cuSZ(+): Optimizing Error-Bounded Lossy Compression for Scientific Data on Modern GPUs

Jiannan Tian, Dingwen Tao
Washington State University
Pullman, WA, United States
{jiannan.tian,dingwen.tao}@wsu.edu

Sheng Di, Franck Cappello
Argonne National Laboratory
Lemont, IL, United States
sdi1@anl.gov,cappello@mcs.anl.gov

Abstract

Error-bounded lossy compression is a critical technique for significantly reducing scientific data volumes. With ever-emerging heterogeneous high-performance computing (HPC) architecture, GPU-accelerated error-bounded compressors (such as cuSZ and cuZFP) have been developed. However, they suffer from either low performance or low reduction rates. To this end, we propose cuSZ(+) to target both high reduction rates and throughputs. Furthermore, we identify that data smoothness is a vital factor for high compression throughputs. Our key contributions are fourfold: (1) We propose an efficient compression workflow to adaptively perform run-length encoding with or without variable-length encoding. (2) We derive Lorenzo reconstruction in decompression as multidimensional partial-sum computation and propose its well-formed GPU implementation. (3) We optimize essential kernels and scale to the state-of-the-art A100 GPU. (4) We evaluate cuSZ(+) using real-world HPC datasets on V100 and A100. Experiments show cuSZ(+) improves the compression throughputs and ratios by up to 18.4× and 5.3×, respectively, over cuSZ.

1 Introduction

Large-scale scientific applications and advanced instruments produce vast volumes of data for post hoc analysis, pressuring and quickly saturating parallel file system that is orders of magnitude lower in I/O bandwidth [1, 2]. For instance, a trillion-particle HACC[1] simulation produces petabytes of data with hundreds of snapshots. Error-bounded lossy compressors thrive to address two scientific concerns: low reduction rate seen in lossless compression [5, 6] and the uncertainty of information loss exhibited in traditional lossy compressors (e.g., JPEG [7] and JPEG2000 [8]). The state-of-the-art error-bounded lossy compressors can get multi-hundred-fold reduction rates [1, 9, 10, 11] and strictly control the data distortion regarding the user-set error bound.

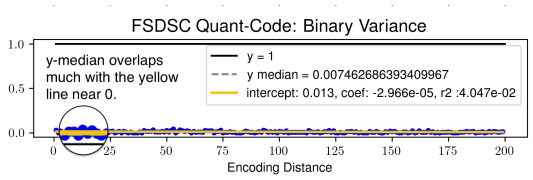
Scientific lossy compressors qualify by addressing three primary concerns: high fidelity, high reduction rate, and high throughput. With the first two satisfied, most existing error-bounded lossy compressors (e.g., SZ [9, 10], FPZIP [12], ZFP [11]) target CPU architecture and hence lack high-throughput processing, far from the multi-hundred-GBps goal for scientific projects[13]. The currently GPU-based error-controlled lossy compressors (e.g., cuSZ [14], and cuZFP [15]) emphasize on processing capability, leaving the reduction rate far from optimal. For instance, cuZFP's reduction rate is limited to 16×, and cuSZ's to 32×. We hereby present an efficient compression framework cuSZ(+) on top of cuSZ [14], with key contributions summarized: (1) we design an adaptive workflow featuring run-length encoding (RLE) for a higher reduction rate; (2) we attribute first-order Lorenzo reconstruction to N-D partial-sum and have well-formed implementation. (3) we optimize essential kernels and scale to the leading NVIDIA A100 GPU. (4) we evaluate on real-world HPC datasets from [16] on V100 and A100.

[1]Hardware/Hybrid Accelerated Cosmology Code [3, 4].

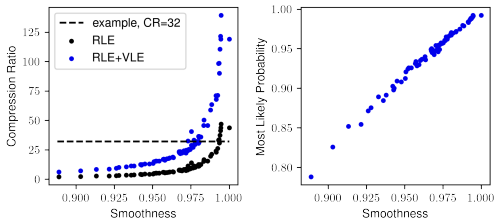
2 Design

The design is in two parts, (1) the pattern-finding method based on data smoothness and (2) the N-D partial-sum-based Lorenzo reconstruction.

