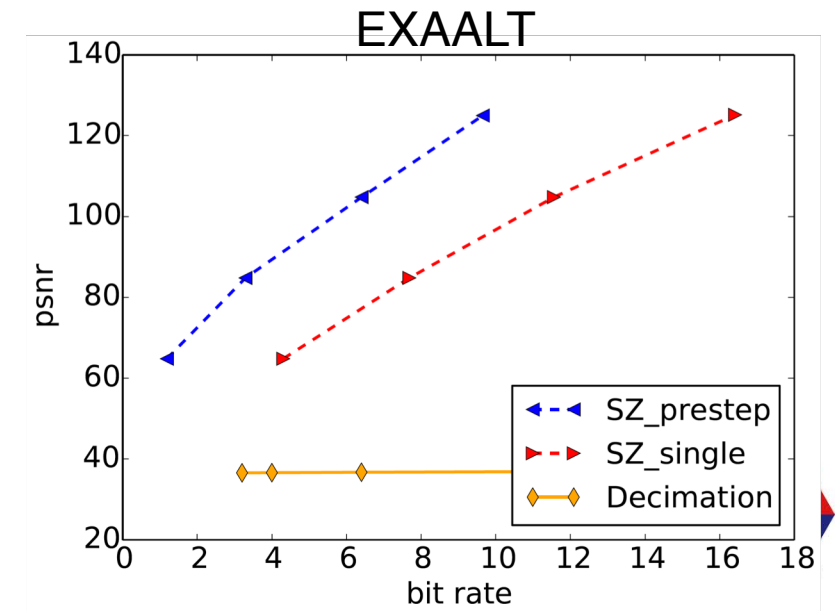
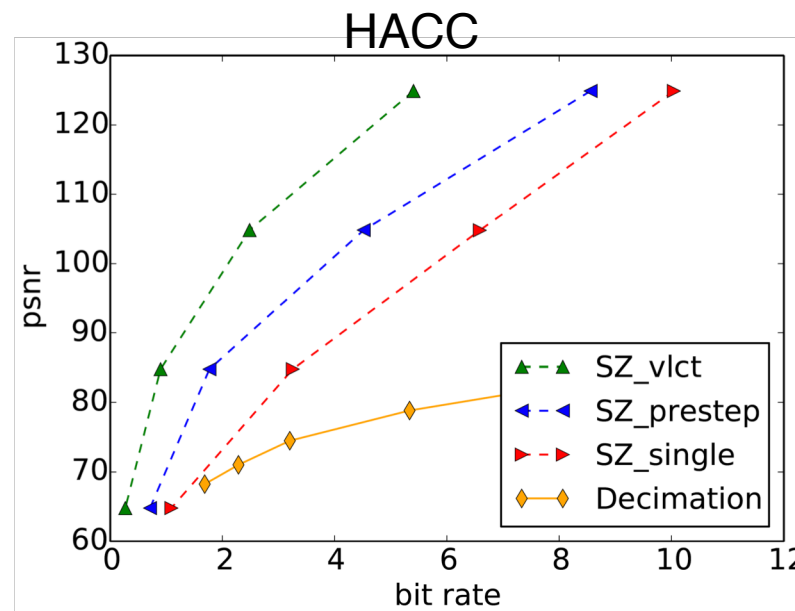
Temporal decimation (1/2, 1/5, 1/10, etc.) + linear interpolation

Lossy Compression with **SZ** (spatial only and temporal)

	Snapshots	Variables	Particles (million)
HACC	100	x, y, z, v_x, v_y, v_z	15-20
EXAALT	83	x, y, z	1



Rate distortion of HACC and EXAALT data on variable x

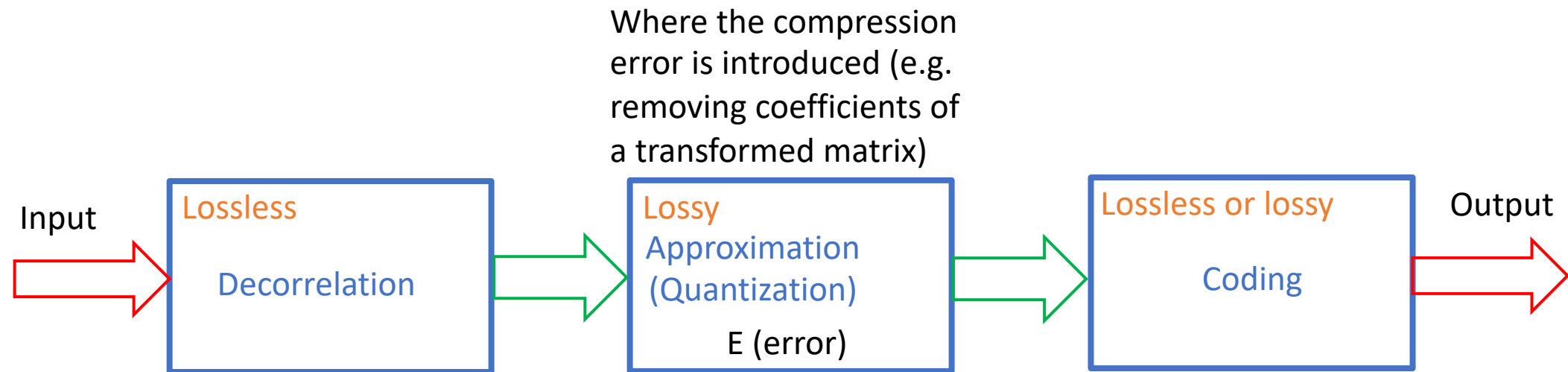


Why this is not already solved?

- Problem did not really exist 5-10 years ago
 - No investment from application and CS people to develop lossy compressors for scientific data (no research on impact of lossy compression error)
- Lossy compression of scientific data is a hard problem
 - The mantissas of floating point data in scientific datasets are fairly random
 - Scientific users are worried about the error/noise introduced by lossy compression
 - >40 years old (LZ77), >70 years (Shannon's information theory - 1948)

How modern lossy compressors look like?

- Three main stages (each stage may use multiple sub-stages)
+ potentially additional preconditioning of the data + post processing



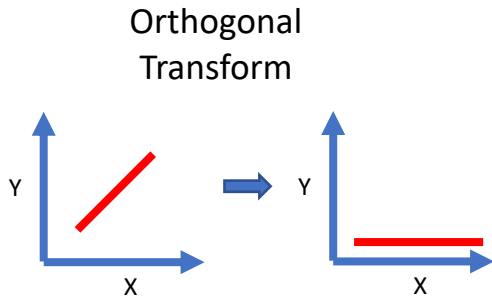
Convert the dataset into another less correlated one (exploit autocorrelation).

e.g. concentrate signal energy on less data

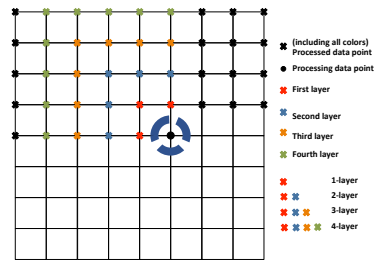
Use the minimum number of bits to represent a string of symbols

Where the research is in lossy compressors for scientific data?

Decorrelation stage that leverages redundancy:
Similarities, Autocorrelation, Smoothness of the data in the dataset

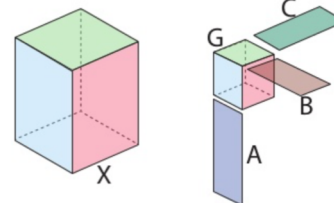


Prediction



$$\chi_{(i^0, j^0)} = \sum_{(i^1, j^1) \in (i^0, j^0)}^{0 \leq i^1 \leq i^0, 0 \leq j^1 \leq j^0} (-1)^{i^1 + j^1} C_{i^1}^{i^0} C_{j^1}^{j^0} \Lambda(i^0 - i^1, j^0 - j^1)$$

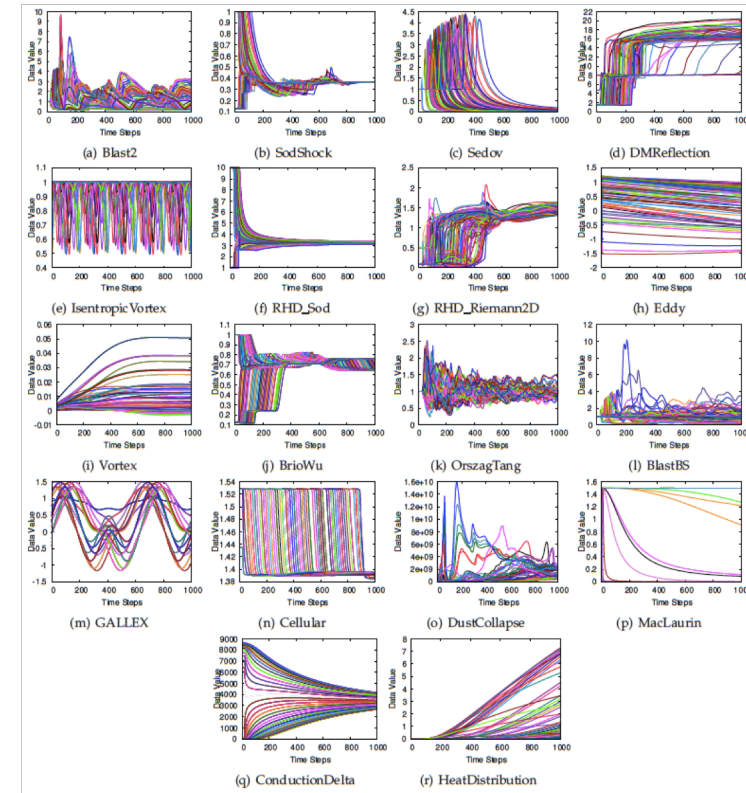
Decomposition
 (e.g. HOSVD: Tucker Tensor Decomposition)



Tucker-decomposition (Tucker, 1966)

$$\mathcal{X}_{ijk} = \sum_{l=1}^{r_1} \sum_{m=1}^{r_2} \sum_{n=1}^{r_3} \mathcal{G}_{lmn} a_{il} b_{jm} c_{kn} =: [[\mathcal{G}; A, B, C]]$$

- \mathcal{G} is called *core tensor*.
- Tucker-rank = (r_1, r_2, r_3)

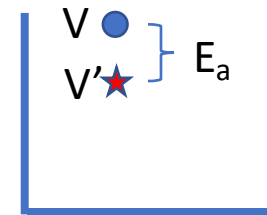


What's the most important feature for a lossy compressor of scientific data?

- **Providing a guarantee of data accuracy after decompression**
 - Users want the same results from analysis run on decompressed data
- Lossy compressors provides point wise error controls (**bounds**)
- **Two main types of error bounds** (E: error, V: initial dataset, V': decompressed dataset):

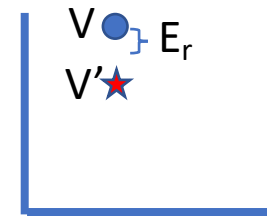
- **Absolute error bound:**
(absolute quantities.
E.g. position in mesh),

$$E_a = |V - V'|$$



- **Relative error error bound:**
(e.g. velocity, temperatures),

$$E_r = \frac{|V - V'|}{V}$$



Ok but how can we assess accuracy?

- **User analysis code**

- Metric of distance between analysis results from initial and decompressed dataset
- Not always doable: analysis too expensive to compute only before storage

- **Error Metrics (amount of error)**

- Respect of error bounds
 - RMSE
 - PNSR
 - Rate distortion
- } Point wise metrics
- } Statistical metrics

- **Advanced Error Metrics (nature of the error)**

- Error distribution
- Pearson correlation of the initial and decompressed data
- Autocorrelation of the compression error
- Correlation between the initial data and the error
- Spectral alteration
- Structural Similarity Index (SSIM)
- Preservation of derivatives

Example: PSNR

Peak signal to noise ration

$$X = \{x_1, x_2, \dots, x_N\} \xrightarrow{\text{Compression}} \tilde{X} = \{\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_N\} \xrightarrow{\text{Decompression}}$$

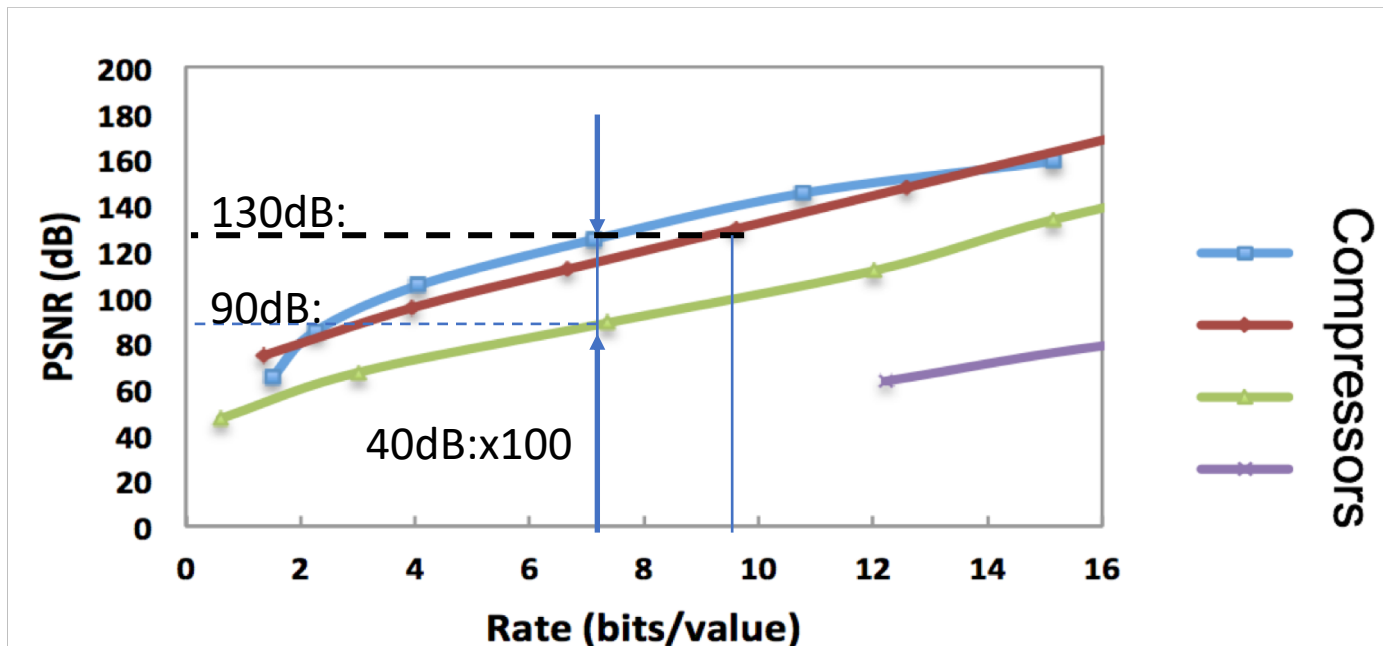
Value range: $R_X = x_{max} - x_{min}$

Point wise error: $e_{abs_i} = x_i - \tilde{x}_i$

$$rmse = \sqrt{\frac{1}{N} \sum_{i=1}^N (e_{abs_i})^2}$$

$$psnr = 20 \cdot \log_{10}\left(\frac{R_X}{rmse}\right)$$

Rate distortion diagram



dB	Amplitude ratio
-100 dB	10^{-5}
-50 dB	0.00316
-40 dB	0.010
-30 dB	0.032
-20 dB	0.1
-10 dB	0.316
-6 dB	0.501
-3 dB	0.708
-2 dB	0.794
-1 dB	0.891
0 dB	1
1 dB	1.122
2 dB	1.259
3 dB	1.413
6 dB	$2 \approx 1.995$
10 dB	3.162
20 dB	10
30 dB	31.623
40 dB	100
50 dB	316.228
100 dB	10^5

Example: Structural Similarity Index (SSIM)

