Lossy compression of floating-point data

iCAS 2017

Peter Lindstrom

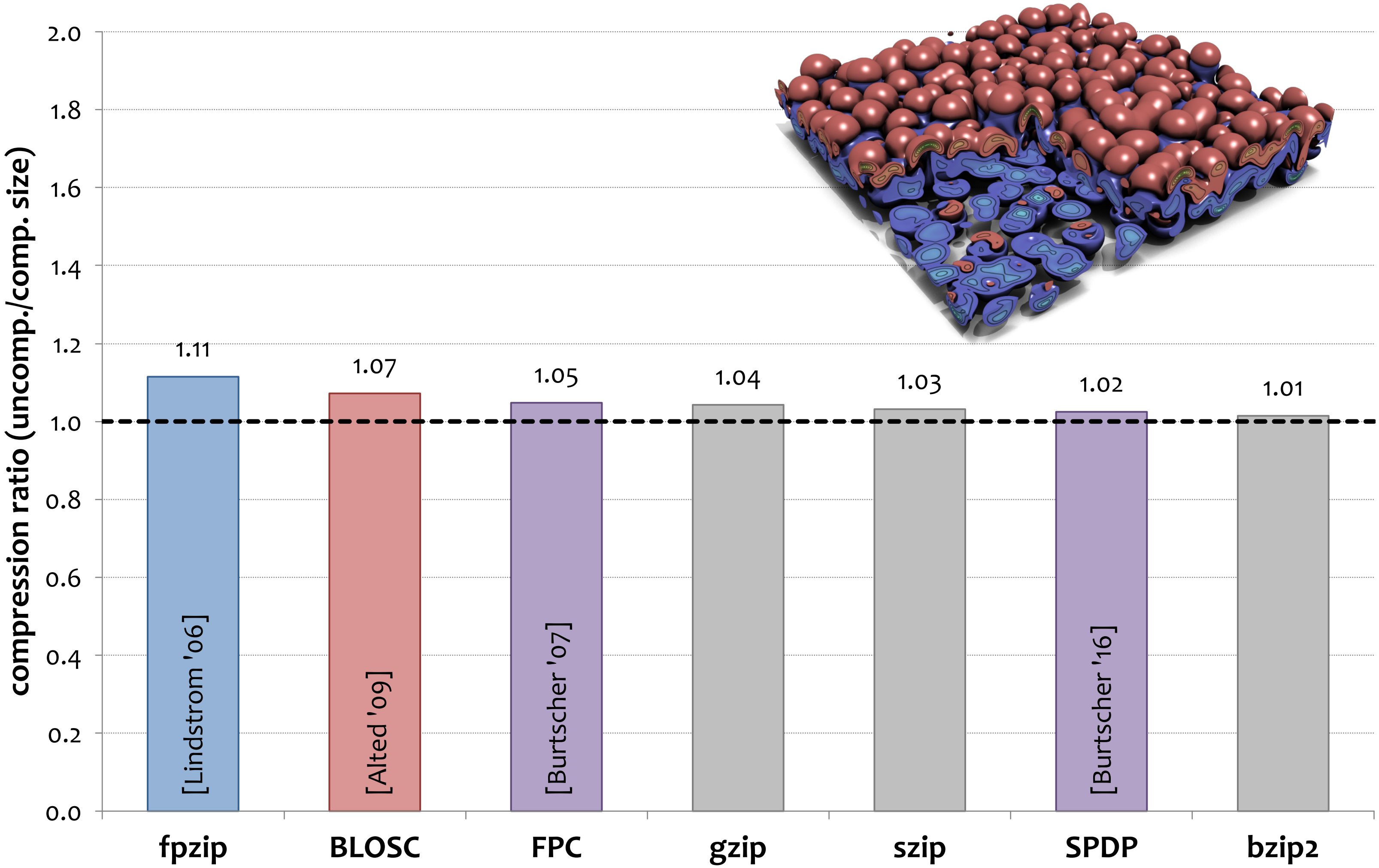
September 12, 2017

LLNL-PRES-739189

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

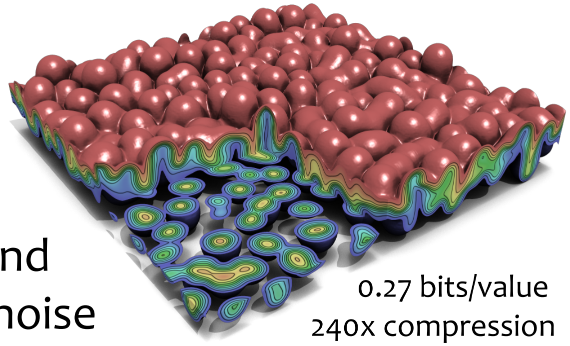
Lawrence Livermore National Laboratory

Numerical data is challenging to compress losslessly



Lossy compression enables greater reduction, but is often met with skepticism by scientists

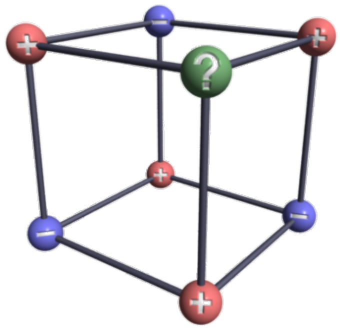
- Large improvements in compression are possible by allowing even small errors
  - Simulation often computes on meaningless bits
    - Round-off, truncation, iteration, model errors abound
    - Last few floating-point bits are effectively random noise



- Still, lossy compression often makes scientists nervous
  - Even though lossy **data reduction** is ubiquitous
    - **Decimation** in space and/or time (e.g. store every 100 time steps)
    - **Averaging** (hourly vs. daily vs. monthly averages)
    - **Truncation** to single precision (e.g. for history files)
  - Moreover, most compressors support **error tolerances**

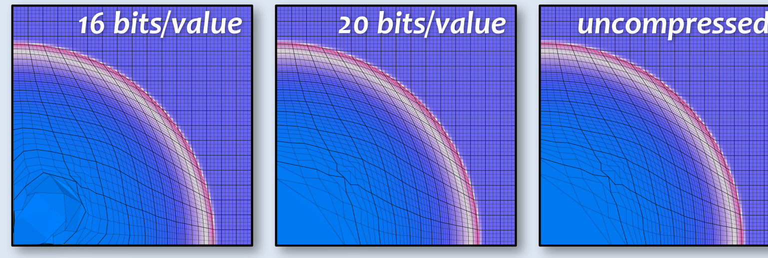
LLNL has developed two high-speed floating-point compressors

- fpzip: lossless floating-point compressor
  - — Lossy compression via truncation to desired precision
    - fpzip loslessly compresses pre-truncated floats
    - Similar to casting double to single precision
    - Truncation enables **relative-error bound**
  - — Streaming compressor integrates well with I/O

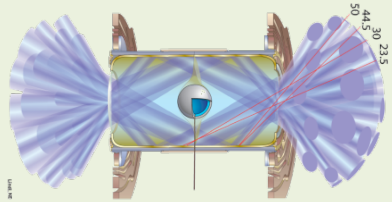


Lossy compression may also be used to reduce the memory footprint of simulation state

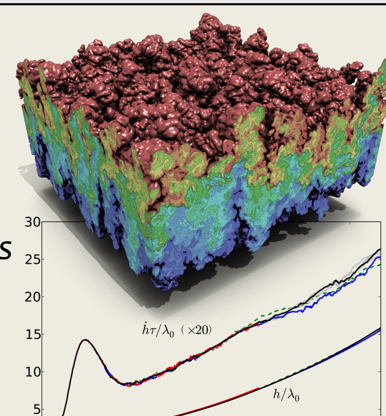
LULESH: Lagrangian shock hydrodynamics
- QoI: *radial shock position*
- 25 state variables compressed over 2,100 time steps
- At **4x compression**, relative **error < 0.06%**



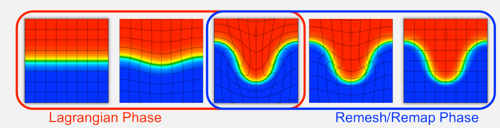
pf3D: Laser-plasma multi-physics
- QoI: *backscattered laser energy*
- At **4x compression**, relative **error < 0.1%**

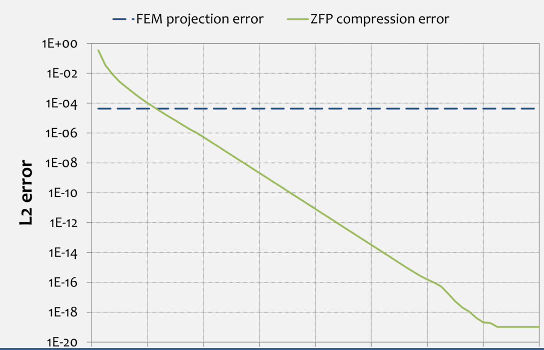


Miranda: High-order Eulerian hydrodynamics
- QoI: *Rayleigh-Taylor mixing layer thickness*
- 10,000 time steps
- At **4x compression**, relative **error < 0.2%**



MFEM: Cubic finite elements
- QoI: *function approximation*
- **6x compression** with ZFP **error < 0.7%** relative to FEM error

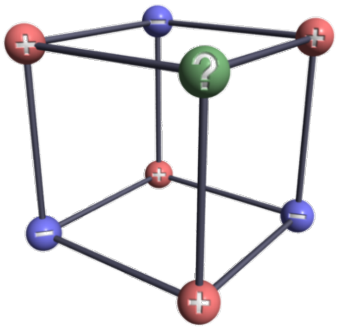




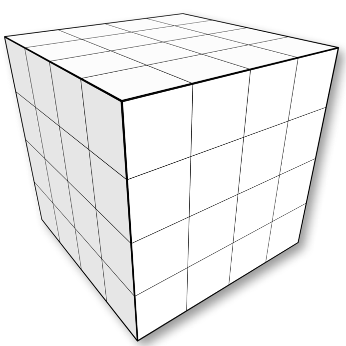
4x lossy compression of state is viable, but streaming compression *increases* data movement

LLNL has developed two high-speed floating-point compressors

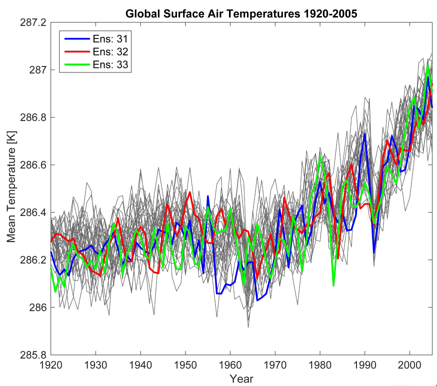
- fpzip: lossless floating-point compressor
  - Lossy compression via truncation to desired precision
    - fpzip loslessly compresses pre-truncated floats
    - Similar to casting double to single precision
    - Truncation enables **relative-error bound**
  - Streaming compressor integrates well with I/O