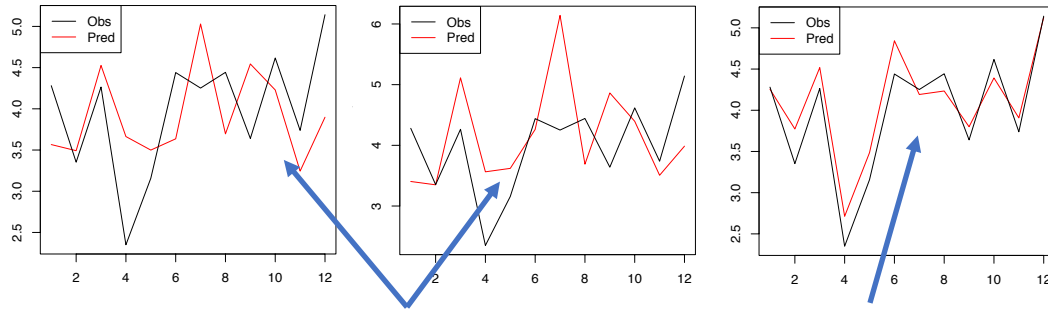
SSIM is a perception-based model that considers image degradation as perceived change in structural information
 SSIM is the product of 3 components that assess differences between two signals: Luminance, Contrast, Structure
 The closer to 1, the better match between signals in terms of mean, variance, and covariance.

$$SSIM(x, y) = \frac{(2\mu_x \mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

μ_x the **average** of x ;
 μ_y the **average** of y ;
 σ_x^2 the **variance** of x ;
 σ_y^2 the **variance** of y ;
 σ_{xy} the **covariance** of x and y ;

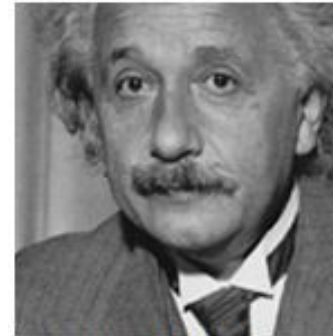
(Source: geophysical simulation data from atmospheric model)

lum: 1 cont: 0.947 struc: 0.305 SSIM: 0.289 lum: 0.999 cont: 0.994 struc: 0.328 SSIM: 0.326 lum: 0.999 cont: 0.985 struc: 0.961 SSIM: 0.946

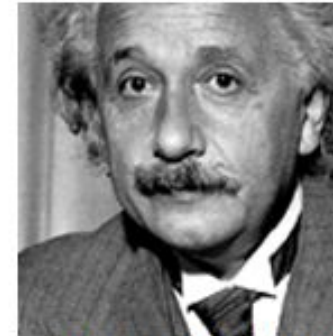


Signals have the same mean, but different co-variability

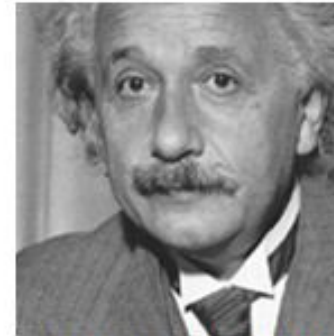
Signals have similar properties



MSE=0, SSIM=1



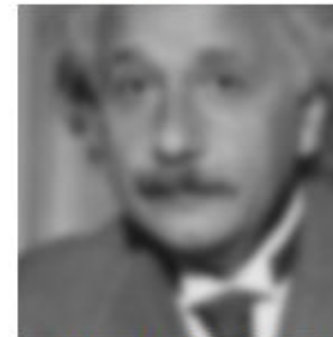
MSE=309, SSIM=0.928



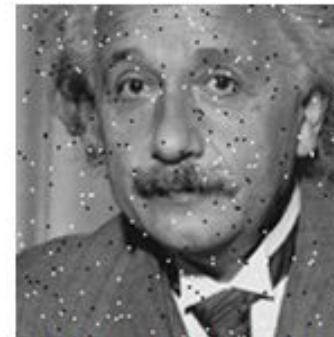
MSE=309, SSIM=0.987



MSE=309, SSIM=0.580



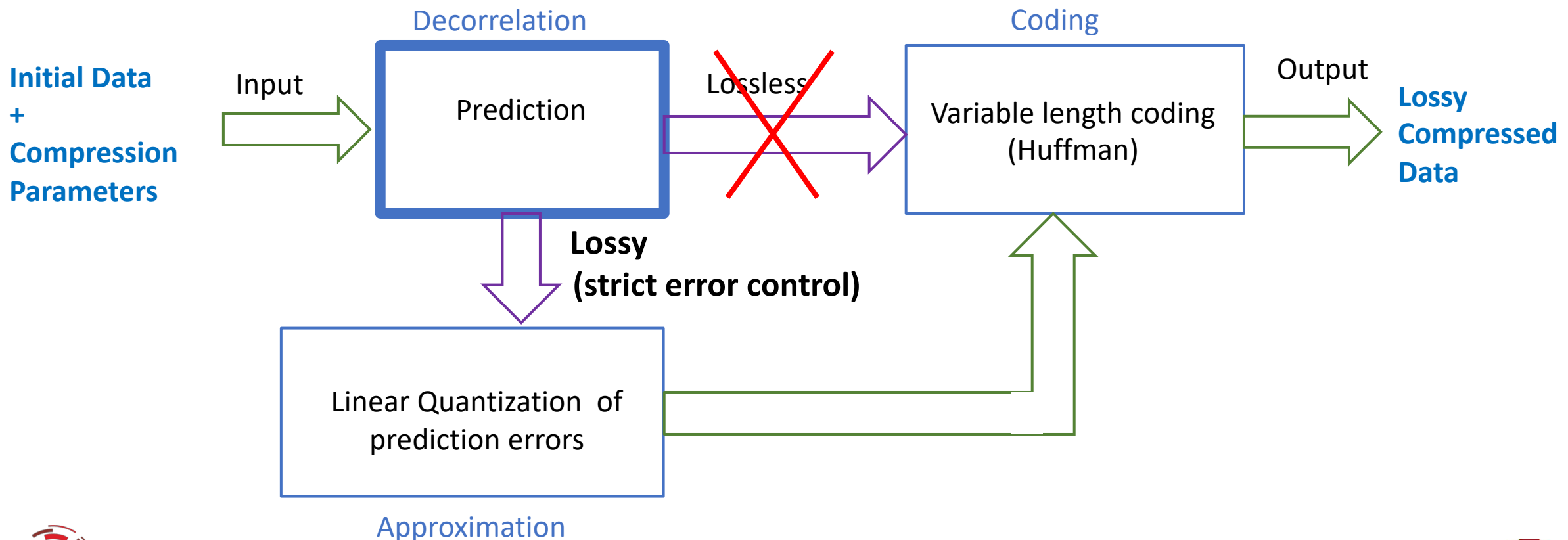
MSE=309, SSIM=0.641



MSE=309, SSIM=0.730

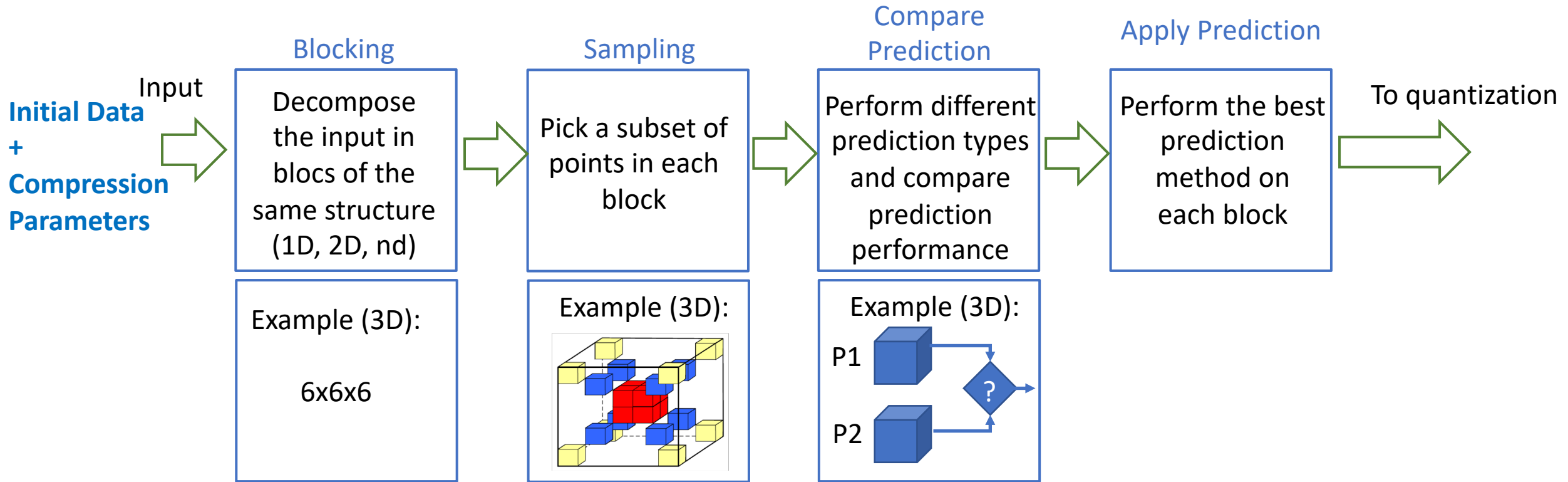
Let's look at ANL SZ

- Multi-stages, Prediction based lossy compressor
- Current version: **SZ 2.0** (Previous versions: SZ 1.1, SZ 1.4)



SZ Prediction stage

- Multi-stages, Prediction based lossy compressor
- Current version: **SZ 2.0** (Previous versions: SZ 1.1, SZ 1.4)



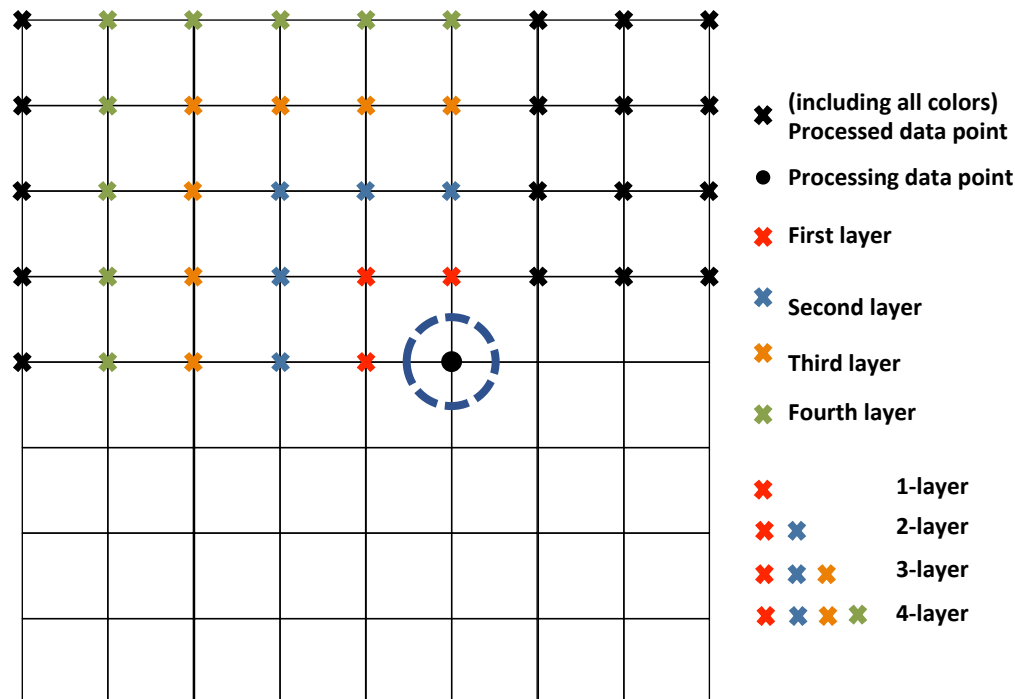
SZ 2.0 Prediction Method 1

Example for 2D:

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

1) Multi-dimensional
Multi-layer **Prediction**
(extension of Lorenzo)
2D example (9X9 block)

Input bloc of
Floating point
data



Produces floating point
Numbers (predictions
for each data of the bloc)

The prediction function is known by the decompressor.

→ No need to store its description in the compressed file.

SZ 2.0 Prediction Method 2

2) Regression

Compute hyperplane coefficients

The objective of the regression model is to minimize the squared error (SE) between predicted and original values

The regression coefficients of a 3D dataset with dimensions n_1, n_2, n_3 can be calculated as:

$$\begin{cases} \beta_1 = \frac{6}{n_1 n_2 n_3 (n_1 + 1)} \left(\frac{2V_x}{n_1 - 1} - V_0 \right) \\ \beta_2 = \frac{6}{n_1 n_2 n_3 (n_2 + 1)} \left(\frac{2V_y}{n_2 - 1} - V_0 \right) \\ \beta_3 = \frac{6}{n_1 n_2 n_3 (n_3 + 1)} \left(\frac{2V_z}{n_3 - 1} - V_0 \right) \\ \beta_0 = \frac{V_0}{n_1 n_2 n_3} - \left(\frac{n_1 - 1}{2} \beta_1 + \frac{n_2 - 1}{2} \beta_2 + \frac{n_3 - 1}{2} \beta_3 \right) \end{cases}$$

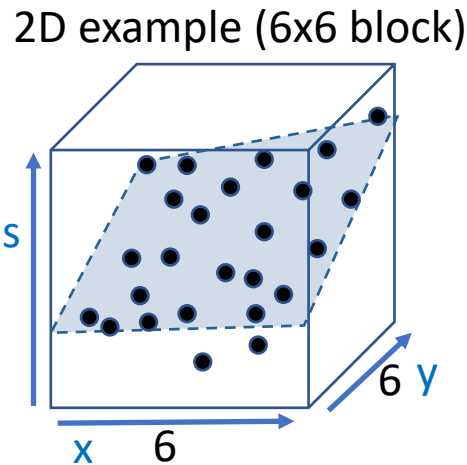
$$\text{where } V_0 = \sum_{i=0}^{n_1-1} \sum_{j=0}^{n_2-1} \sum_{k=0}^{n_3-1} f_{ijk}, \quad V_x = \sum_{i=0}^{n_1-1} \sum_{j=0}^{n_2-1} \sum_{k=0}^{n_3-1} i * f_{ijk},$$

$$V_y = \sum_{i=0}^{n_1-1} \sum_{j=0}^{n_2-1} \sum_{k=0}^{n_3-1} j * f_{ijk}, \quad V_z = \sum_{i=0}^{n_1-1} \sum_{j=0}^{n_2-1} \sum_{k=0}^{n_3-1} k * f_{ijk}.$$

Input bloc of Floating point data



Values

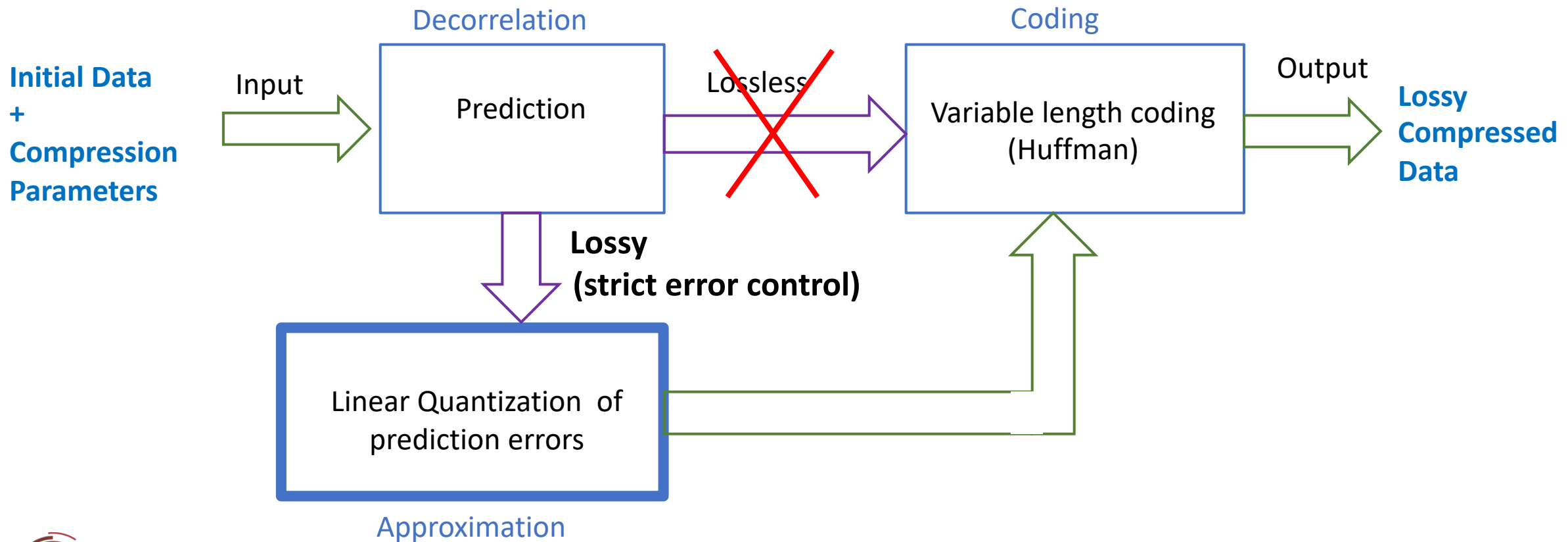


Produces floating point Numbers (predictions for each data of the bloc)

The coefficients resulting from the regression are NOT known by the decompressor.
 → Need to store the coefficients for each block in the compressed file.