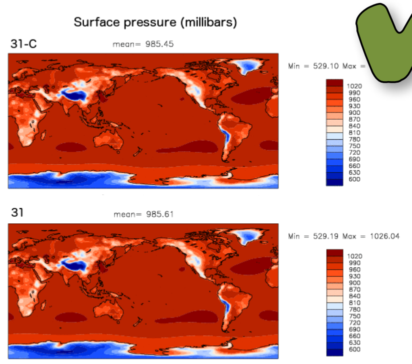


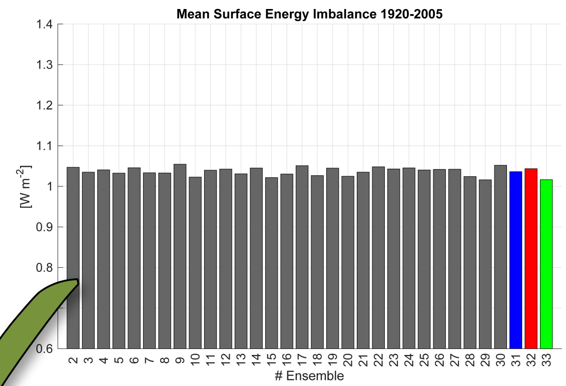
- ZFP: random-accessible compressed arrays
  - Partition *d*-dim. array into blocks of 4^d values
    - Blocks are independently (de)compressed on demand
  - Can truncate compressed bit stream anywhere
    - Variable rate enables **absolute-error bound**
    - Fixed rate enables read/write **random access** to blocks
  - C++ operator overloading facilitates app. integration
    - double a[n] ⇔ std::vector<double> a(n) ⇔ zfp::array<double> a(n, **precision**)

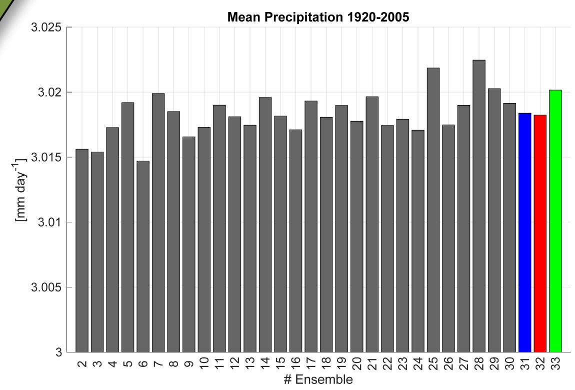


fpzip systematic rounding toward zero leads to occasional issues in climate data analysis

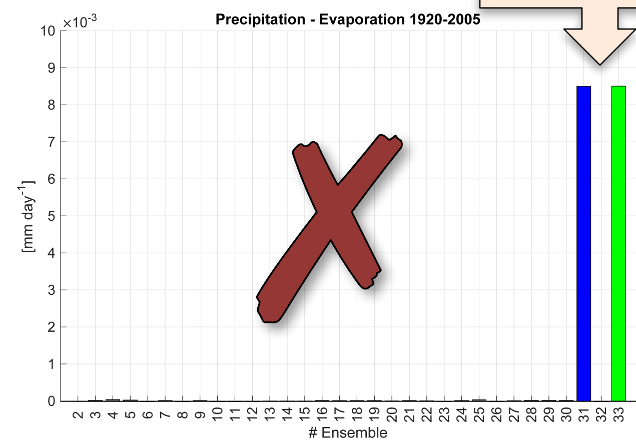




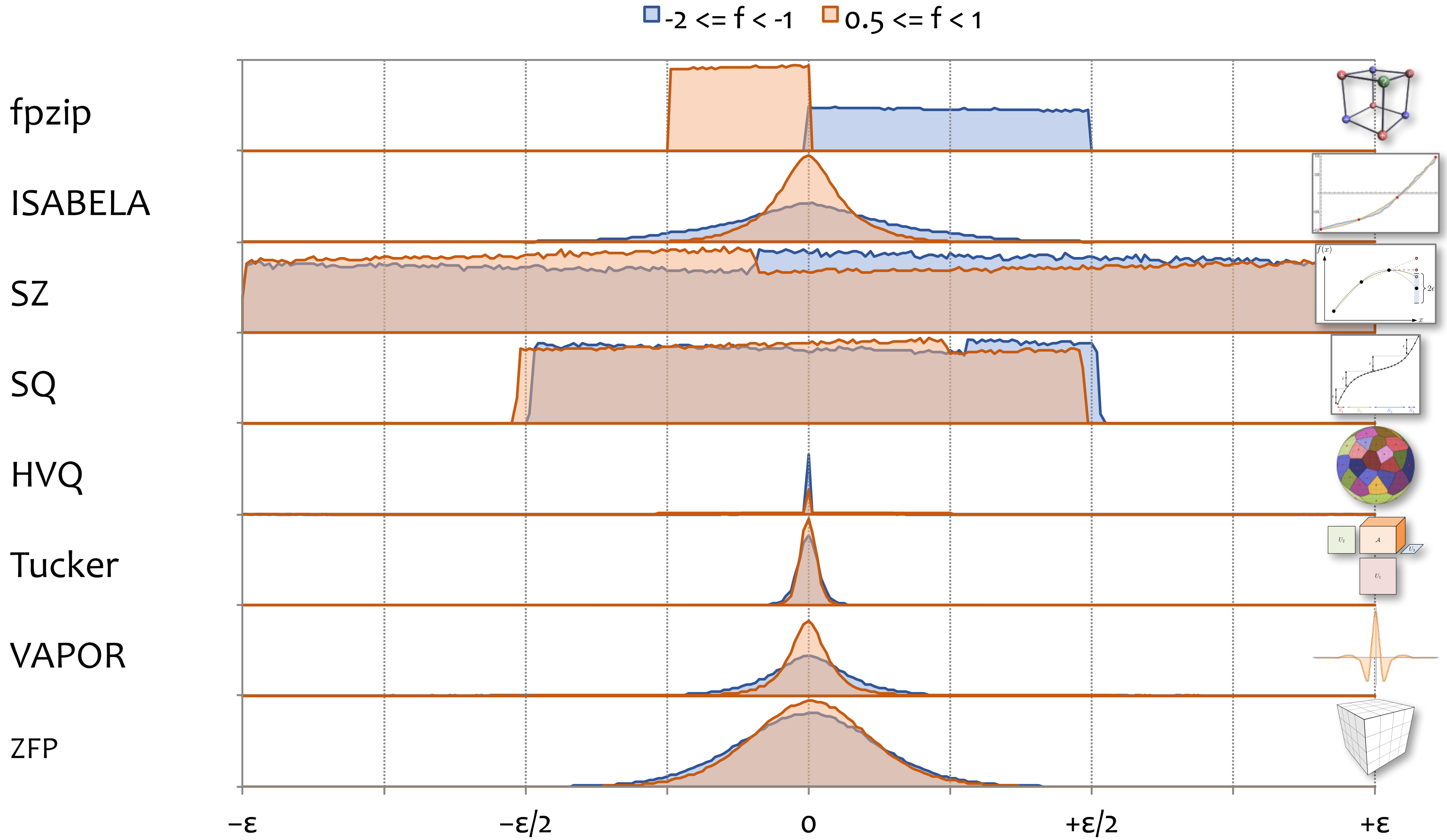




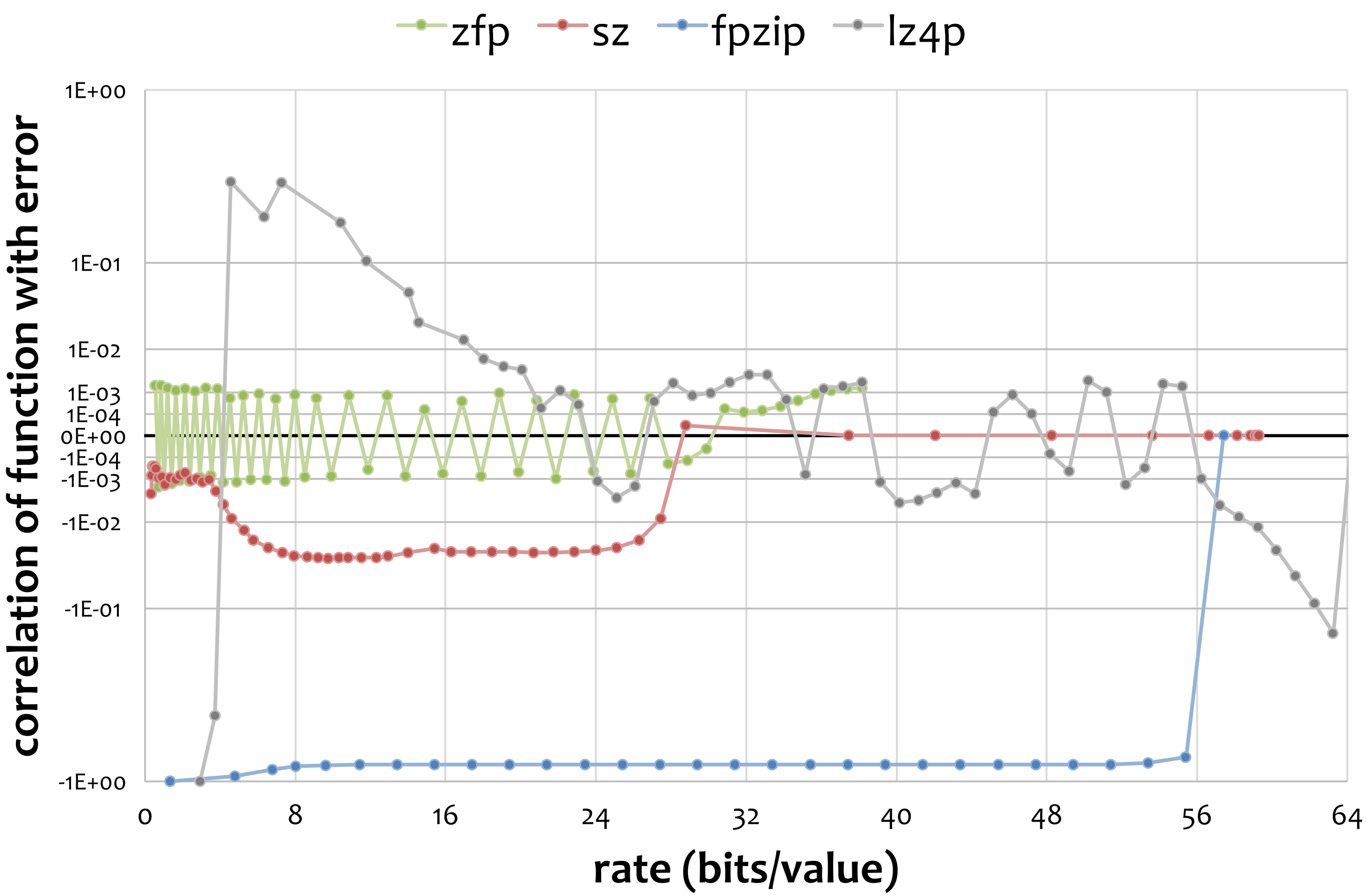
#31, #33 compressed ensemble members



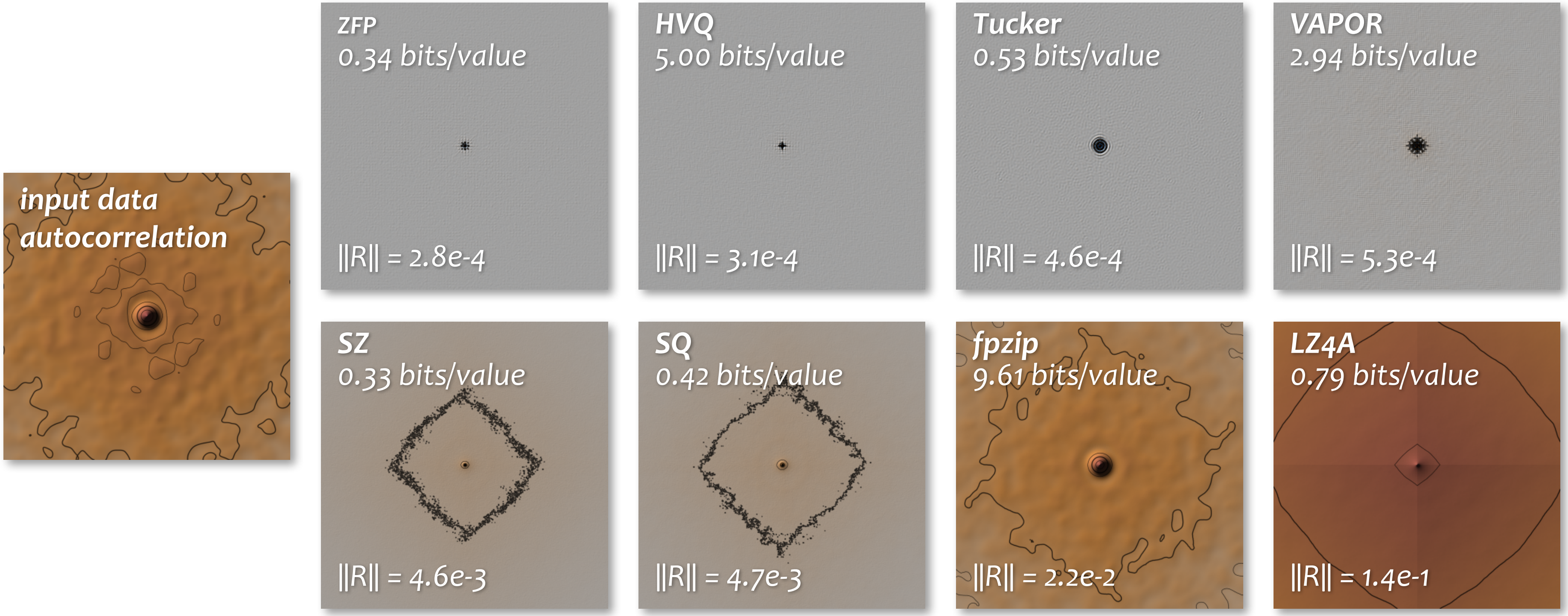
fpzip error distribution is highly biased; ZFP distribution is normal (central value thm.)



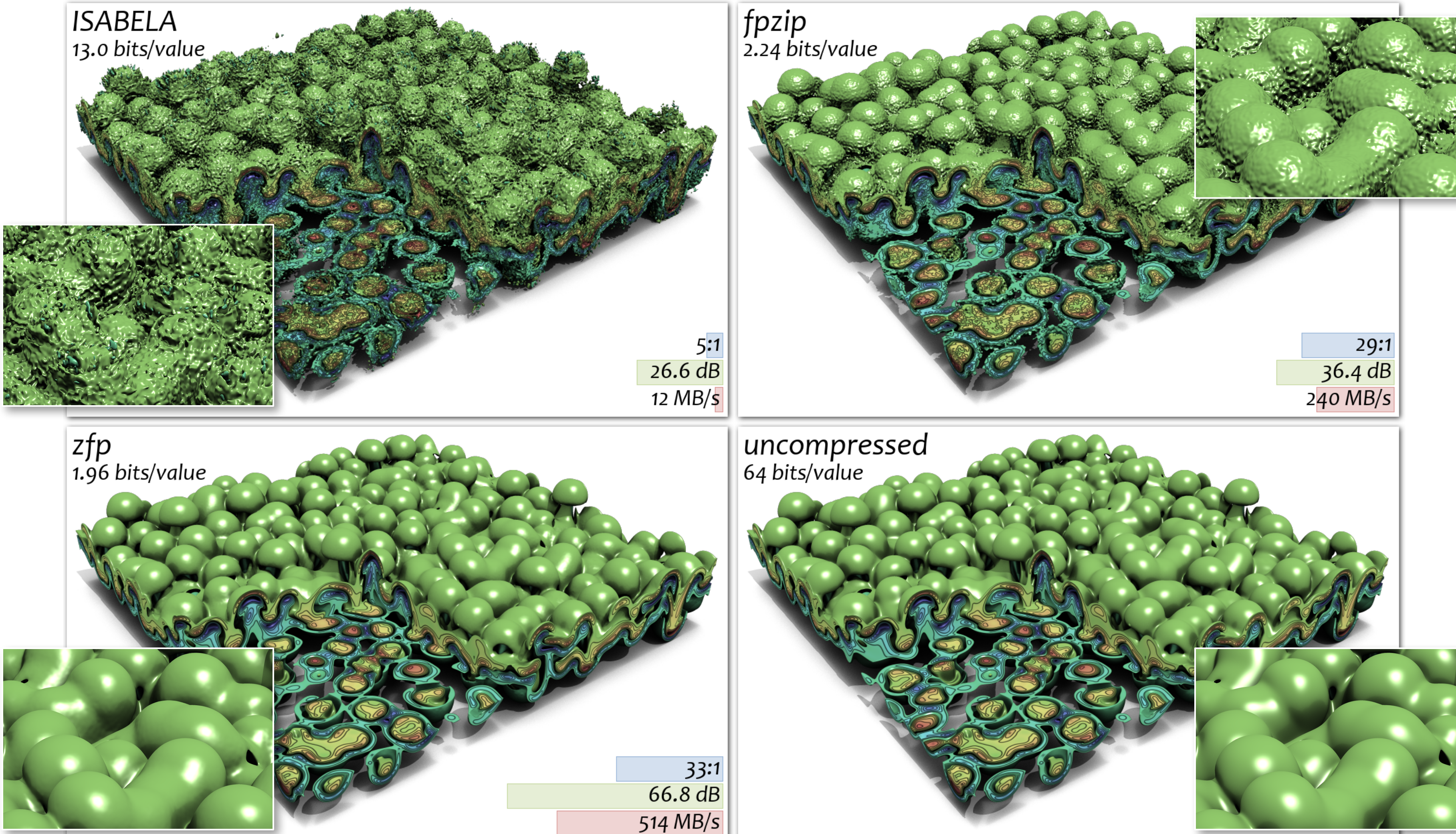
ZFP and SZ decorrelate error with function (note nonlinear vertical axis)



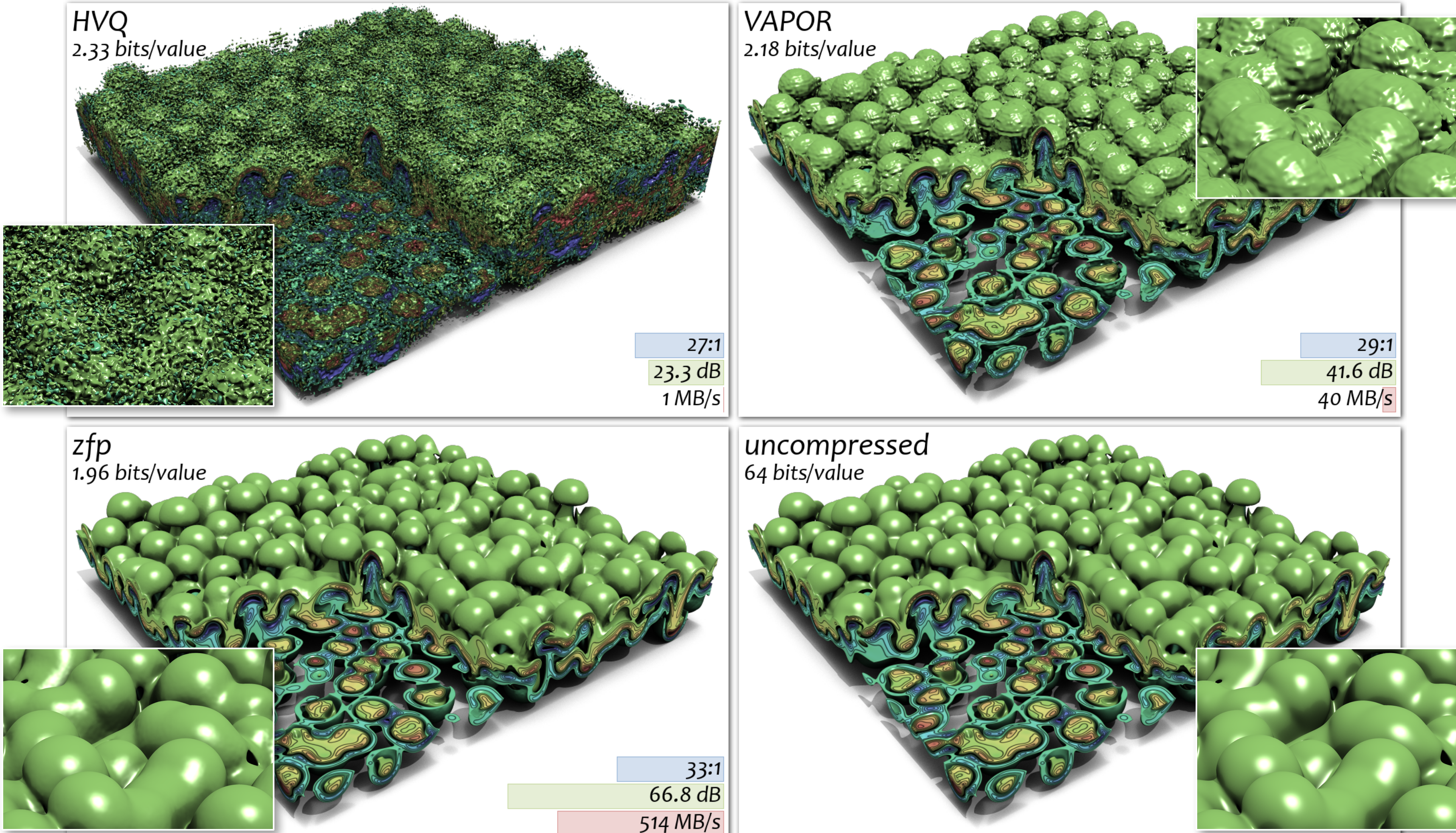
Some compressors yield autocorrelated errors