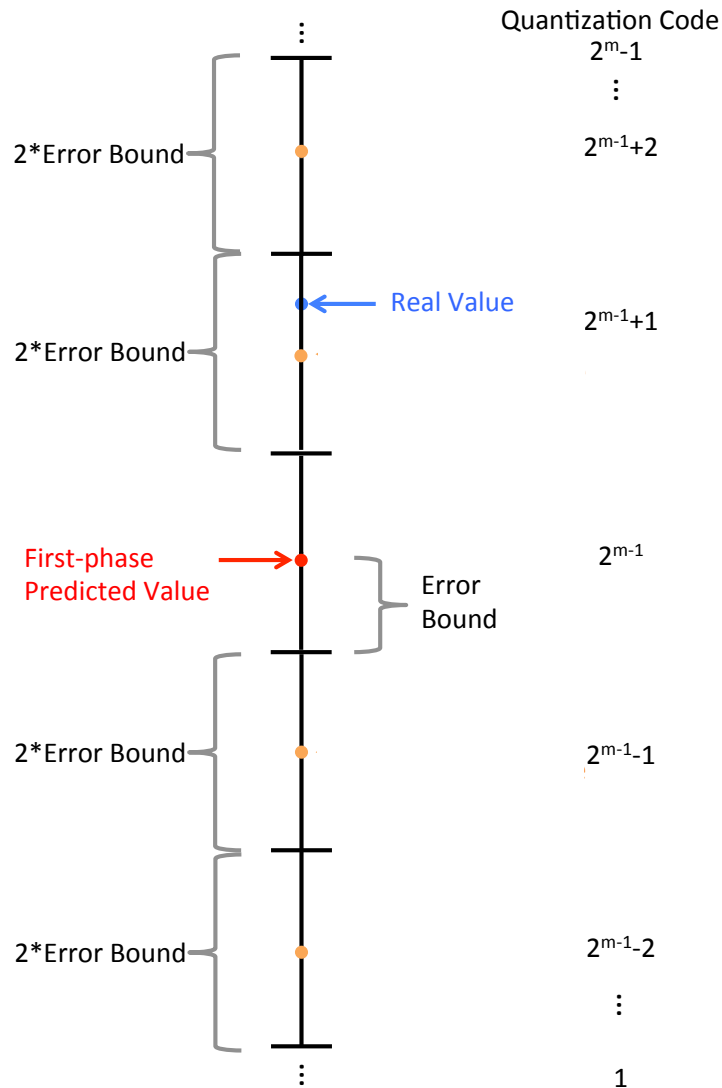
SZ Design Principles

- Multi-stages, Prediction based lossy compressor
- Current version: **SZ 2.0** (Previous versions: SZ 1.1, SZ 1.4)

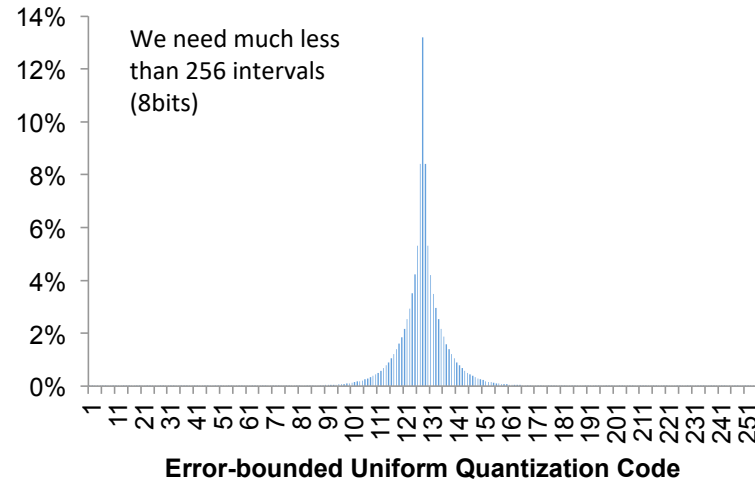


SZ 2.0 Quantization Stage

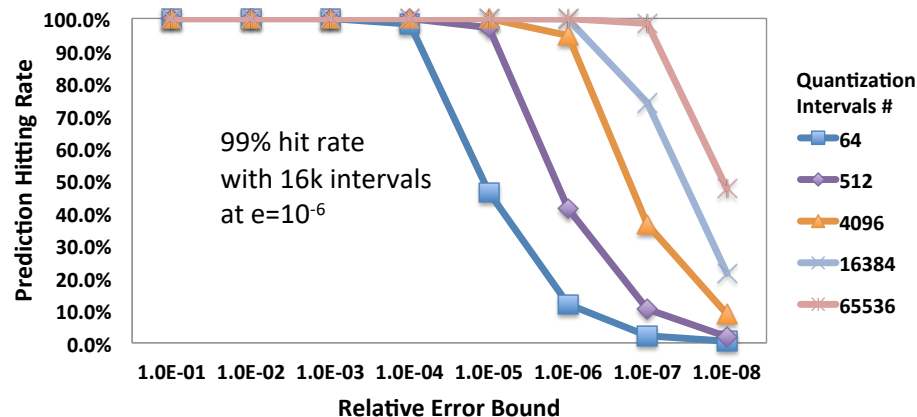
2) Linear Quantization of prediction error (map data into quantization bins, #bins defined by users or SZ)



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)



1 2 3

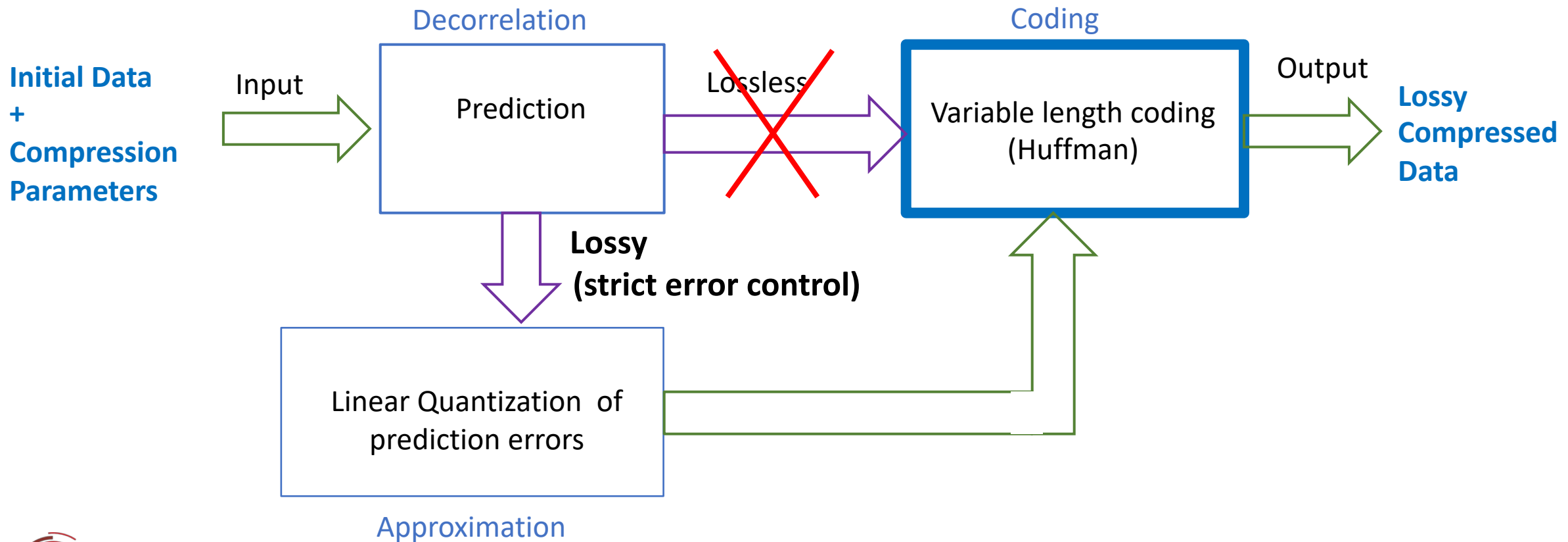
Array of unpredictable data (FP)



Data are ordered in the arrays in their initial order

SZ Design Principles

- Multi-stages, **Multi-algorithm** Prediction based Lossy compressor
- Current version: **SZ 2.0** (Previous versions: SZ 1.1, SZ 1.4)

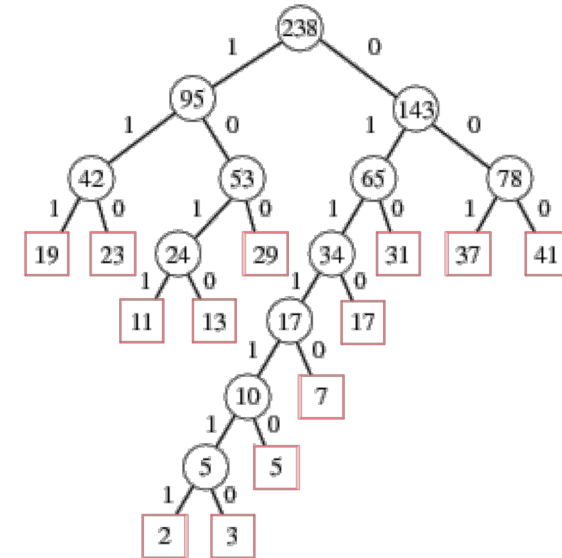


SZ 2.0 Coding Stage

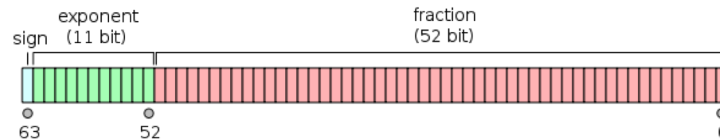
3) Variable length coding (Huffman)

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 Zstandard (L77 + Finite State Entropy): improve the compression by about 10% (Zstandard much faster than GZIP, improves speed of SZ 2.0 compared to SZ 1.4)

SZ Use-cases

We are seeing an increasing diversity/number of use-cases

“Classic” use-cases:

- 1) Visualization
- 2) Reducing storage footprint (offline compression)
- 3) Reducing I/O time (on-line, in-situ compression)

Recently identified use-cases:

- 4) Reducing streaming intensity (recent for generic floating-point compressors)
- 5) Lossy checkpoint/restart from lossy state
 - reduce checkpoints footprint on storage – adjoint, accelerate checkpointing
- 6) Accelerate computation
 - By improving the memory-byte/flop ratio (reducing the pressure on memory bandwidth)
- 7) Re-computation Avoiding
 - By reducing the memory footprint
- 8) Running larger simulations
 - By reducing the memory footprint

Goals:

- Show the diversity of uses cases through real life applications
- Present real-life performance gains
- Inspire potential new use-cases

Source:

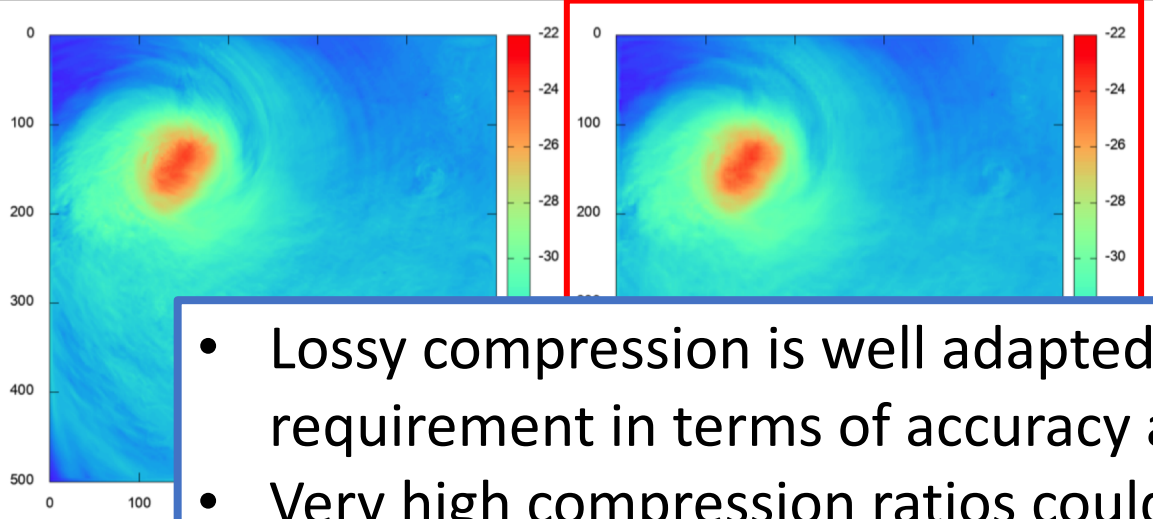
Most examples from the Exascale Computing Project (ECP).

Most example use the SZ compressor.