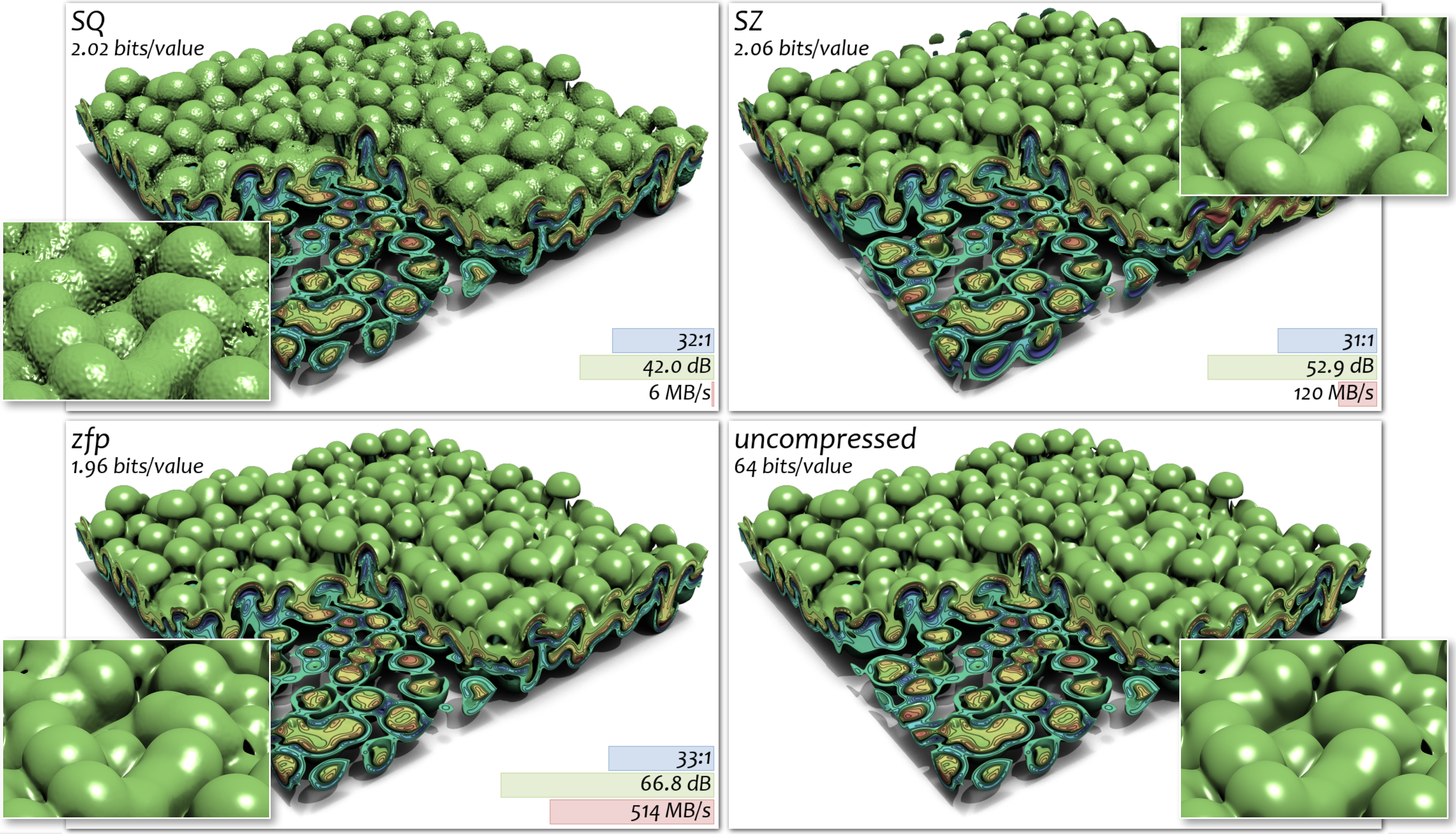
Some compressors show artifacts in derivative computations (velocity divergence)



Some compressors show artifacts in derivative computations (velocity divergence)



Some compressors show artifacts in derivative computations (velocity divergence)

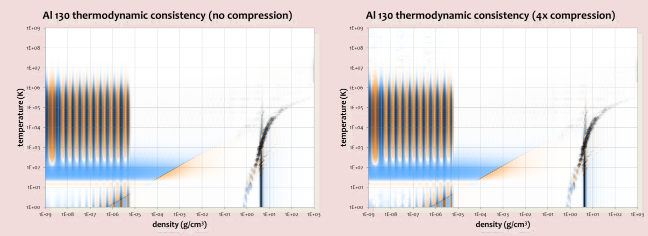


ZFP is being used in production in numerous petascale applications

With P. Sterne @ LLNL

LEOS: Livermore Equation of State library

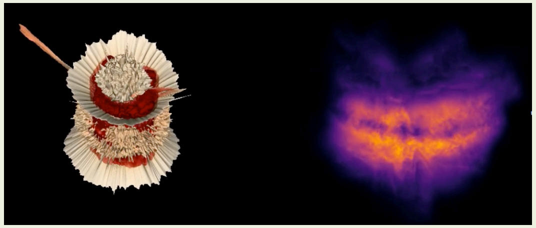
- EOS material tables consume a large fraction of memory
- ZFP arrays enable random access to compressed tables
- **4x compression** ensures thermodynamic consistency



With S. Langer @ LLNL

HYDRA: Simulated ICF X-rays

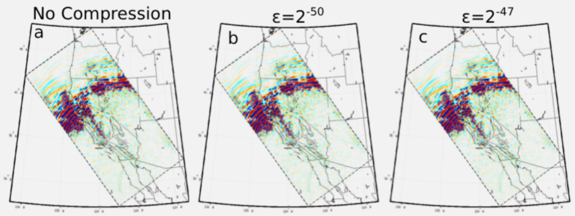
- Trinity runs generated **4 PB** of uncompressed data
- 15 months to transfer to LLNL; 2 months of disk space available
- **10x compression** allowed for successful data transfer



With P. Chen @ UW

AWP-ODC: Full-3D Seismic Tomography

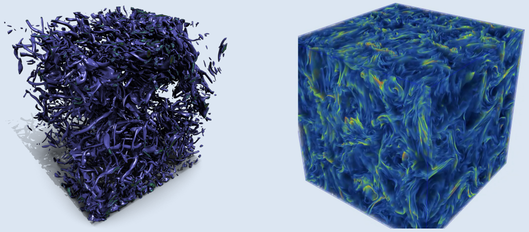
- Adjoint-wavefield competitor uses 200 GB
- Scattering-integral formulation uses **1.8 PB**
- **40x compression, 6x less I/O time**
- 10^{-6} compression error < 10^{-2} observation noise



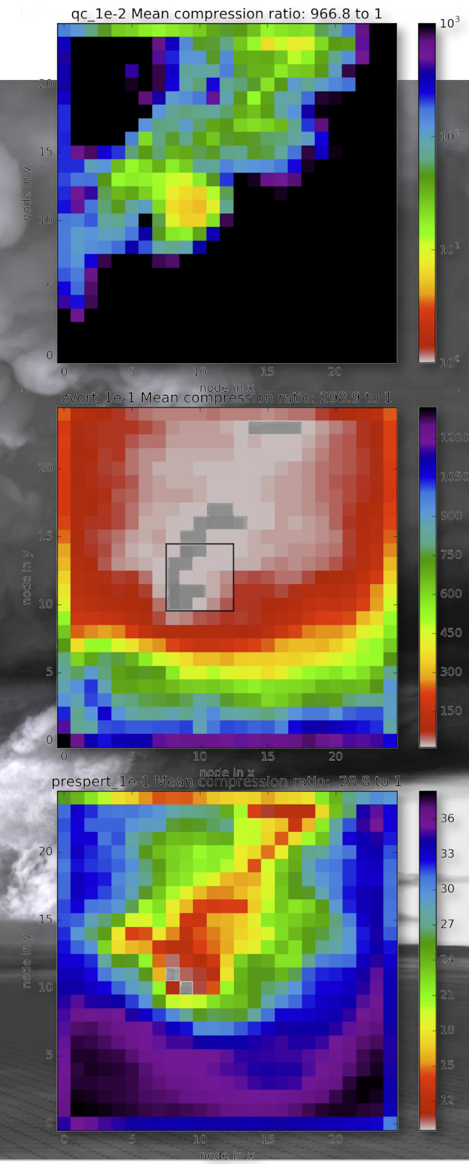
With R. Burns @ JHU

JHTDB: Johns Hopkins Turbulence Database

- Trinity runs use $8K^3$ grids = **2 TB/field/time step**
- Without compression, only selected features saved
- ZFP allows retaining full fields
- Compressed data server being considered