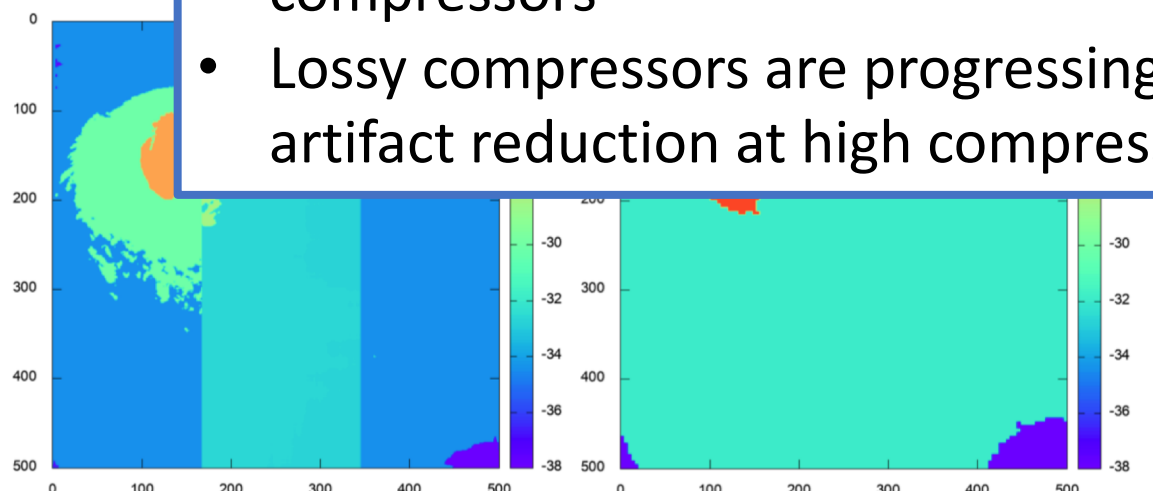
1) Visualization

NYX (velocity x:slice 100) CR=**156:1** (EB: 0.03)

Cosmology: Adaptive mesh hydrodynamics + N-body simulation



(a)

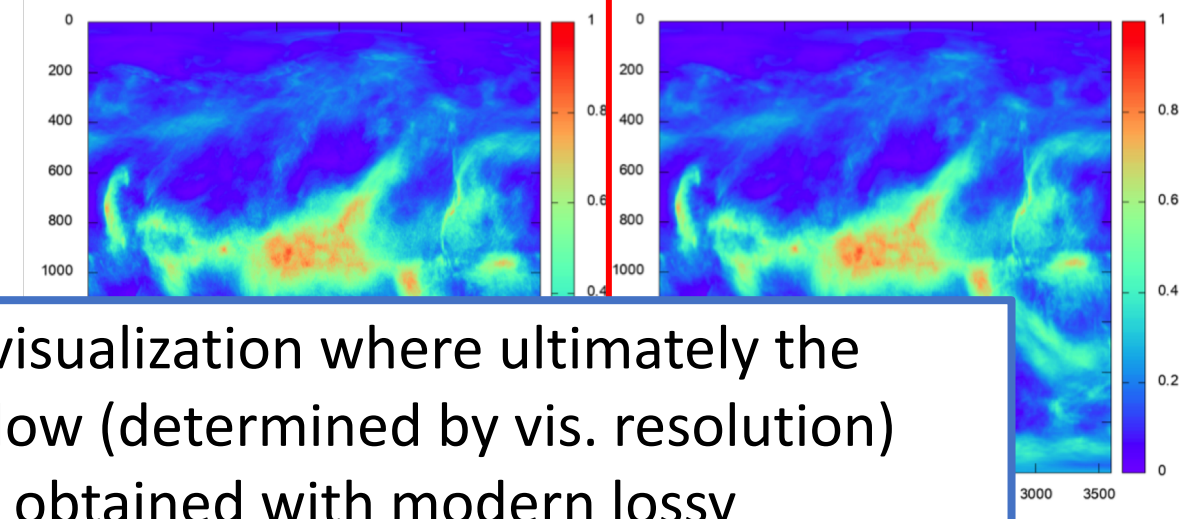


SZ 1.4 (PSNR=39)

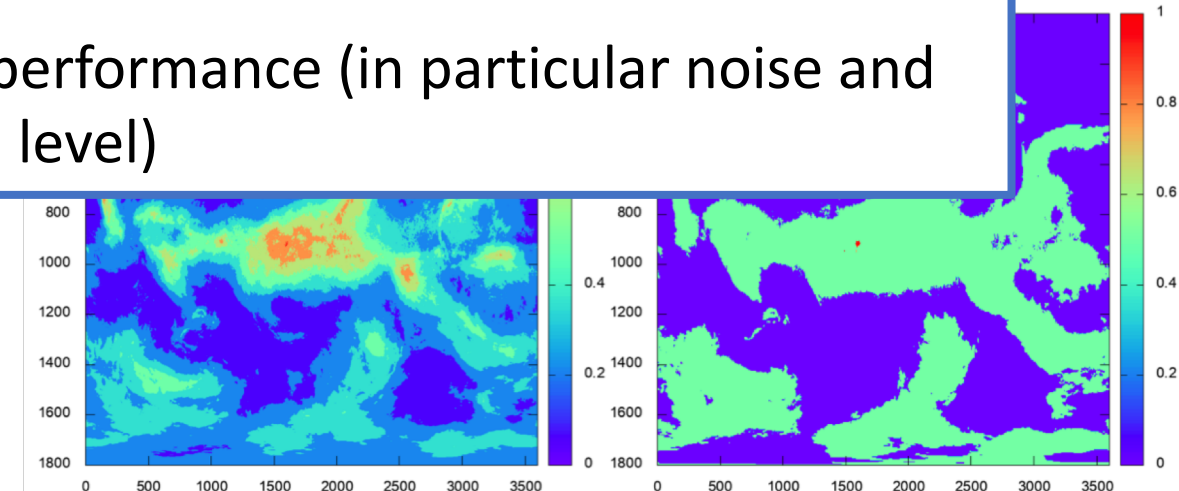
(d) ZFP (PSNR=31)

CESM ATM(CLDHGH) CR=**295:1** (EB: 0.06)

Climate Atmospheric mode (High cloud cover): fluid dynamics



M=0.9806)



SZ 1.4 (PSNR=27,SSIM=0.8842)

(d) ZFP (PSNR=15,SSIM=0.3168)

- Lossy compression is well adapted to visualization where ultimately the requirement in terms of accuracy are low (determined by vis. resolution)
- Very high compression ratios could be obtained with modern lossy compressors
- Lossy compressors are progressing in performance (in particular noise and artifact reduction at high compression level)

2) Reducing Storage Footprint

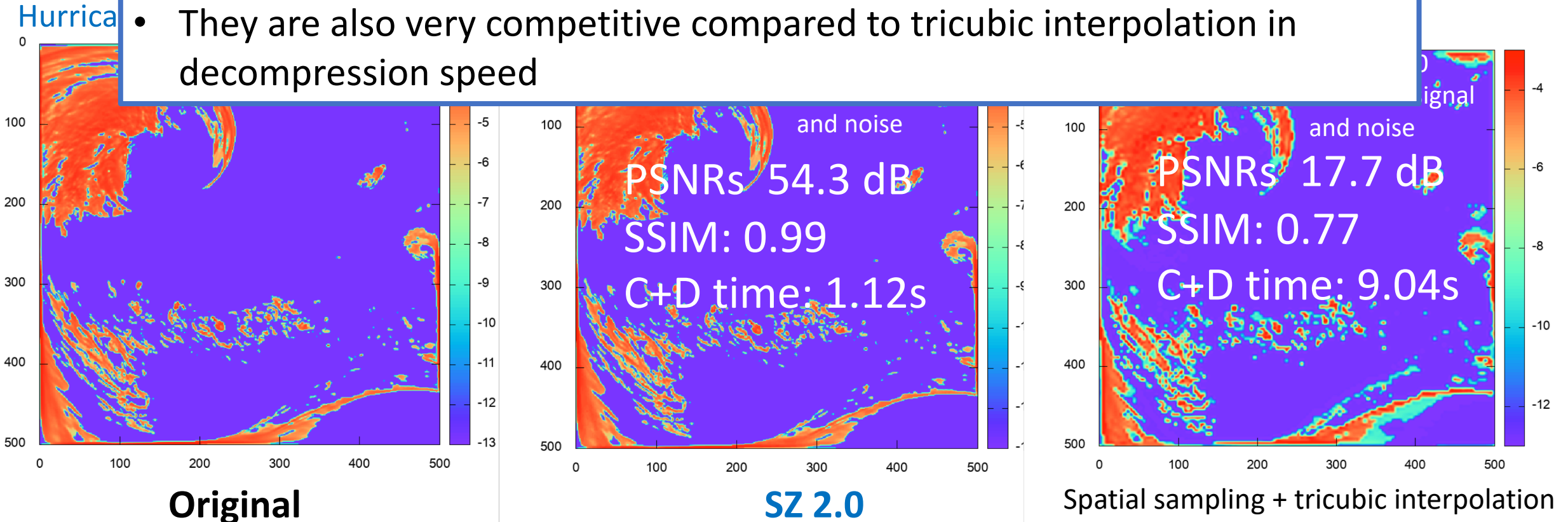
$$psnr = 20 \cdot \log_{10}\left(\frac{R_X}{rmse}\right)$$

Goal for this use-case is to keep a ratio of about 1000 between the initial signal and the noise introduced by the lossy compression error.

And compare with the classic technique of spatial sampling and reconstruction

X. Liang, S. Di, D. Tao, S. Li, S. Li, H. Guo, Z. Chen, F. Cappello, et al. *IEEE Transactions on Big Data*, 2018

- Modern lossy compressors can reduce drastically the footprint on storage while keeping a very high accuracy
- They are also very competitive compared to tricubic interpolation in decompression speed



3) Accelerating I/O

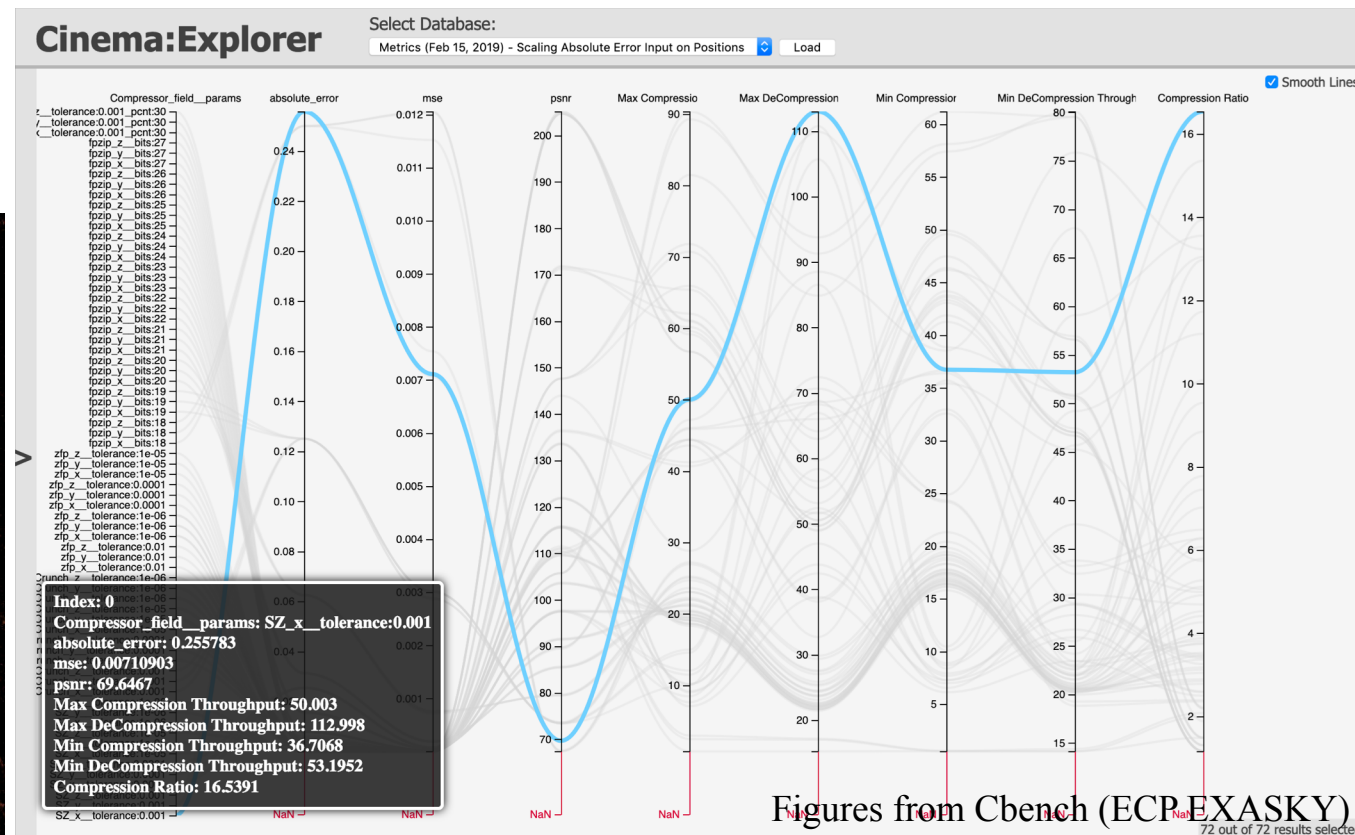
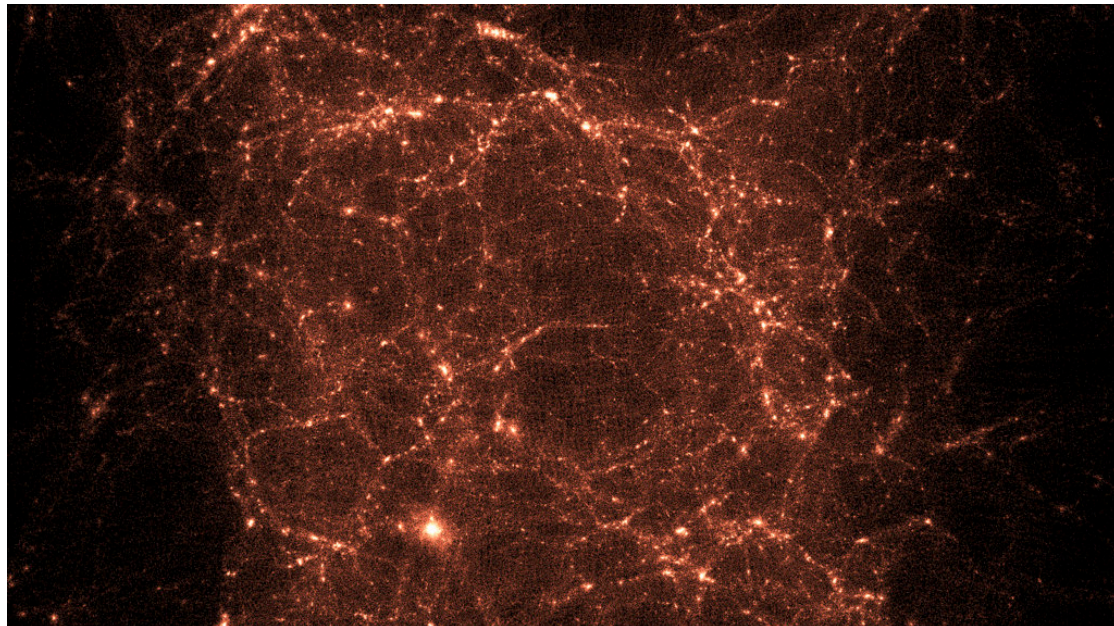
ECP HACC: N-body problem with domain decomposition, medium/long-range force solver (particle-mesh method), short-range force solver (particle-particle/particle-mesh algorithm).

Particle dataset: 6 x 1D array (x, y, z, vx, vy, vz)

SZ 2.0: CR ~5 (~6bits/value) at 10^{-3} error bound

Preferred error controls:

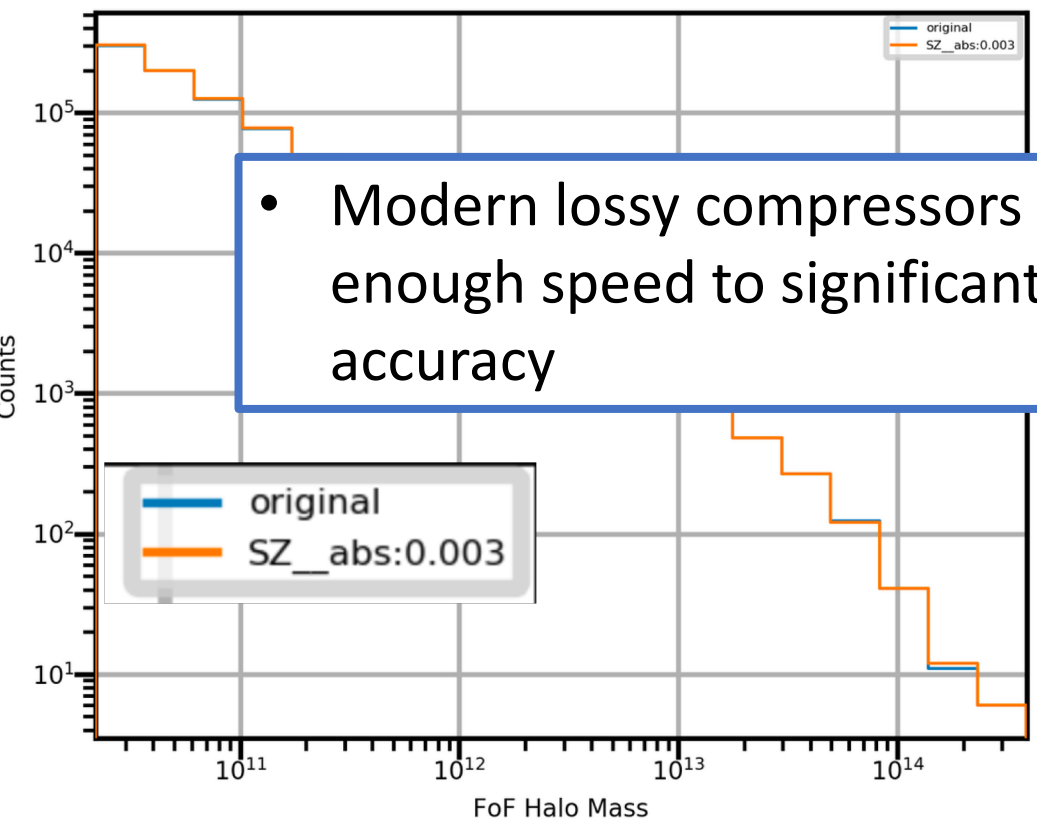
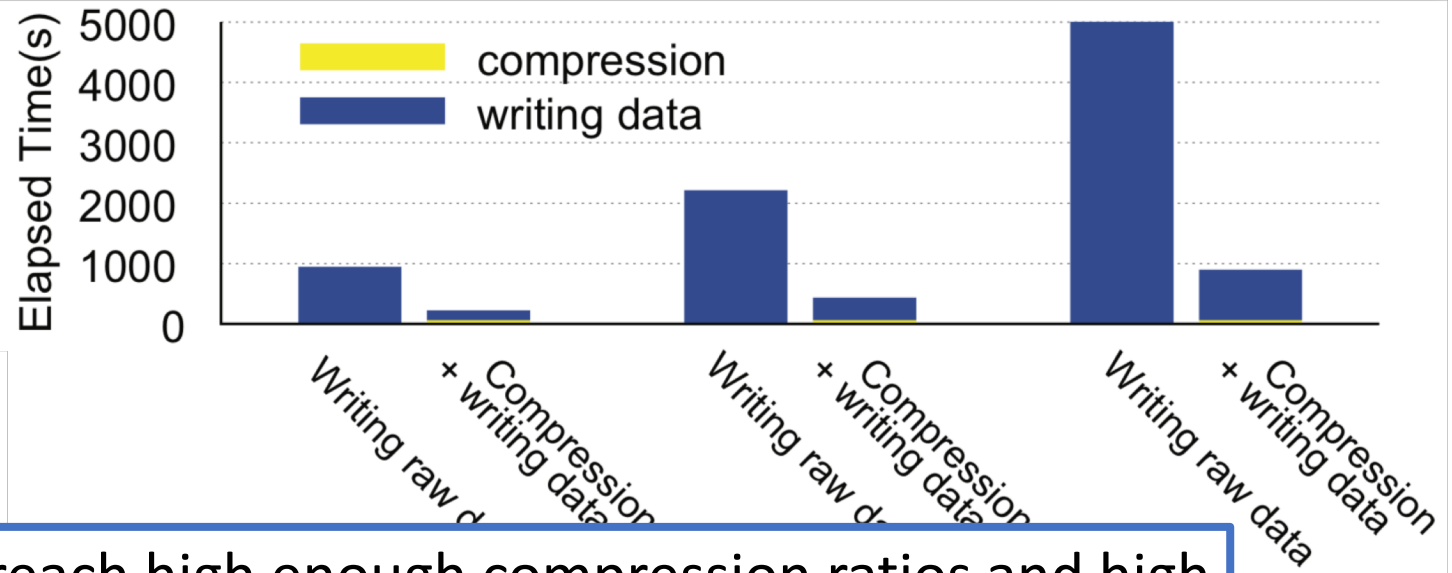
- Point wise max error (Relative) bound
- Absolute (position), Relative (Velocity)



3) Accelerating I/O

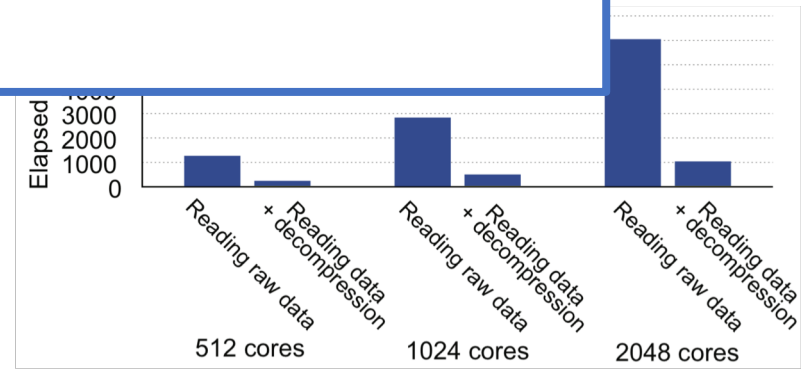
ECP HACC

Results validation
3kpc absolute error bound



• Modern lossy compressors reach high enough compression ratios and high enough speed to significantly reduce I/O time, while keeping a high accuracy

- File-per-process mode with POSIX I/O for reading/writing data in //
- Each core has 3.14GB to write



I/O time reduced by ~5x-~6x

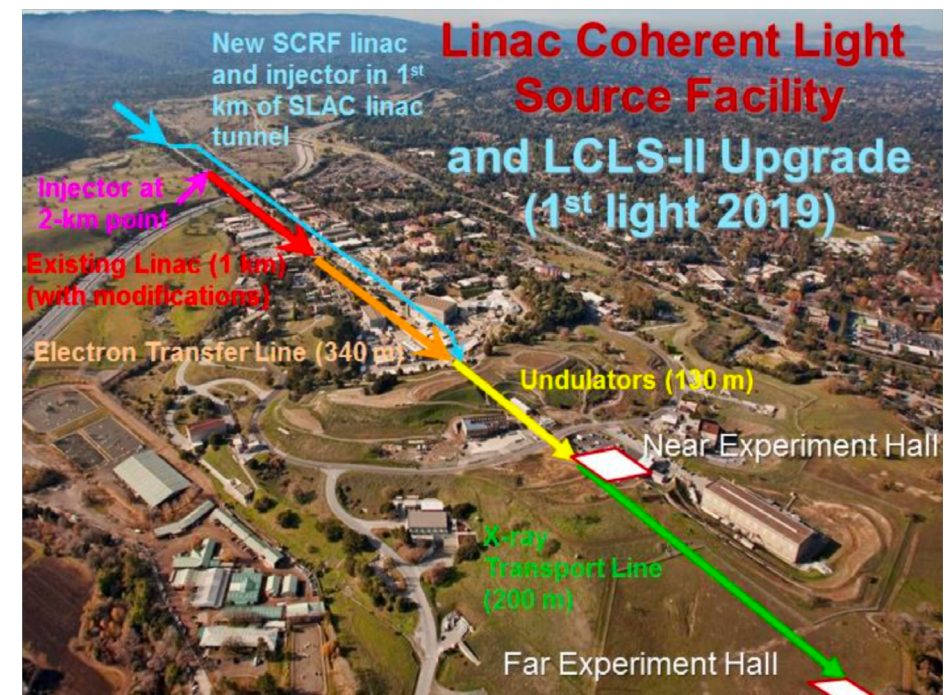
The writing and reading times without compression are estimations (too long for experiment)



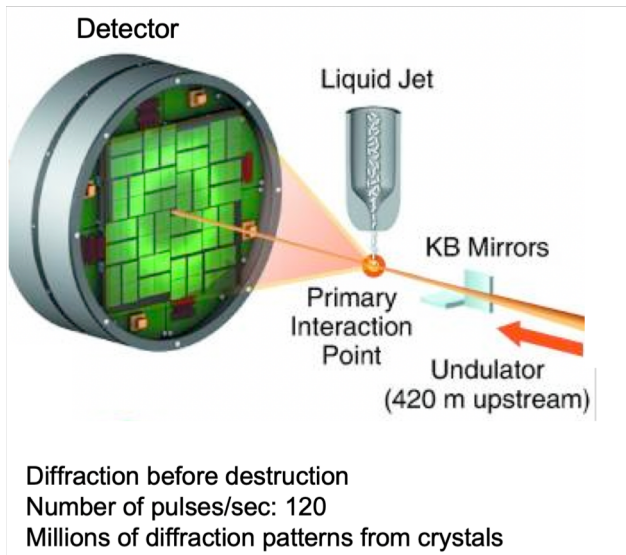
4) Reducing Streaming Intensity

ECP EXAFEL: Context of LCLS-II

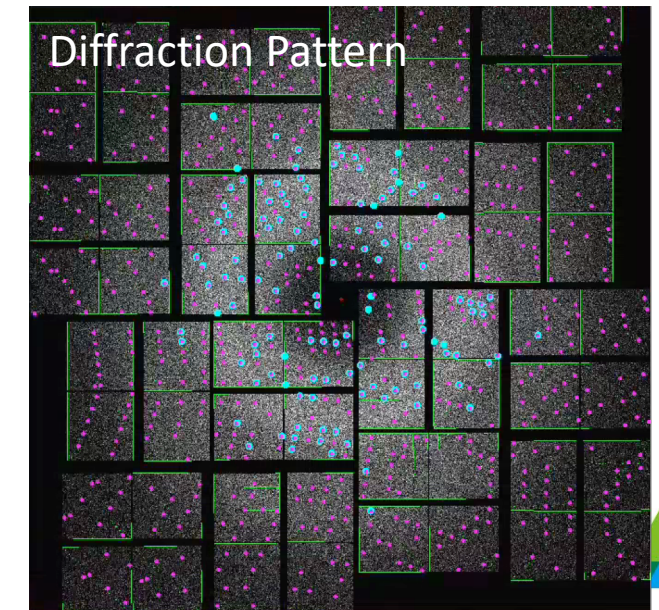
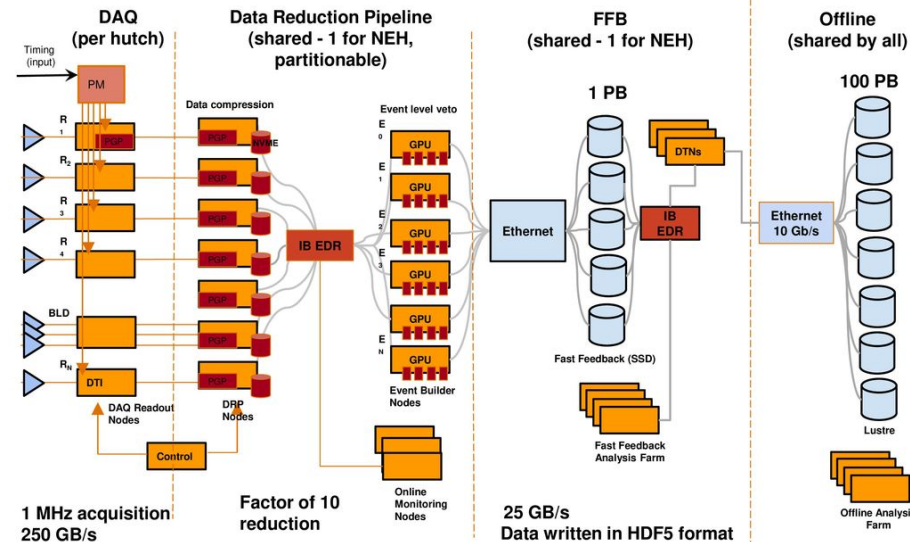
- X-ray crystallography to determine the atomic and molecular structure of a crystal
- Crucial for medical applications (e.g. understand membrane protein structure)
- A liquid jet injects the crystal and an X-ray pulse strikes the molecules.
- This produces diffraction patterns on the detector
- Patterns are analyzed to discover the structure of the crystal/molecule



T.O. Raubenheimer for the LCLS-II Collaboration, SLAC, Menlo Park, CA 94025, USA
6th International Particle Accelerator Conference, VA, USA, 2015, JACoW Publishing



LCLS-II Data System



4) Reducing Streaming Intensity

Experiment cxic0415 run 90 event 104859

-Detector produces 2D images:

-4M pixel/event

-All LCLS-II area detectors per experiment: 250GB/s

-Today technology: ~x1000s disks

to sustain

-Data is u

(RAW, Ca

XTC2 format

Compression goals:

-**Goal: CR of 10 with error bound**

-Speed 500 MB/s/core

→ True Co-Design (algorithm, hardware)

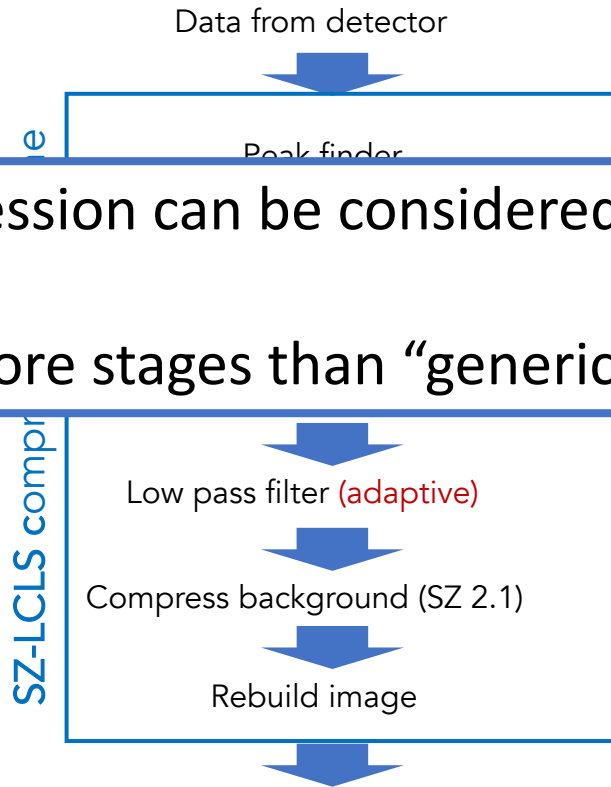
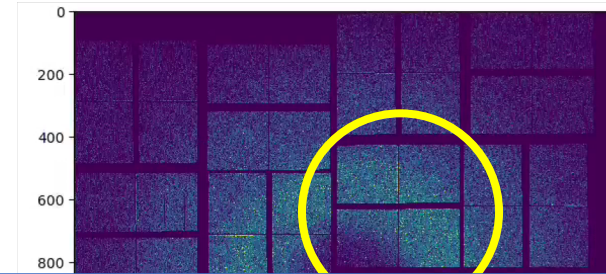
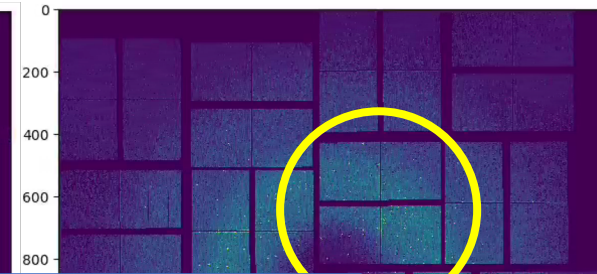


Image from the detector

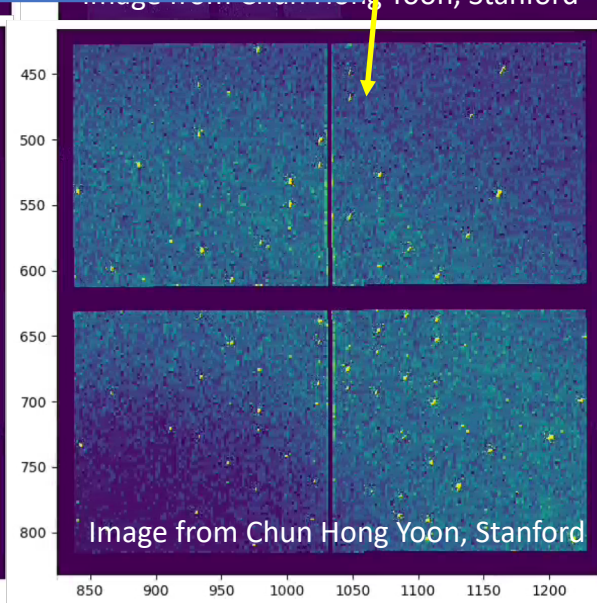
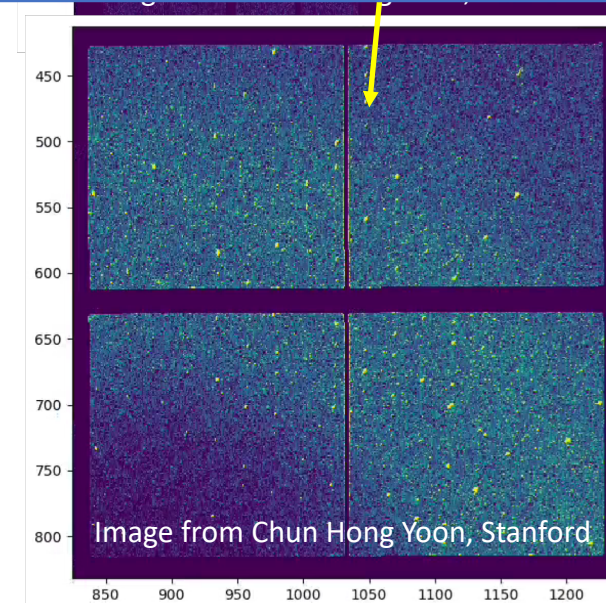


Decompressed image from SZ-LCLS

Compression ratio: 30



• Lossy compression can be considered for effectively reducing streaming intensity.
• May need more stages than “generic” lossy compression



5) Accelerating Checkpoint/Restart

ECP NWCHEM: Particles are represented by wave functions
Coupled-cluster approaches are some of the most successful methods used to solve quantum states of complex many-body systems.