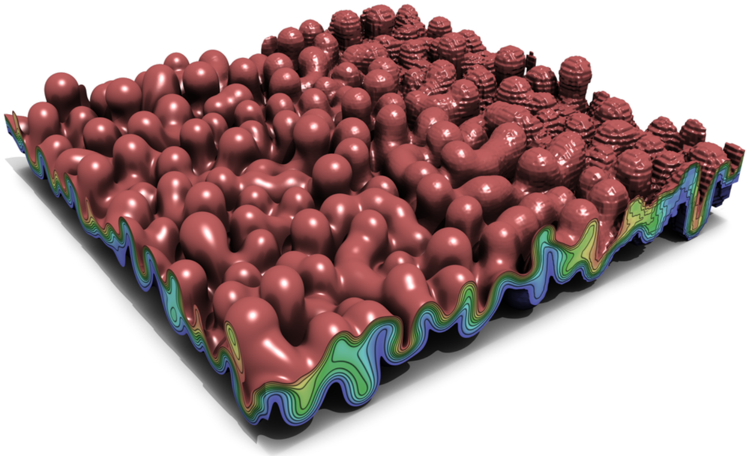


ZFP reduces I/O by 30x on average in CM1 tornado simulation [work done by Leigh Orf, UW-Madison]



fpzip and ZFP are publicly available

- fpzip: casc.llnl.gov/fpzip
- ZFP: github.com/LLNL/zfp
  - BSD licensed open source
  - Development is funded by ECP
- Several ZFP I/O plugins are available
  - netCDF is in the works



HDF — github.com/LLNL/H5Z-ZFP

ADIOS — github.com/suchyta1/AtoZ

VTK — gitlab.kitware.com/third-party/zfp

References

[fpzip] Lindstrom & Isenburg, "Fast and efficient compression of floating-point data," 2006

[HVQ] Schneider & Westermann, "Compression domain volume rendering," 2003

[ISABELA] Lakshminarasimhan et al., "ISABELA for effective in situ compression of scientific data," 2013

[LZ4A, LZ4P] Kunkel et al., "Decoupling the selection of compression algorithms from quality constraints with SCIL," 2017

[SQ] Iverson et al., "Fast and effective lossy compression algorithms for scientific datasets," 2012

[SZ] Di & Cappello, "Fast error-bounded lossy HPC data compression with SZ," 2016

[Tucker] Ballester & Pajarola, "Lossy volume compression using Tucker truncation and thresholding," 2016

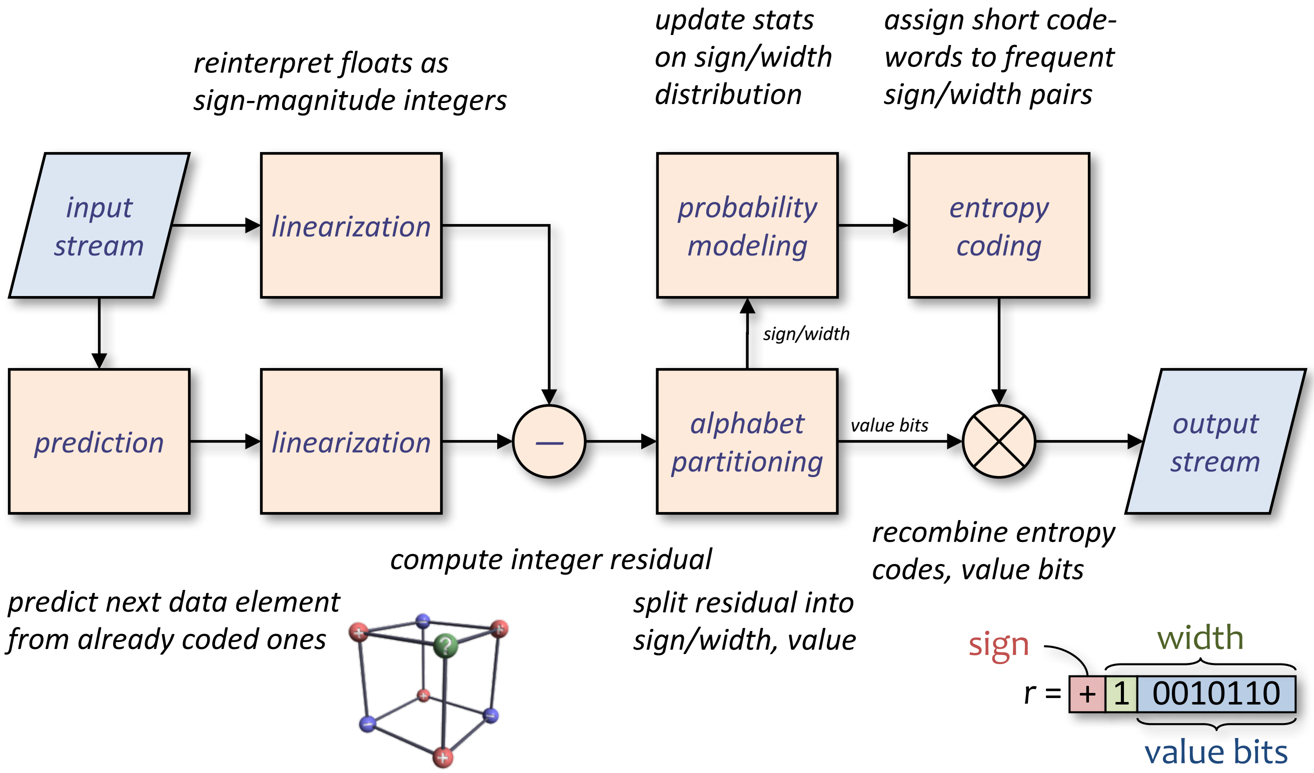
[VAPOR] Clyne et al., "Interactive desktop analysis of high resolution simulations," 2007

[ZFP] Lindstrom, "Fixed-rate compressed floating-point arrays," 2014

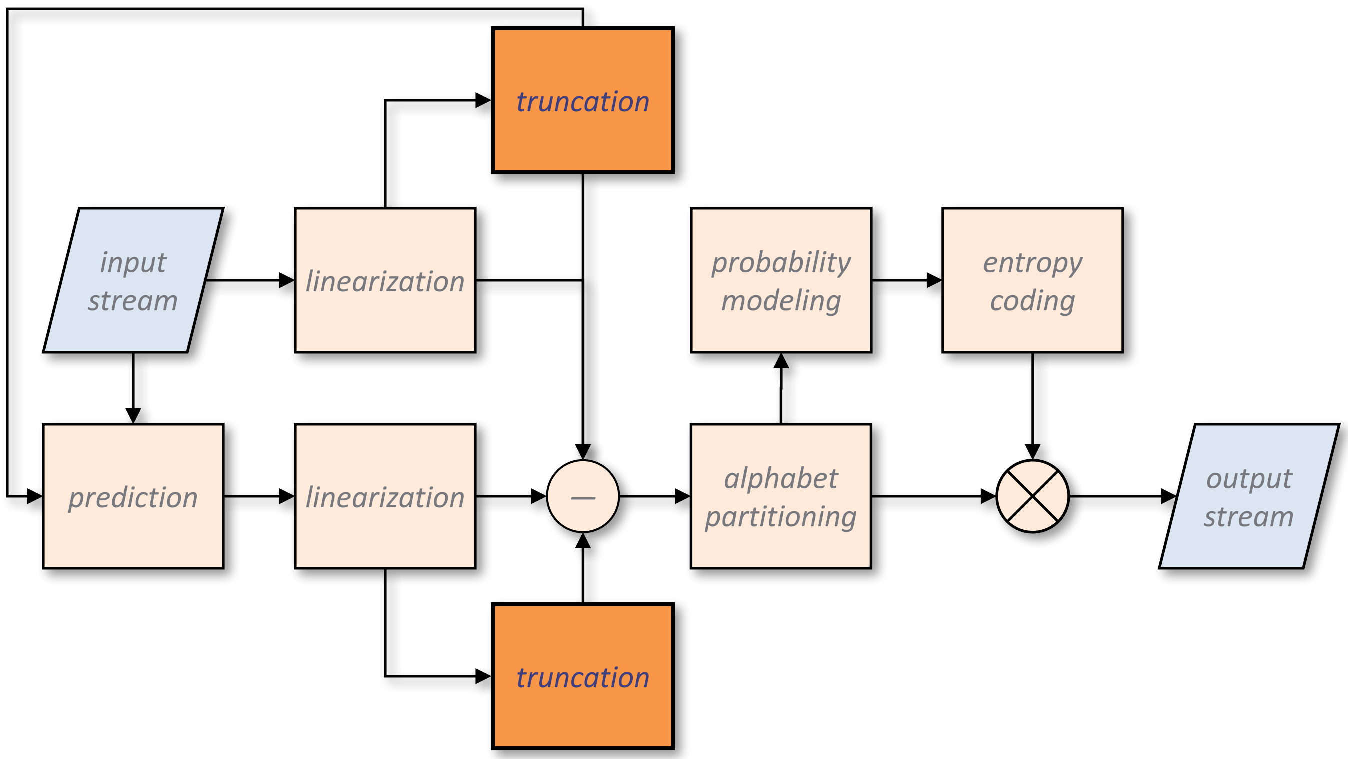
Lawrence Livermore National Laboratory

Additional material

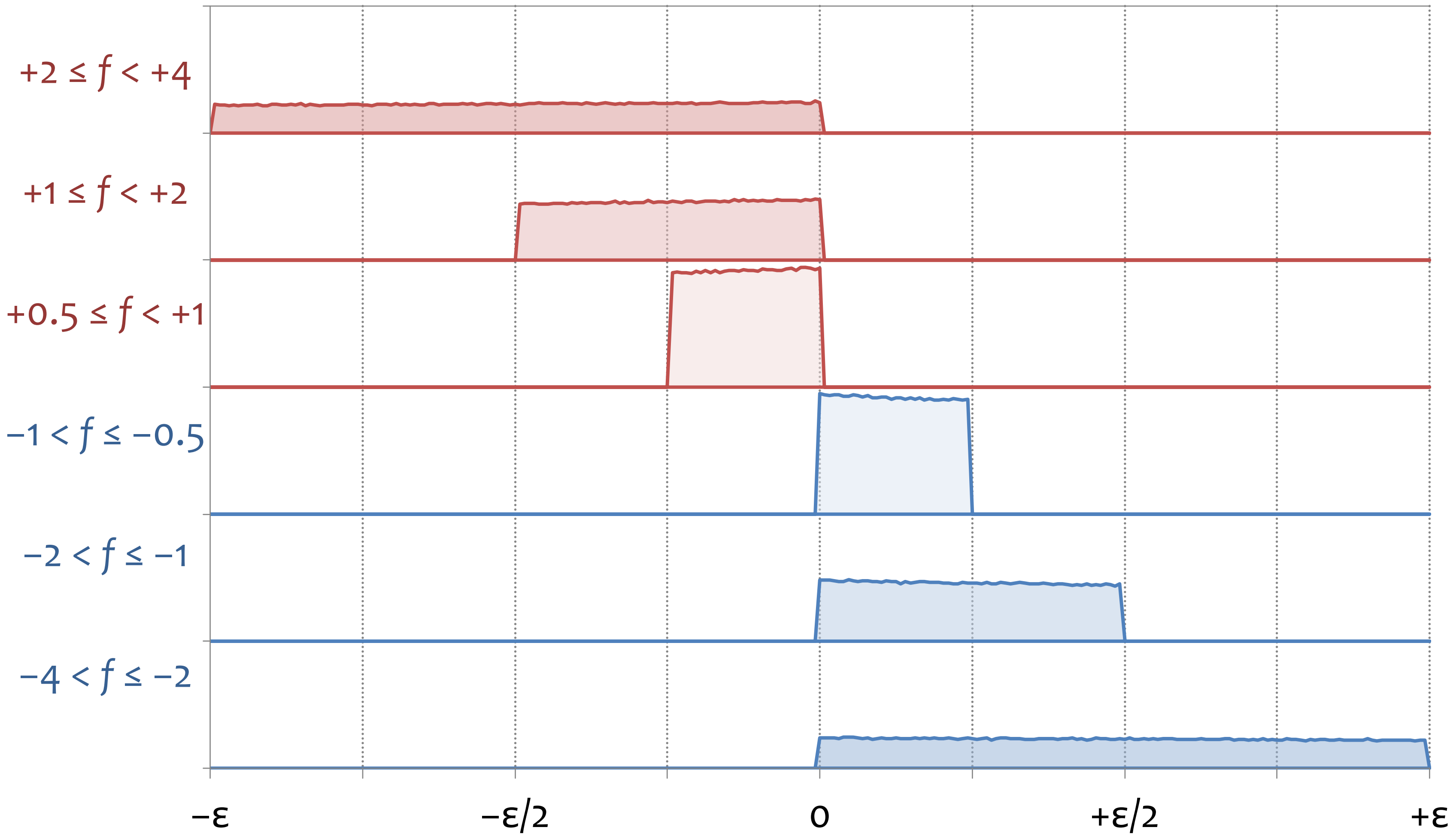
fpzip: Lossless mode combines multi-dimensional prediction with entropy coding



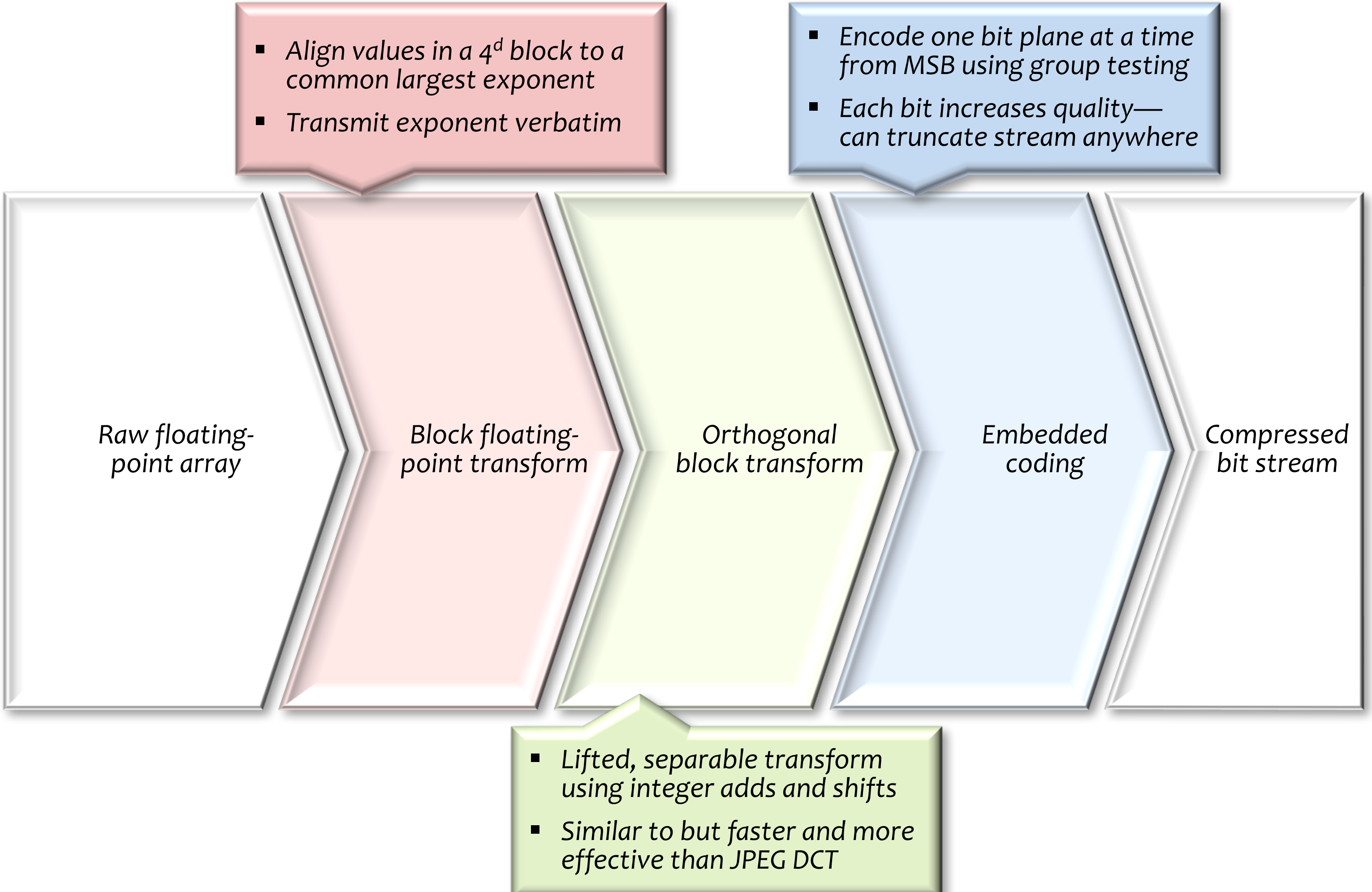
fpzip: Lossy mode truncates (zeros) least significant bits, then compresses losslessly



fpzip error distribution is dependent on function value *f* and is highly biased



ZFP: Compressed floating-point arrays that support random access on demand



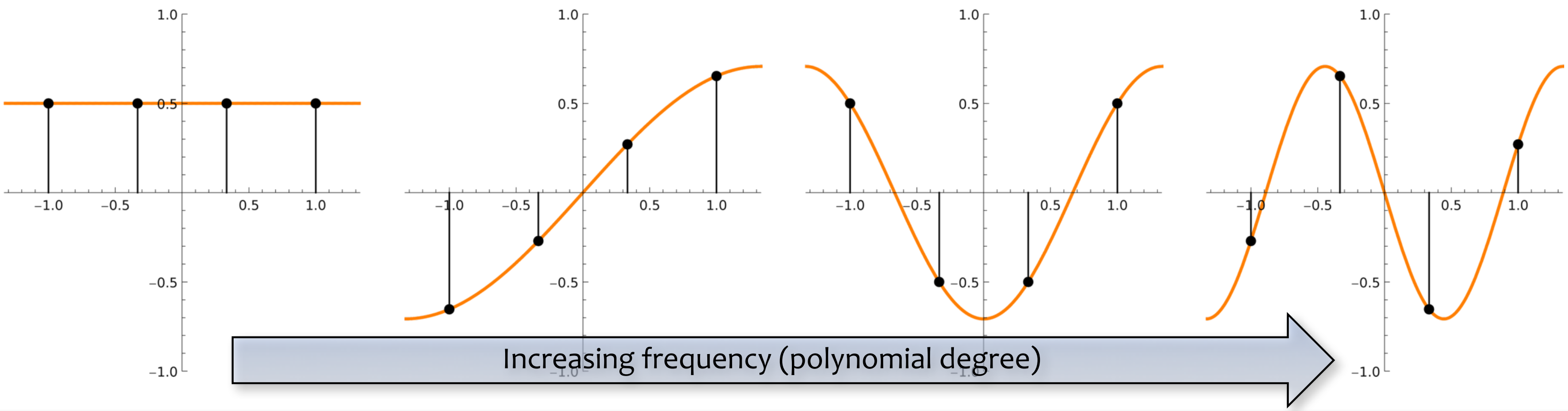
ZFP decorrelates d-dimensional block of 4^d values using an orthogonal transform

$$\underbrace{\begin{pmatrix} \hat{f}_1 \\ \hat{f}_2 \\ \hat{f}_3 \\ \hat{f}_4 \end{pmatrix}}_{\textit{coefficients}} = \frac{1}{2} \underbrace{\begin{pmatrix} 1 & 1 & 1 & 1 \\ c & s & -s & -c \\ 1 & -1 & -1 & 1 \\ s & -c & c & -s \end{pmatrix}}_{\textit{orthogonal transform}} \underbrace{\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix}}_{\textit{block}}$$