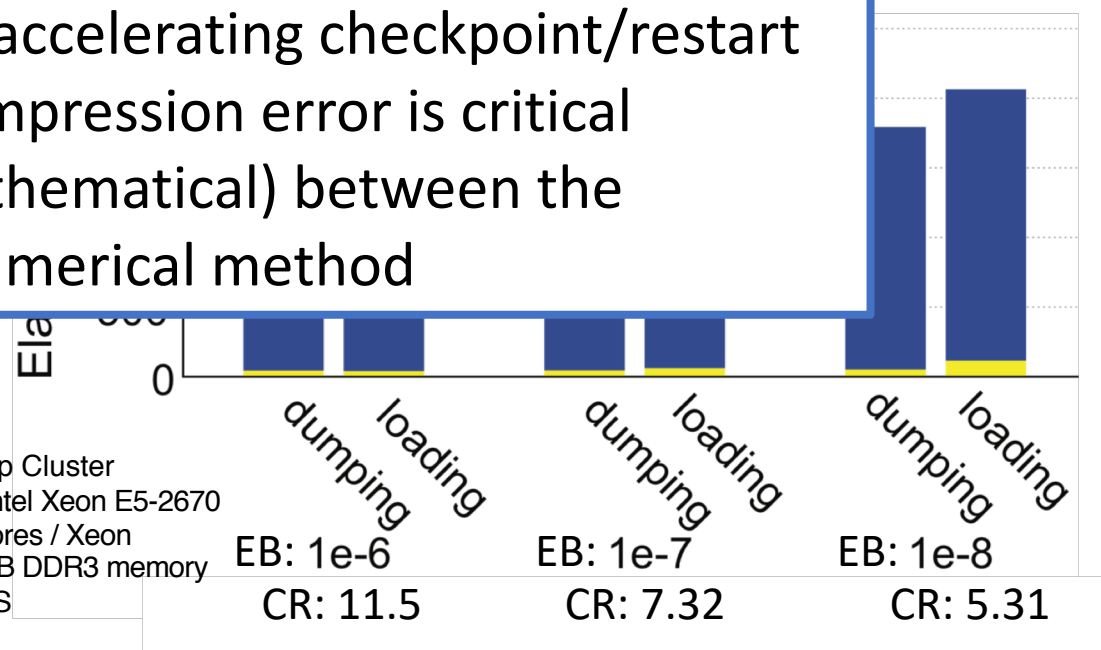
#iter	Error Bound	Comp Ratio	CCSD correlation energy/Hartree
27	original	1	-0.095440748811744
10+17	1E-8	~9	-0.095440748697882
10+17	1E-6	~11	-0.095440744046987
10+17	1E-4	~40	-0.095440297611173

$$|\Psi\rangle = e^{\hat{T}} |\Phi_0\rangle \quad \hat{T} = \hat{T}_1 + \hat{T}_2 + \dots + \hat{T}_N$$

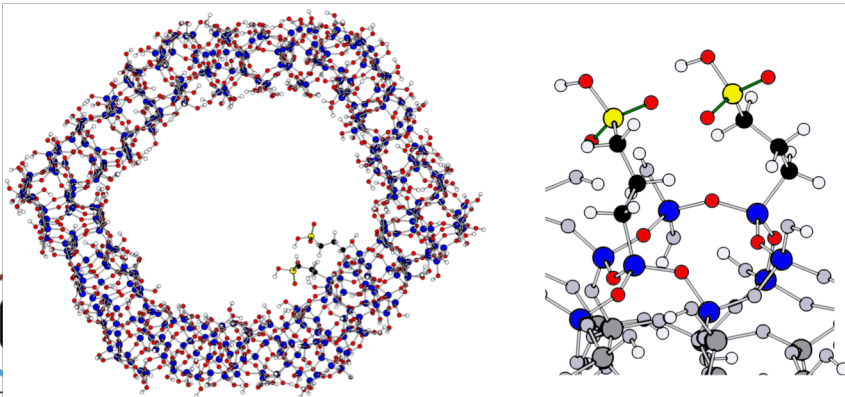
Cluster

Dumping time on PFS (no compress.): ~10k seconds
 Loading time on PFS (no compress.): ~11k seconds.

- Lossy compression can be considered for accelerating checkpoint/restart
- Understanding the level of acceptable compression error is critical
- There is currently no established link (mathematical) between the compression error and the error of the numerical method



- Wavefunction of the CC theo
- Large
- Goal: comp
- 1D dataset, reference error control: Absolute Error Bound
- Improved NWChemEx performance ⇨ tackle larger problems



ANL Bebop Cluster

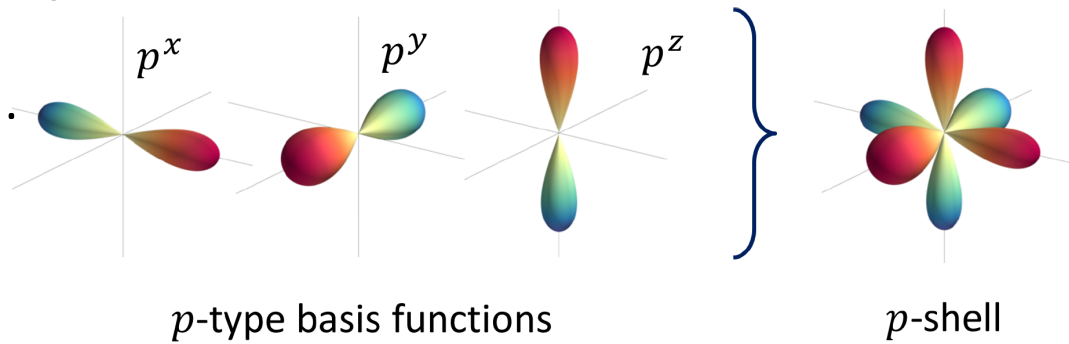
- 2 x Intel Xeon E5-2670
- 16 cores / Xeon
- 64 GB DDR3 memory
- GPFS

Accelerate the dumping and loading time **10x** at acceptable error bound (10E-6).

6) Avoiding re-Computation 1/2

ECP GAMESS: Quantum Chemistry

→ The goal is to obtain the **wavefunction** of a chemical system by solving the **Schrödinger equation**.



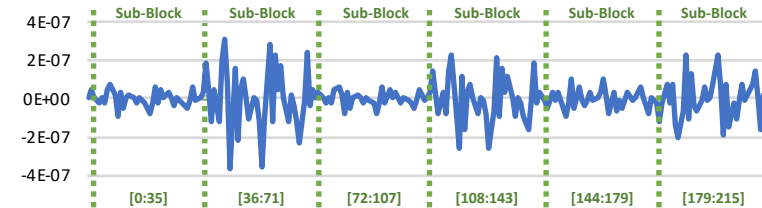
Solving the Schrödinger equation requires a huge amount of Electron Repulsion Integrals (ERIs)

The number of ERIs is so large that they cannot fit in memory. They need to be recomputed at each iteration.

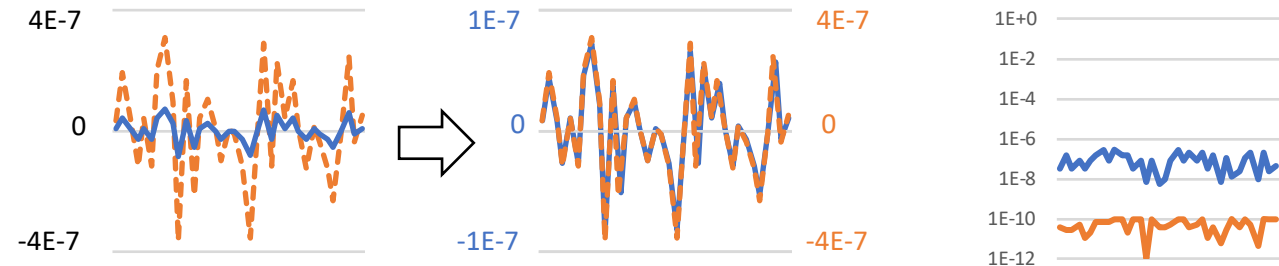
Compression is used to reduce the size of the ERIs to make them fit in memory, removing the need for re-computation.

1D dataset, preferred error controls:

- Point wise max error (absolute) bound
- Extremely low error bounds: 10^{-10}



Data points are generated in blocks. Typically four nested FOR loops traverse all different types of orbital shells of each atom. → Produces periodic behavior in the output data, where the periodicity is defined by the shell types in atom couples.



New SZ capability: Pattern based predictor

→ PaSTRI algorithm (developed at ANL)

6) Avoiding re-Computation 2/2

- **ECP GAMESS**

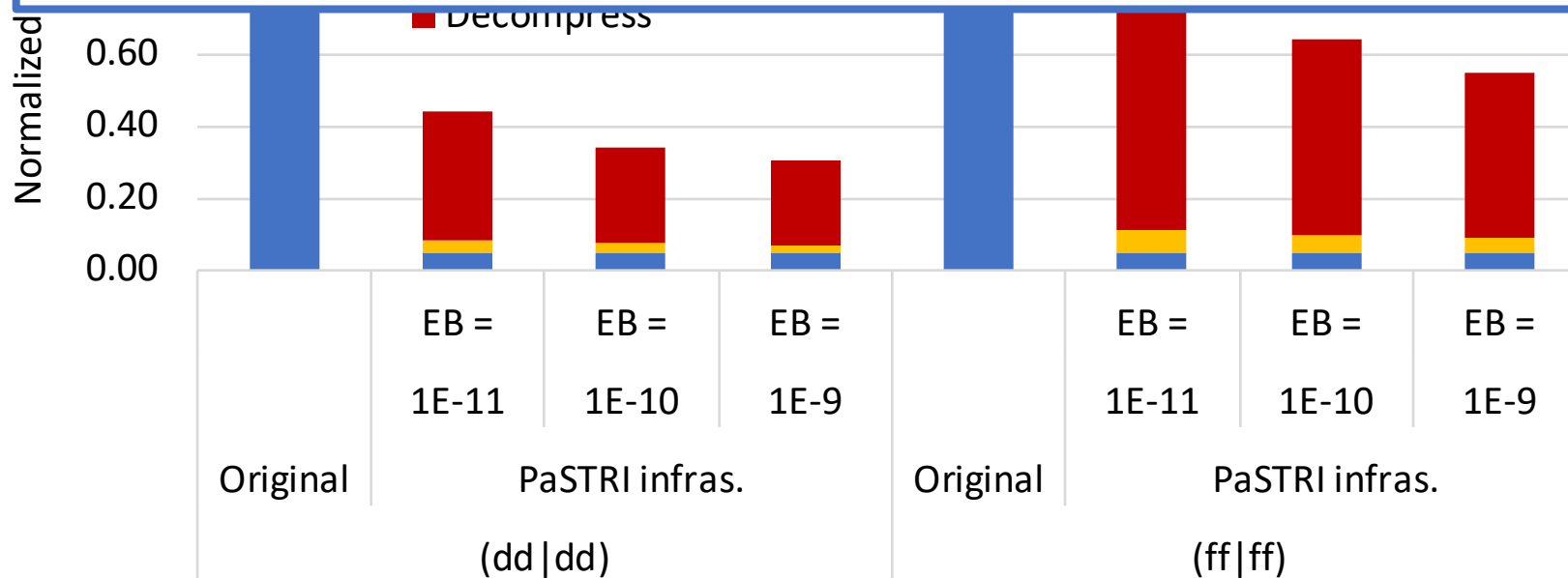
- 16.8X compression ratio,

- 66% lossy compression can be accelerate computation via re-computation avoidance if some data need to be recomputed at each iteration because they do not fit in memory

- 1. The data should be write once/read many time to be beneficial

- de Decompression need to be fast enough to not increase the overall computation time

award
IEEE Cluster 2018



7) Reducing Memory Footprint

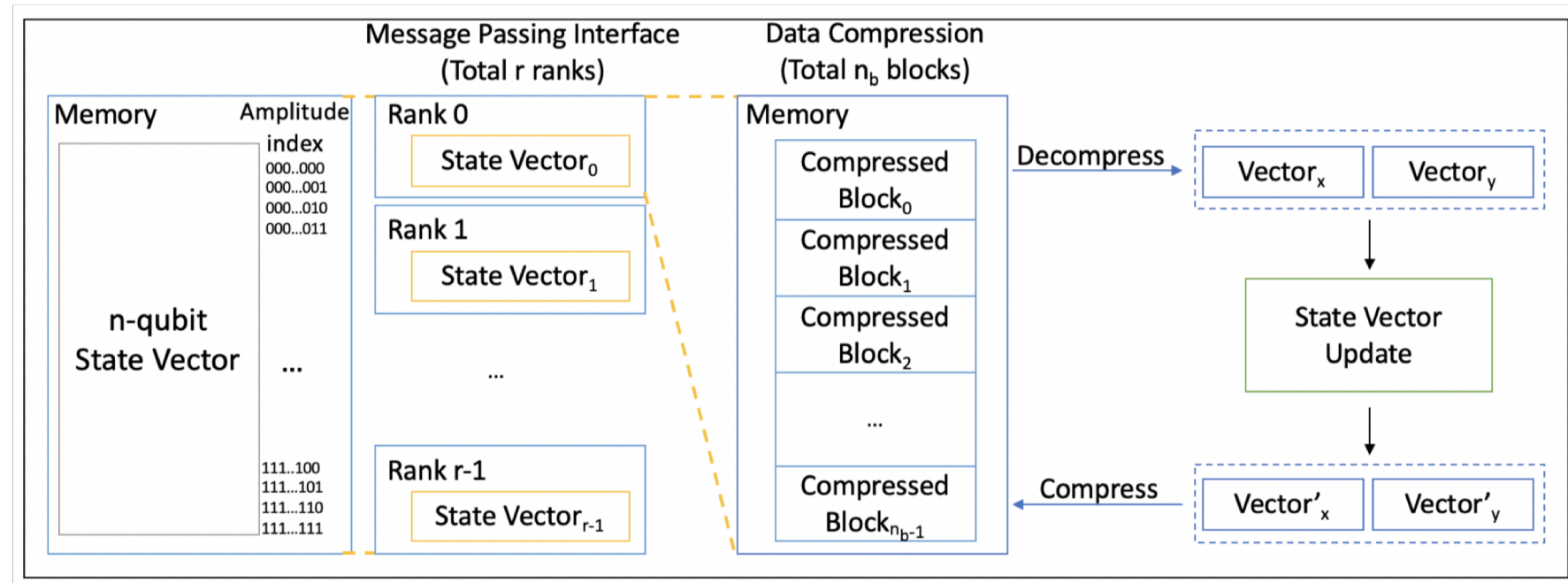
One of the problem in quantum computing simulators is the storage of the quantum state. The quantum state is a vector of probabilities (1 probability for each quantum bit configuration). The size of the vector state is 2^n , with n being the number of quantum bits to simulate. **1D dataset.**

Simulating 50 qubits would require $2^{50} \times 2^4 = 16\text{PB}$ of RAM

So compromises are made: lower #qubits, Depth, #amplitudes:

	General Technique	Qubits	Depth	# of Amplitudes
Intel	Full amplitude-vector update	45	High	All
IBM	Tensor-slicing with minimized communication	7x7	27	All
		7x8	23	2^{37} out of 2^{56}
Google	Preprocessing using undirected graphical model	7x8	30	1
USTC	Qubit partition with partial vector update	8x9	22	1
Sunway	Dynamic programming qubit partition	7x7	39	All
		7x7	55	1
Alibaba	Undirected graphical model with parallelization	9x9	40	1

Use lossy compression of the Quantum state to increase the number of Qubits that could simulated with given memory size



7) Reducing Memory Footprint

Fidelity:

$$F(ideal, sim) = |\langle \psi_{ideal} | \psi_{sim} \rangle| \geq \sum_{i=0}^{2^n-1} a_i^2 (1 - \delta) = 1 - \delta.$$

ANL Theta full size: 4K nodes, 768 TB

QAOA: polynomial time algorithm for finding “a ‘good’ solution to an optimization problem (NP-Hard problem)

Grover: Find entries in unsorted databases in $O(\text{square root of the number items in the data base})$.

QFT: Quantum Fourier transform

- Lossy compression can be considered for running problems that cannot fit in memory (run larger problem)
- The application need to tolerate the introduction of compression error at each load from the memory.
- Execution speed might decrease significantly because the compression/decompression happen for each store/load

Benchmark										QFT
Number of Qubits (Memory Required)										36 (1 TB)
Number of Gates										3258
Number of Nodes										1
Total System Memory (Sys Mem / Required)										192 GB (18.75%)
Total Time (Hours)										78.98
Compression Time										57.86%
Decompression Time	1.87%	3.73%	4.08%	31.47%	22.19%	33.78%	30.59%	27.64%	25.52%	37.68%
Communication Time	32.7%	20.98%	36.73%	0.12%	0.57%	0.02%	0.03%	0.22%	0.23%	2.56%
Computation Time	63.47%	70.70%	57.15%	12.60%	36.97%	7.08%	10.8%	27.16%	33.22%	1.9%
Time per Gate (Sec)	93.34	40.49	5.78	64.69	119.22	173.65	107.86	61.02	92.64	87.27
Simulation Fidelity	0.996	0.996	1	0.987	0.993	0.933	0.985	0.999	0.999	0.962
Compression Ratio	7.39×10^4	8.26×10^4	1.06×10^4	6.03	9.40	8.16	10.05	4.85	9.25	21.34

Accelerating Computation

H. Fu, et al., "18.9-Pflops Nonlinear Earthquake Simulation on Sunway TaihuLight: Enabling Depiction of 18-Hz and 8-Meter Scenarios", Proceedings of SC17

2017 Gordon Bell Award: 18.9-Pflops Nonlinear Earthquake Simulation on Sunway TaihuLight: Enabling Depiction of 18-Hz and 8-Meter Scenarios

Finite difference, non-linear scheme

Optimization of Finite Difference Method (FDM) on Sunway TaihuLight

Designed a lossy compression scheme
Benefit from

- 24% more computation
- double the accuracy

- Lossy compression can be considered to accelerate computation through the reduction of pressure on the memory bandwidth
- The application need to tolerate the introduction of compression error at each load from the memory.
- Additional hardware structure are needed (scratch pad)

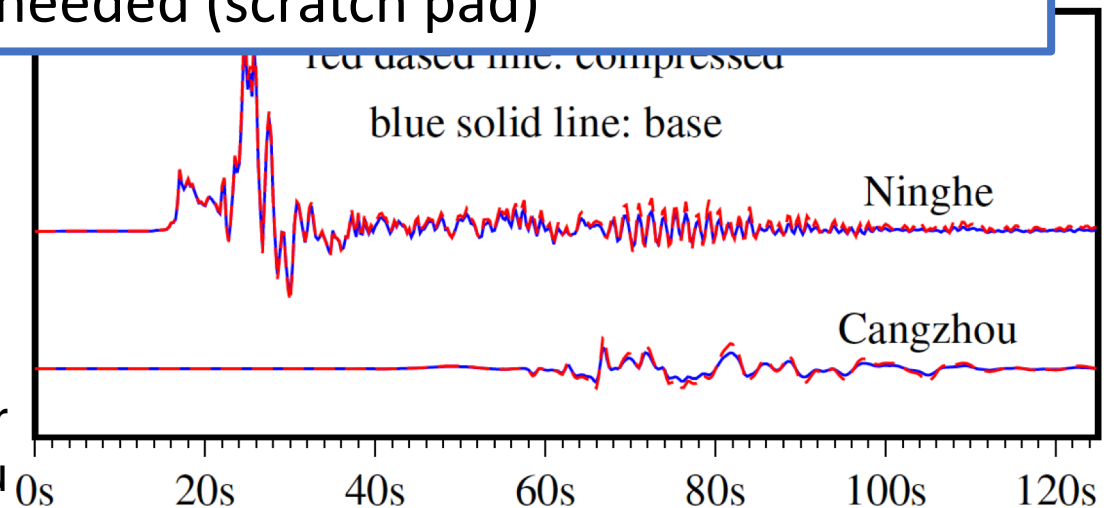
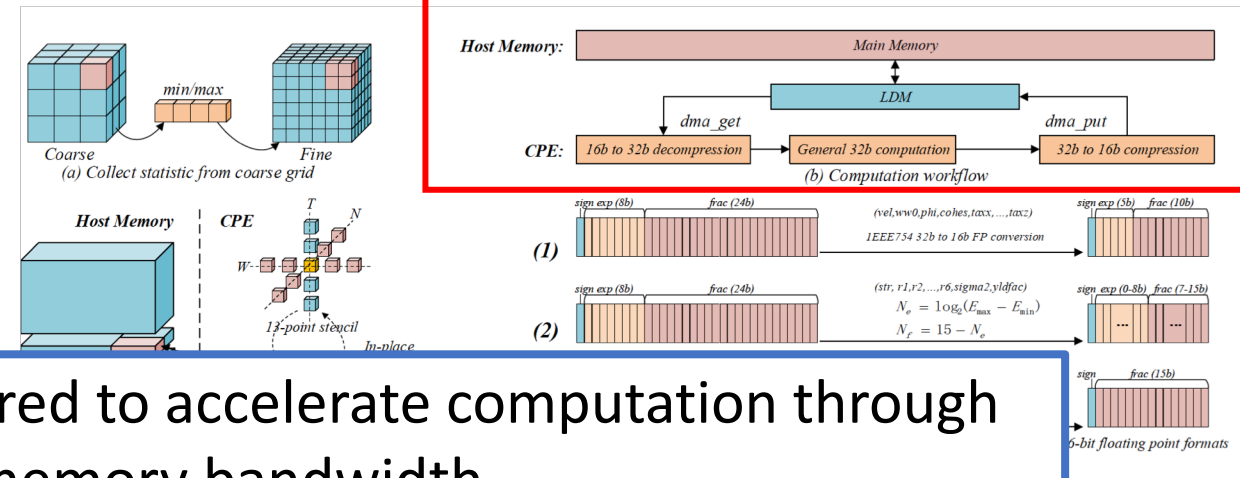
On-The-Fly Compression, explored 3 methods.

M1: Directly conversion to half precision IEEE 754 standard. However, dynamic is too large for 5 bits of exponent, for some variables.

M2: Determines the required exponent bit-width according to the recorded maximum dynamic range, and uses the remaining bits for mantissa.

M3: Normalize all the values of the same array to the range between 1 and 2, which corresponds to an exponent value of zero (small dynamic).

Seismograms comparison for 2 stations, Ninhe and Cangzhou



ANL SZ Lossy compressor today

Key features:

- **Production quality** lossy compressor for scientific data respecting user set error bounds
- **For 1D, 2D, 3D structured and unstructured datasets.** E.g. 3D simulation fields, 2D instruments data, time series.
- For **floating point** and **integer** data
- **Strict error controls** (absolute error, relative error, PSNR, *error distribution*)
- **Thorough testing procedures**, bug tracking, tests on the CORAL systems
- **Integrated in the ADIOS, HDF5 and PnetCDF I/O libraries.**
- Reader/Writer for ADIOS, HDF5, netCDF
- **Optimized compression ratios** (transforms, decompositions, multiple predictors, compression in time, lossless compression, etc.)
- **High compression/decompression speed** (*MPI + OpenMP*)

ANL SZ Application domains

Application domains:

- Fluid dynamics: climate/weather (ECP)
- Particle physics: cosmology, molecular dynamics (ECP)
- Fusion energy: plasma simulation (ECP)
- Seismology: Oil and Gas
- Quantum chemistry (ECP)
- Quantum Computing simulation
- Medical imaging
- Physics instruments: light sources (LCLS, APS) (ECP)
- Deep learning (models and training sets)

What's next 1/2

Need to improve SZ:

- **Structure (more flexible):**
 - lossy compression framework instead of lossy compressor
 - Ultimate: auto-composition/tuning of compression stage/parametrization
- **Prediction algorithms (compress more):**
 - DNN based,
 - Transform based
- **Error control (higher fidelity):**
 - Shape of the error distribution
 - Auto-tuning of the compressor error bound
- **Performance (compress faster):**
 - FPGA implementations
 - Much faster/reduced SZ for compression between memory and CPU/GPU

What's next 2/2

Need to improve fidelity assessment

Some analysis compute derivatives: e.g.
streamlines in fluid dynamics

No systematic way to assess preservation of derivatives

New metrics: Preservation of derivatives

Gradient Vector: 1st order derivatives

Hessian Matrix: 2nd order derivatives



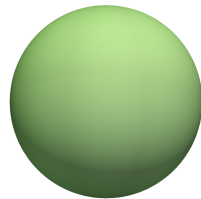
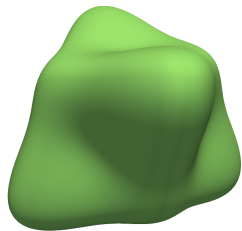
Compute vector 2-norm (1st order derivatives)
Compute Frobenius-Norm (2nd order derivatives)



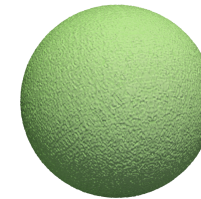
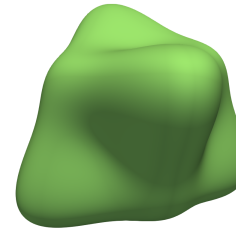
Compute PSNR (1st order derivatives)
Compute PSNR (2nd order derivatives)



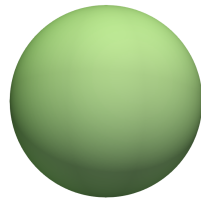
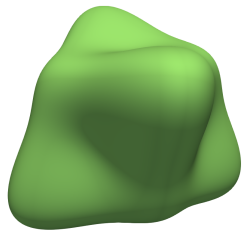
d1_M, d2_M



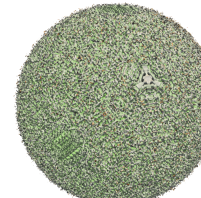
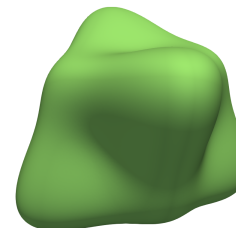
EB PSNR SSIM
0 , INF , 1
d1_M = INF , d2_M = INF



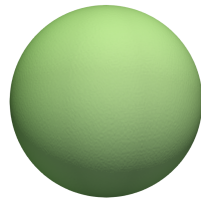
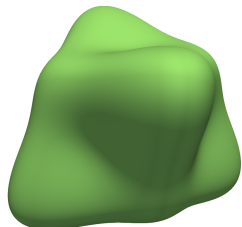
EB PSNR SSIM
1E-5 , 114.80 , 1
d1_M = 85.29 , d2_M = 53.45



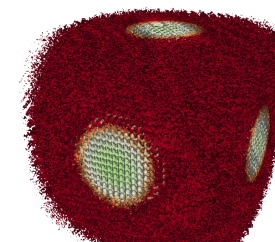
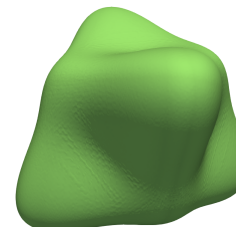
1E-7 , 154.92 , 1
d1_M = 125.32 , d2_M = 93.47



1E-4 , 94.81 , 1
d1_M = 65.29 , d2_M = 33.38



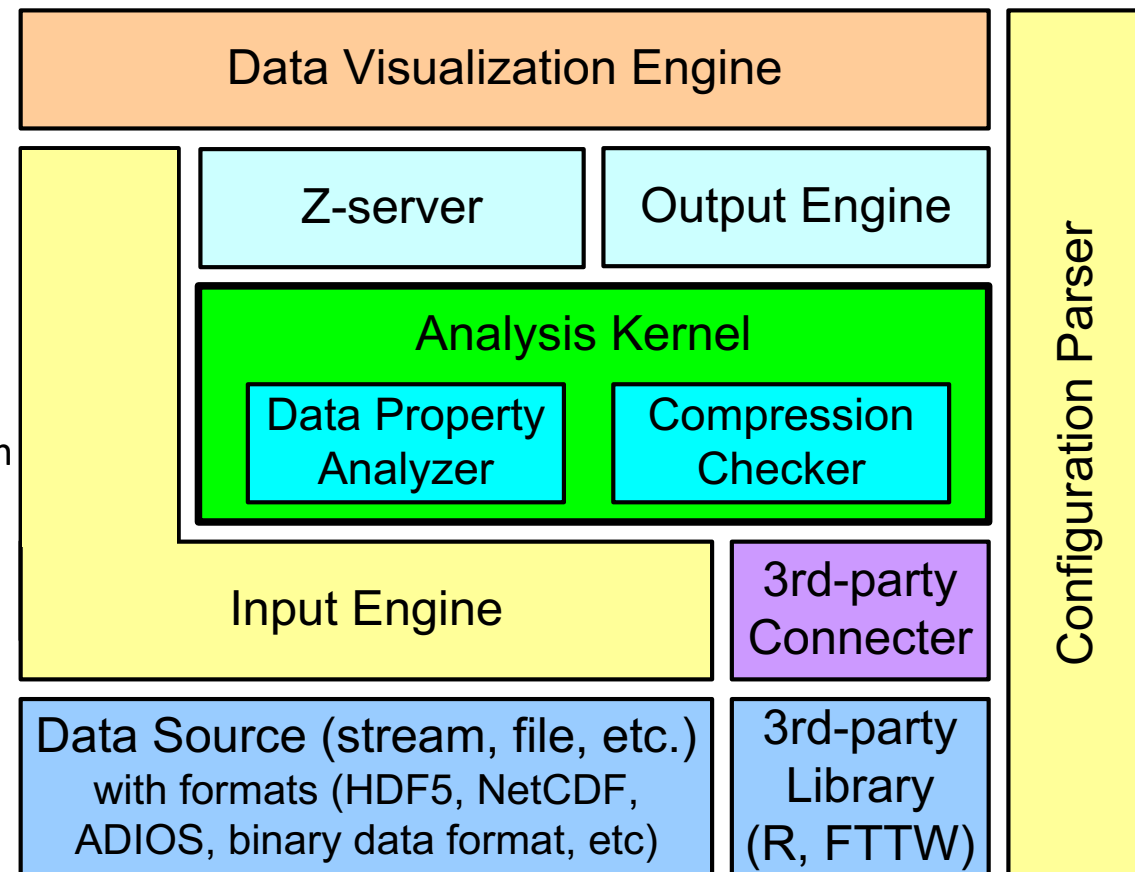
1E-6 , 134.94 , 1
d1_M = 105.37 , d2_M = 73.51



1E-3 , 75.27 , 0.999968
d1_M = 46.02 , d2_M = 12.14

Z-checker reduction error assessment tool (google z-checker github)

- Visualization Engine: local visualization, web visualization
- Analysis kernel : (1) Data property (2) Compression quality
- Input engine: Adios, HDF5, NetCDF, plain binary stream of data
- Output engine
 - Single value of a specific metric (such as entropy and compression ratio)
 - Sequence of values (such as distribution and auto-correlation of compression errors)
 - Generate report (in the form of .pdf file)
- Configuration parser
 - Switch on/off the assessment metrics
 - Specify the error bounds and comparison cases, etc.
- 3rd-party connector
 - Support R, FFTW, etc.