Free parameter t

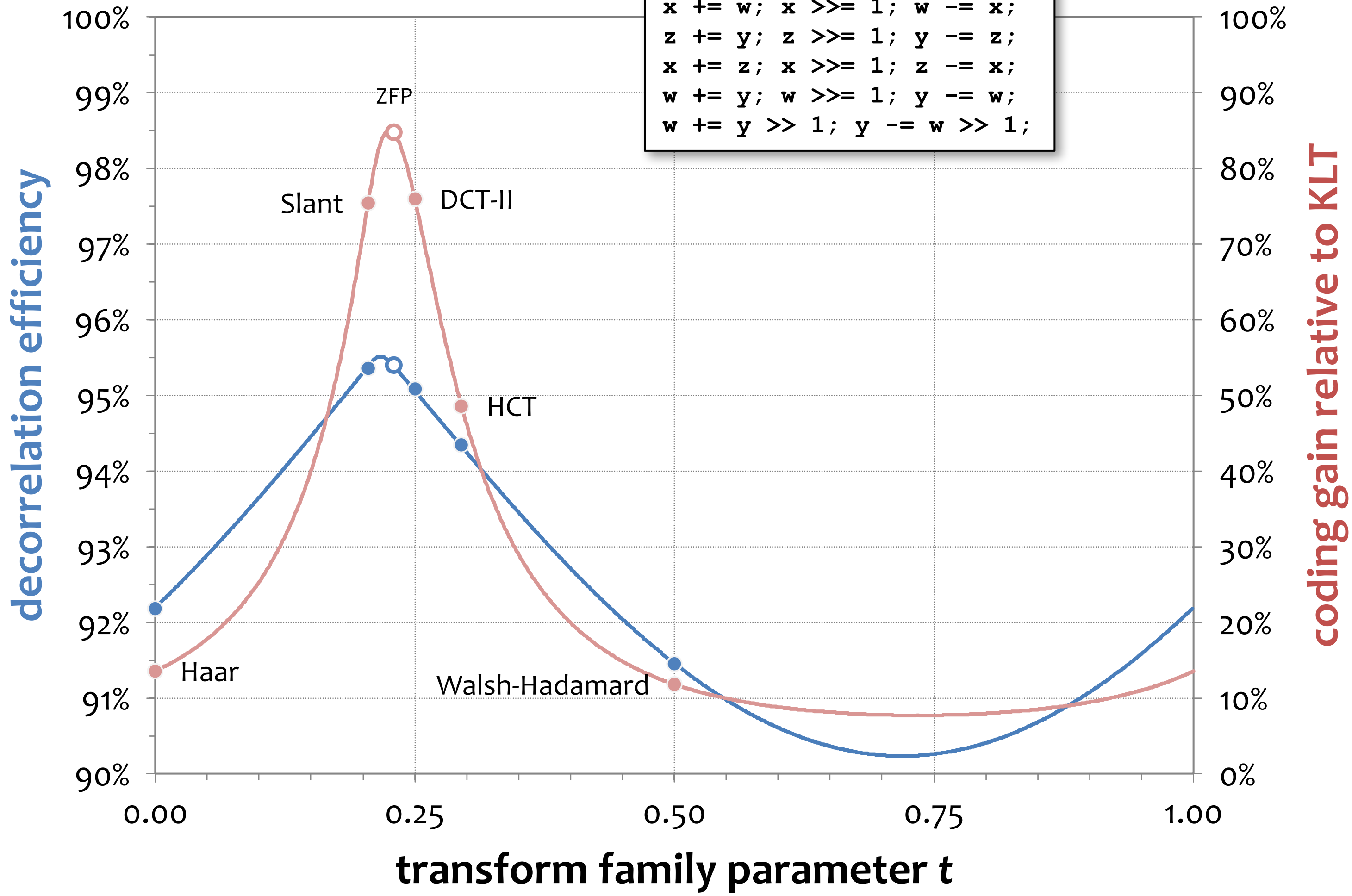
$$s = \sqrt{2}\sin\tfrac{\pi}{2}t \qquad c = \sqrt{2}\cos\tfrac{\pi}{2}t$$

Basis functions for $t = 1/4$

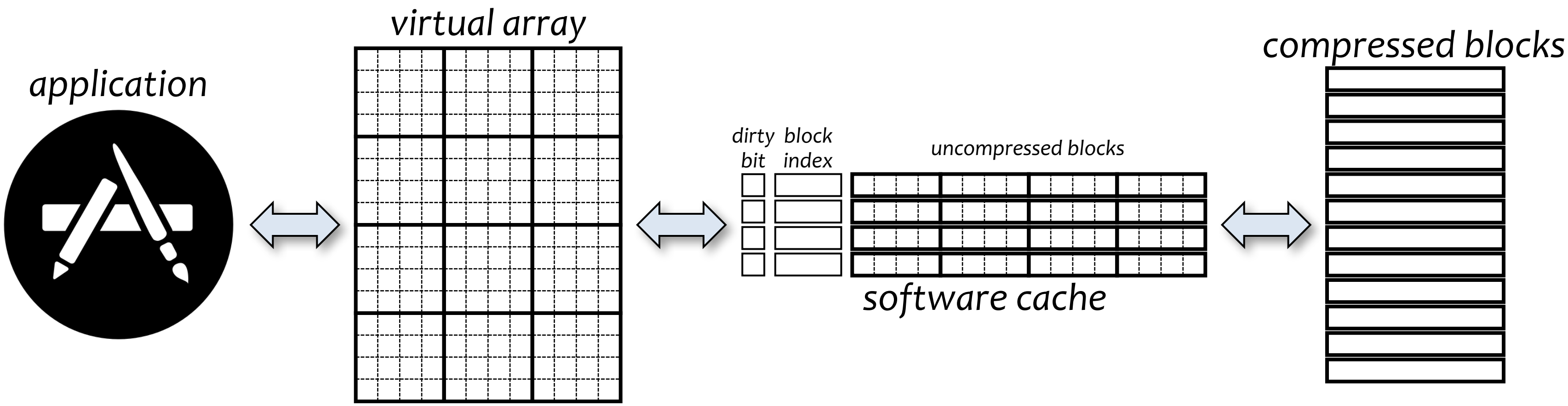


ZFP's integer transform is efficient, effective, and well-suited for h/w implementation

```
x += w; x >>= 1; w -= x;
z += y; z >>= 1; y -= z;
x += z; x >>= 1; z -= x;
w += y; w >>= 1; y -= w;
w += y >> 1; y -= w >> 1;
```

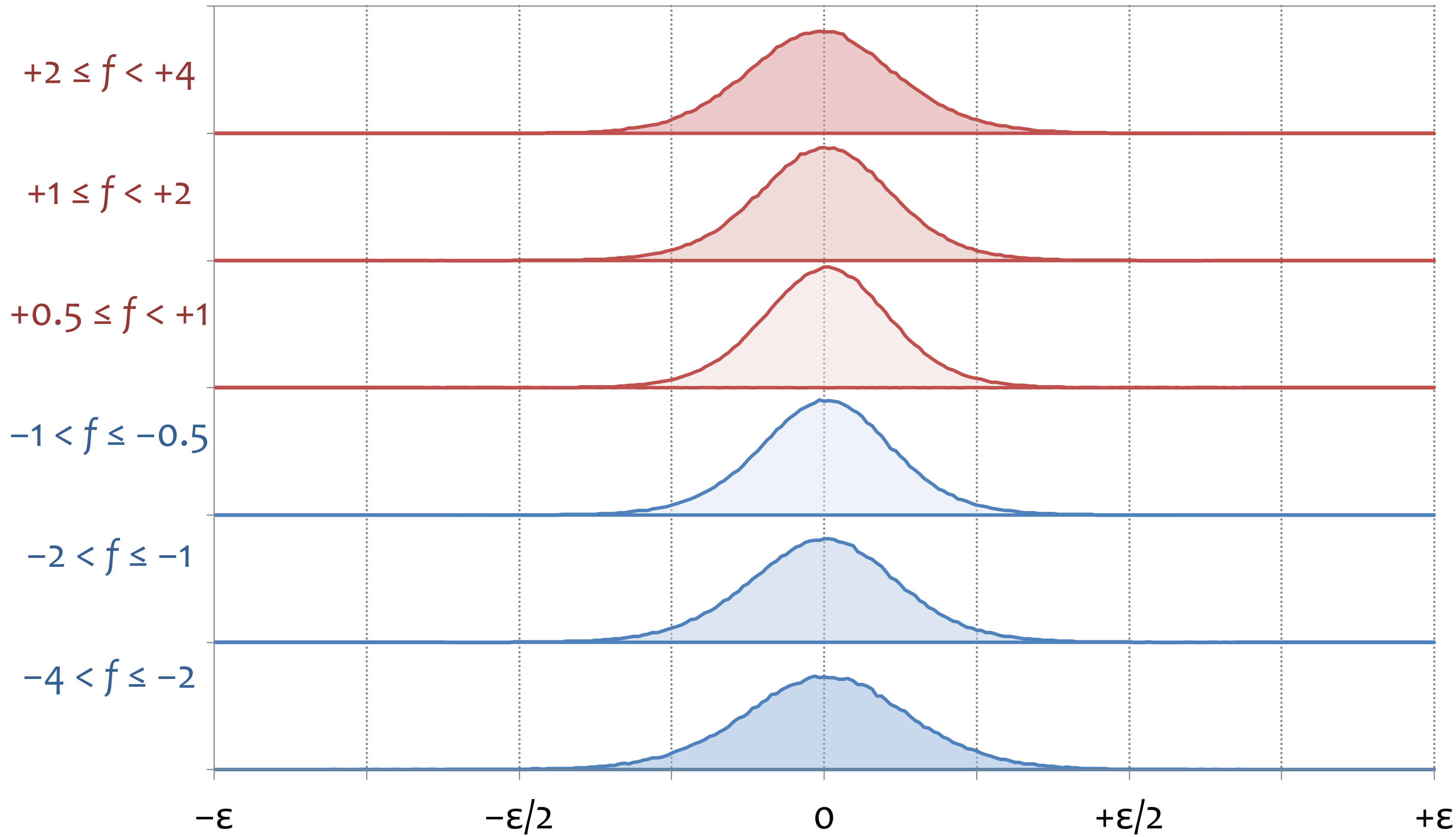


ZFP limits data loss via a small write-back cache

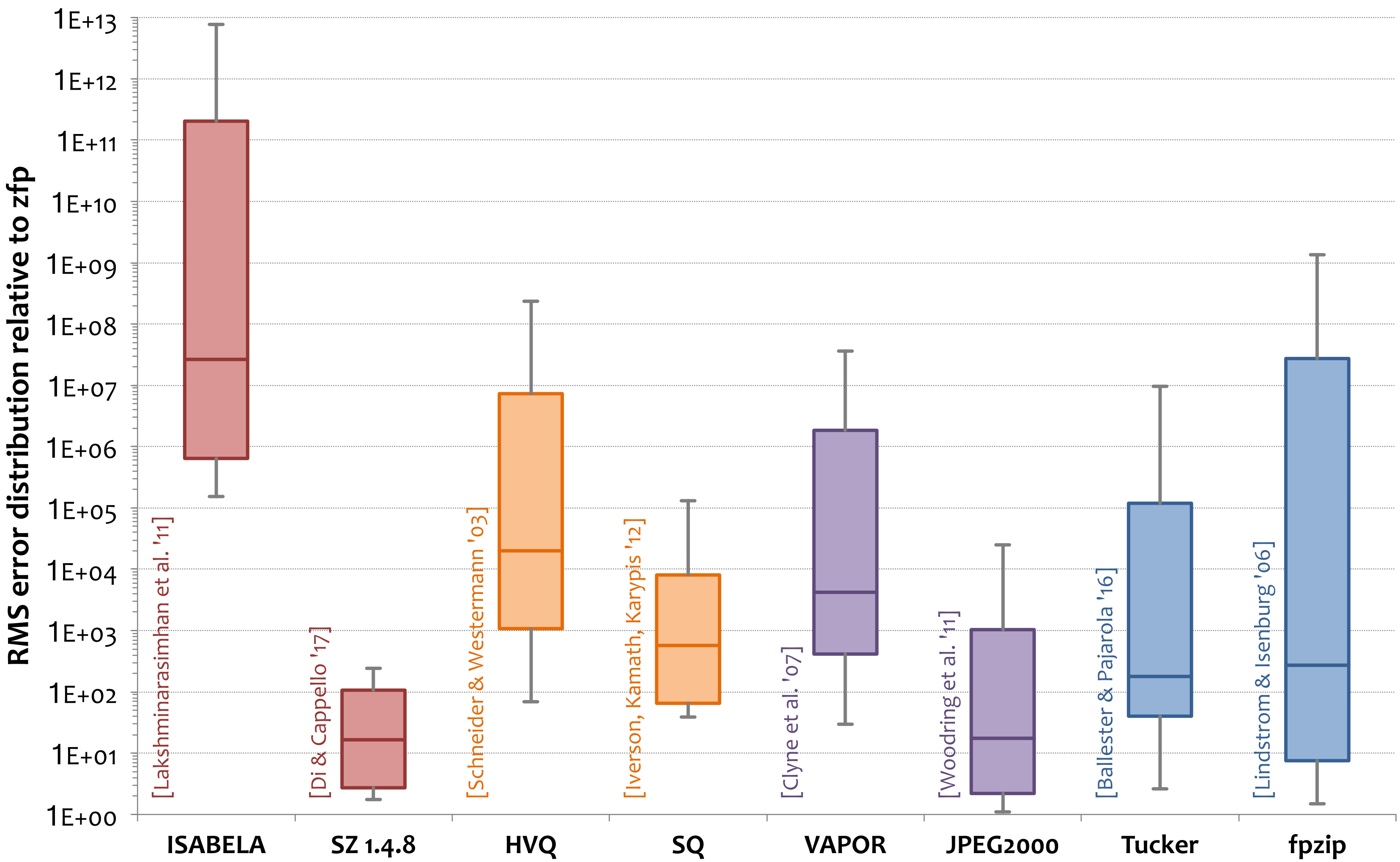


Compress only "dirty" blocks when evicted from the cache

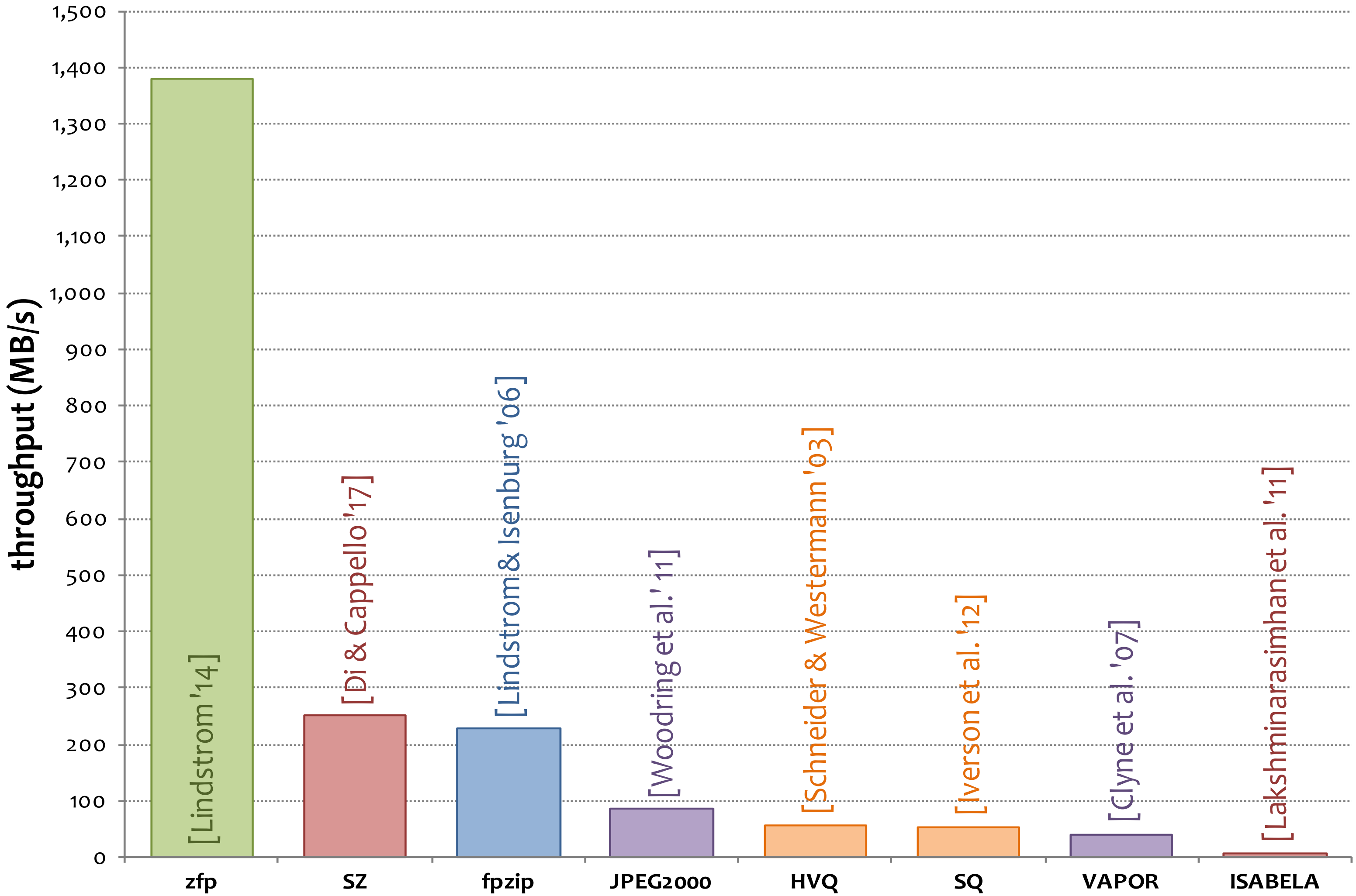
ZFP error distribution is normal due to linear transform of iid. errors (central limit theorem)



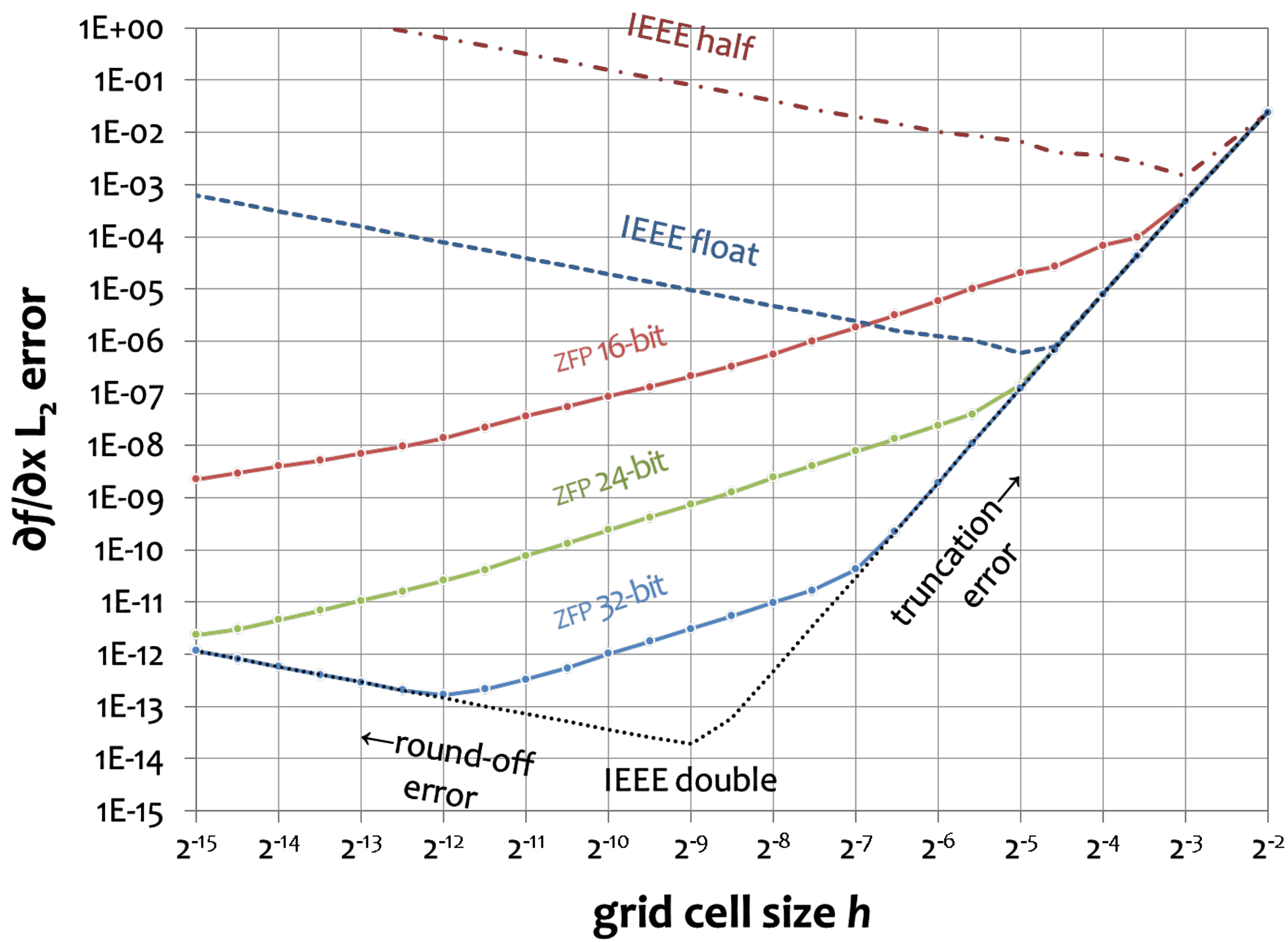
ZFP consistently achieves high quality across a diverse collection of scientific data sets



ZFP is among the fastest compressors available



ZFP *improves* accuracy in finite difference computations using less precision than IEEE



Shock wave propagation using 16-bit ZFP arrays agrees well with double-precision solution