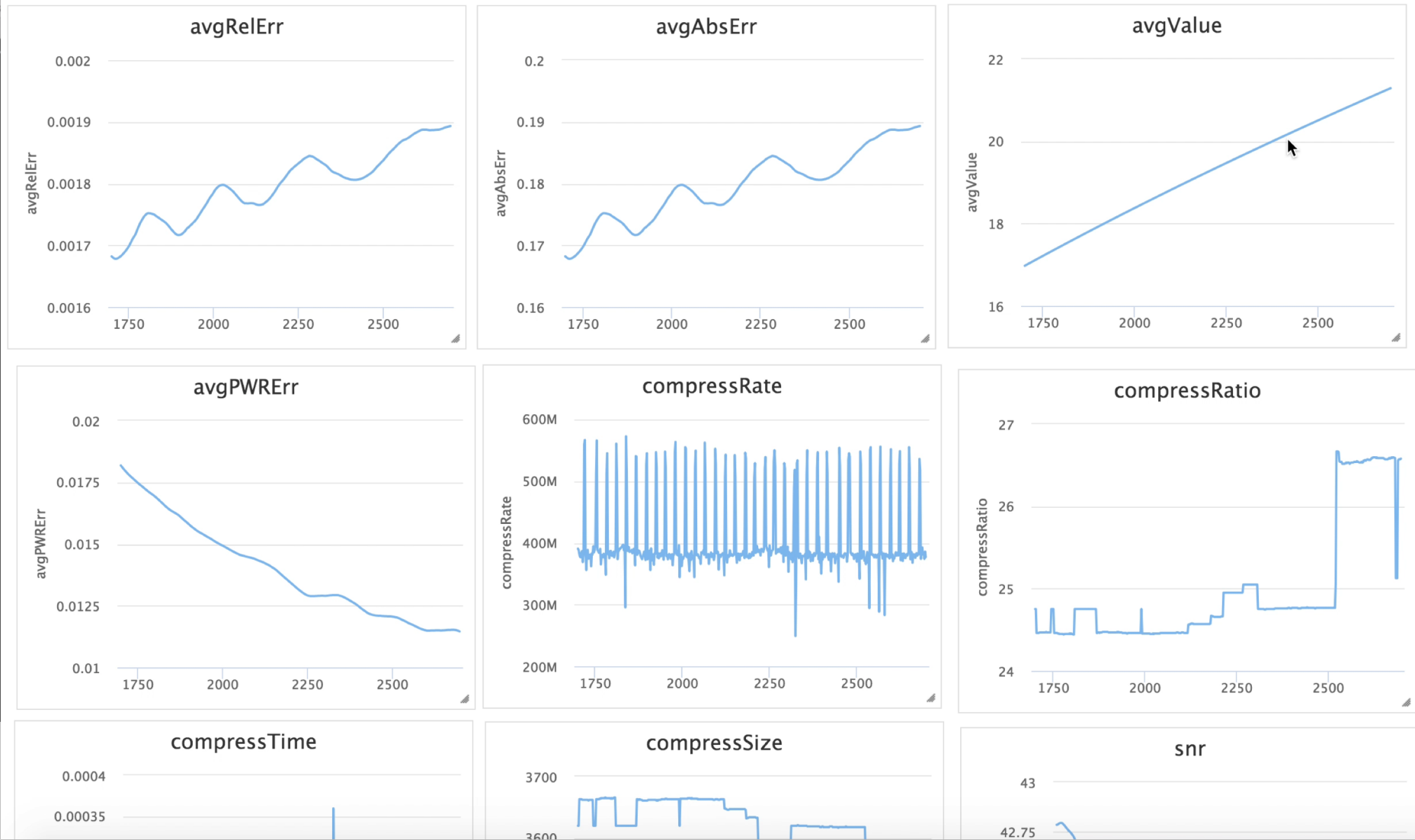


On-line Z-checker: Video

Example: heat distribution test (Jacobi algorithm simulating the heat diffusion in a flat board over time)

All metrics computed in parallel using MPI

```
bin — hguo@mcswl114 — -zs
vim ..hecker/public
→ bin git:(cmake) * mpiexec -np 4
fig zc.config temp sz
```



SDRBench: Scientific Data Reduction Benchmarks

Google (SDRBench)

Scope and objectives

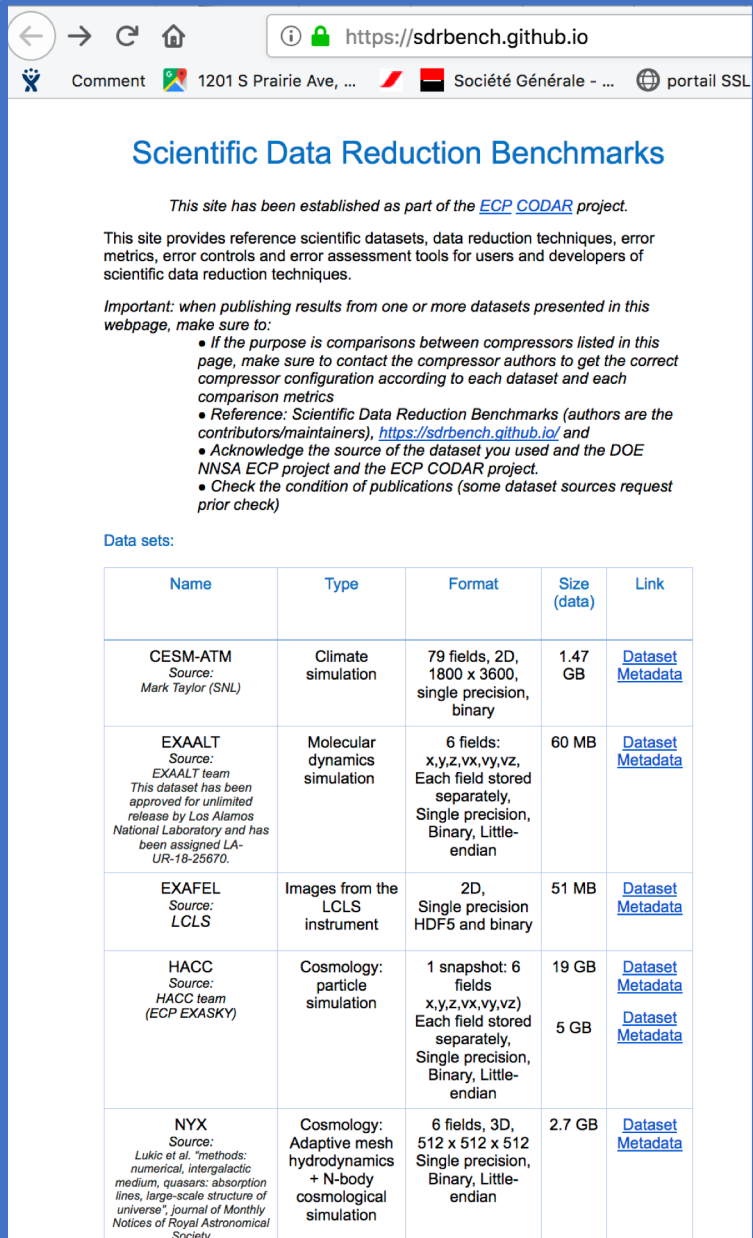
- A community repository providing reference scientific datasets, compressors (lossless and lossy), and error analysis tools
- Significance: Improve the methodology in the domain by providing reference information for scientific data compressor users and developers

Impact

Opened on July 1st 2018. Already recognized as a reference source of information for the developers of the main lossy and lossless compressors

Project accomplishment

- Collection of representative datasets from ECP and other applications via direct communication with application developers and users
- Storage of the datasets on the Petrel server at Argonne with Terabytes of storage capacity
- Fast access to the datasets using Globus and GridFTP
- Access open to public



The screenshot shows a web browser window with the URL <https://sdrbench.github.io>. The page title is "Scientific Data Reduction Benchmarks". Below the title, there is a paragraph stating: "This site has been established as part of the [ECP CODAR](#) project." followed by a description of the site's purpose. An important note follows, advising users to contact compressor authors for correct configurations and to acknowledge the source of the datasets used. Below this, a table titled "Data sets:" lists several datasets with columns for Name, Type, Format, Size (data), and Link. The table includes entries for CESM-ATM, EXAALT, EXAFEL, HACC, and NYX, each with detailed source information and links to dataset and metadata pages.

Name	Type	Format	Size (data)	Link
CESM-ATM Source: Mark Taylor (SNL)	Climate simulation	79 fields, 2D, 1800 x 3600, single precision, binary	1.47 GB	Dataset Metadata
EXAALT Source: EXAALT team <i>This dataset has been approved for unlimited release by Los Alamos National Laboratory and has been assigned LA-UR-18-25670.</i>	Molecular dynamics simulation	6 fields: x,y,z,vx,vy,vz, Each field stored separately, Single precision, Binary, Little-endian	60 MB	Dataset Metadata
EXAFEL Source: LCLS	Images from the LCLS instrument	2D, Single precision HDF5 and binary	51 MB	Dataset Metadata
HACC Source: HACC team (ECP EXASKY)	Cosmology: particle simulation	1 snapshot: 6 fields x,y,z,vx,vy,vz Each field stored separately, Single precision, Binary, Little-endian	19 GB 5 GB	Dataset Metadata Dataset Metadata
NYX Source: Lukic et al. "methods: numerical, intergalactic medium, quasars: absorption lines, large-scale structure of universe", <i>Journal of Monthly Notices of Royal Astronomical Society</i>	Cosmology: Adaptive mesh hydrodynamics + N-body cosmological simulation	6 fields, 3D, 512 x 512 x 512 Single precision, Binary, Little-endian	2.7 GB	Dataset Metadata

SZ Publications

- 27 X. Wu, S. Di, M. Dasgupta, F. Cappello, Y. Alexeev, H. Finkel, F. Chong, Full State Quantum Circuit Simulation by Using Data Compression, **ACM/IEEE SC2019**
- 26 X. Liang, S. Di, S. Li, D. Tao, B. Nicolae, Z. Chen, F. Cappello, Significantly Improving Lossy Compression Quality based on An Optimized Hybrid Prediction Model, **ACM/IEEE SC2019**
- 25 S Jin, S. Di, X. Liang, J. Tian, D. tao, F. Cappello, DeepSZ: A Novel Framework to Compress Deep Neural Networks by Using Error-Bounded Lossy Compression, **ACM HPDC 2019**
- 24 X. Zou, T. Lu, W. Xia, X. Wang, W. Zhang, S. Di, D. Tao, F. Cappello, Accelerating Relative-error Bounded Lossy Compression for HPC datasets with Precomputation-Based Mechanisms, **IEEE MSST 2019**.
- 23 F. Cappello, S. Di, S. Li, X. Liang, A. Murat Gok, D. Tao, X.-C. Wu, Y. Alexeev, F. T. Chong, Use-cases of lossy compression for floating-point data in scientific datasets, **IJHPCA**, 2019
- 22 D. Tao, S. Di, X. Liang, Z. Chen and F. Cappello, Optimizing Lossy Compression Rate-Distortion from Automatic Online Selection between SZ and ZFP, **IEEE TPDS**, 2019,
- 21 S. Di, D. Tao, Z. Chen, F. Cappello, Efficient Lossy Compression for Scientific Data based on Pointwise Relative Error Bound, **IEEE TPDS**, 2019
20. X.-C Wu, S. Di, F. Cappello, H. Finkel, Y. Alexeev, F. Chong, Memory-Efficient Quantum Circuit Simulation by Using Lossy Data Compression, PMES workshop at IEEE/ACM SC18, 2018
19. X. Liang, S. Di, S. Li, D. Tao, Z. Chen, F. Cappello, Exploring Best Lossy Compression Strategy By Combining SZ with Spatiotemporal Decimation, DRBSD-4 workshop at IEEE/ACM SC18, 2018
18. X.-C Wu, S. Di, F. Cappello, H. Finkel, Y. Alexeev, F. Chong, Amplitude-Aware Lossy Compression for Quantum Circuit Simulation, DRBSD-4 workshop at IEEE/ACM SC18, 2018
17. X. Liang, S. Di, D. Tao, S. Li, S. Li, H. Guo, Z. Chen, F. Cappello, Error-Controlled Lossy Compression Optimized for High Compression Ratios of Scientific Datasets, **IEEE BigData 2018**
16. S. Li, S. Di, X. Liang, Z. Chen, F. Cappello, Optimizing Lossy Compression with Adjacent Snapshots for N-body Simulation Data, **IEEE BigData 2018**
15. A. Murat Gok, S. Di, Y. Alexeev, D. Tao, V. Mironov, X. Liang, F. Cappello, PaSTRI: Error-Bounded Lossy Compression for Two-Electron Integrals in Quantum Chemistry, **Best paper (overall), IEEE Cluster 2018**
14. X. Liang, S. Di, D. Tao, Z. Chen, F. Cappello, An Efficient Transformation Scheme for Lossy Data Compression with Point-wise Relative Error Bound, **Best Paper (Data, Storage, and Visualization), Cluster 2018**
13. X.-Chuan Wu, S. Di, F. Cappello, H. Finkel, Y. Alexeev, F. T. Chong, Full State Quantum Circuits Simulation by Using Lossy Data Compression, **SC18 Poster**, 2018
12. S. Li, D. Sheng, X Liang, Z. Chen, F. Cappello, Improving Error-bounded Lossy Compression for Cosmological N-body Simulation, **SC18 Poster**, 2018
11. D. Tao, S. Di, X. Liang, Z. Chen, Franck Cappello, Restarting iterative methods from lossy checkpoints, **ACM HPDC 2018**
10. D. Tao, S. Di, X. Liang, Z. Chen, F. Cappello, Fixed-PSNR Lossy Compression for Scientific Data, **IEEE Cluster 2018**, short paper
9. J. Calhoun, F. Cappello, L. N. Olson, M. Snir, and W. Gropp, Exploring the Feasibility of Lossy Compression for PDE Simulations, International Journal of High Performance Computing Applications (**IJHPCA**), **2019 (2018)**
8. D. Tao, S. Di, Z. Chen and F. Cappello, In-Depth Exploration of Single-Snapshot Lossy Compression Techniques for N-Body Simulations, **IEEE Bigdata 2017**
- 7 D. Tao, S. Di, Z. Chen and F. Cappello. Significantly Improving Lossy Compression for Scientific Data Sets Based on Multidimensional Prediction and Error-Controlled Quantization. **IEEE IPDPS**, May 2017
6. D. Tao, S. Di, Z. Chen and F. Cappello. Exploration of Pattern-Matching Techniques for Lossy Compression on Cosmology Simulation Data Sets. 1st International Workshop on Data Reduction for Big Scientific Data as part of ISC2017, Frankfurt DE, June 2017
5. A. Murat Gok, D. Tao, S. Di, V. Mironov, Y. Alexeev and F. Cappello. PaSTRI: A Novel Data Compression Algorithm for Two-Electron Integrals in Quantum Chemistry. Poster at **IEEE/ACM SC17**, Denver, CO US, November 2017
4. S. Di, D. Tao, Z. Chen, F. Cappello, Towards Efficient Error-controlled Lossy Compression for Scientific Data, [Poster], Greater Chicago Area Systems Research Workshop, 2017
3. D. Tao, S. Di, H. Guo, F. Cappello, Z-checker: A Framework for Assessing Lossy Compression of Scientific Data, International Journal of High Performance Computing Applications, **IJHPCA, 2019 (2017)**
2. S. Di, F. Cappello, Optimizing Error-Bounded Lossy Compression for Hard-to-Compress HPC Data, Submitted to IEEE Transactions on Parallel and Distributed Computing **IEEE TPDS, 2017**
1. S. Di, F. Cappello, Fast Error-bounded Lossy HPC Data Compression with SZ, **IEEE IPDPS 2016**

Thanks

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.