2.1 Smoothness and Reduction Rate (RR)



(a) Smoothness against encoding distance (CESM FSDSC at 1e-2). The yellow line denotes linear regression of variances at distances.



(b) Smoothness-Probability of the top-1 symbol relationship.

Figure 1: Relationships that help determine when to use RLE.

Lossless RLE [17] detects consecutive same-value elements and the number they continue to form value-count tuples. Whether or not continuing becomes a binary pattern with regular memory access, beneficial for throughput on GPU. The overhead from storing count raises our caution of proper use: the data should be *smooth* enough. Inspired by sampling-based *variogram* method [18], we adapt its *madogram* variant of absolute difference against distance with a randomized start, $2\gamma(\mathbf{s}_1, \mathbf{s}_2) = E[|Z(\mathbf{s}_1) - Z(\mathbf{s}_2)|]$, where $Z(\mathbf{s})$ is a spatial random field. We further adjust the absolute difference to *binary* variance, defined as $= 0$, when $v_{\text{this}} = v_{\text{next}}$, otherwise 1. Its expected value for each distance is interpreted as *roughness*, with its dual (1-roughness) being *smoothness*. Figure 1 shows the binary variance is a stable indicator of RLE-smoothness, becoming a constant regardless of the distance. The probability of the top-1 symbol (p_1) from the histogram is computationally cheap. Thus, by further bridging smoothness, RR, and p_1 (Figure1b), we

can set the desired threshold (e.g., 32× in RR) for the corresponding p_1. An additional can provide a stable 3× RR. For example, FSDSC has an RLE-RR above 25 while cuSZ-VLE-RR is 26× to 29× (estimated) and the additional VLE makes the total RR above 70×.

2.2 Well-Formed Lorenzo Reconstruction

We use 2D form of Lorenzo predictor for quick grasp. The 2D prediction is given by $p_{[y,x]} = -d_{[y-1,x-1]} + d_{[y-1,x]} + d_{[y,x-1]}$. The reversal, Lorenzo reconstruction, can be $d_{[y,x]} = p_{[y,x]} + q_{[y,x]}$ with $\sum_{j=0}^{y}\sum_{i=0}^{x} q_{[j,i]}$, where q is the error-control code, quantcode. We give a proof by induction on $[y, x]$, as $d_{[y+1,x+1]}$ equals to

$$-\sum_{j=0}^{y}\sum_{i=0}^{x} q_{[j,i]} + \sum_{j=0}^{y}\sum_{i=0}^{x+1} q_{[j,i]} + \sum_{j=0}^{y+1}\sum_{i=0}^{x} q_{[j,i]} + q_{[y+1,x+1]}$$
$$= \sum_{i=0}^{y} q_{[j,x+1]} + q_{[y+1,x+1]} + \sum_{j=0}^{y+1}\sum_{i=0}^{x} q_{[j,i]} = \sum_{j=0}^{y+1}\sum_{i=0}^{x+1} q_{[j,i]}.$$

Computation wise, we define N-D partial-sum of x till index $[k_N, \ldots, k_2, k_1] \in \mathbb{N}^N$ as

$$p\Sigma(x; k_N, \ldots, k_2, k_1) = \sum_{i_N=0}^{k_N} \cdots \sum_{i_2=0}^{k_2} \sum_{i_1=0}^{k_1} x_{[i_N,\ldots,i_2,i_1]},$$

where $p\Sigma$ is a variadic operator for any N. We can decompose it to N-pass 1-D partial-sums, as

$$p\Sigma(x; k_N, \ldots, k_2, k_1) = p\Sigma(p\Sigma(x; k_{N-1}, \ldots, k_2, k_1); k_N)$$
$$= p\Sigma(p\Sigma(\cdots p\Sigma\Big(p\Sigma\Big(x; k_1\Big); k_2\Big)\cdots; k_{N-1}); k_N).$$

That is, the output of a partial-sum on x_m-direction is the input of that on $x_{(m+1)}$-direction. Given the problem size $(X_N, \ldots, X_2, X_1)$, where $k_{(\cdot)} \leq X_{(\cdot)}$, a pass along $x_{(\cdot)}$ features the degree of independence (hence the maximum possible parallelism) equal to $\prod_{i\neq(\cdot)} X_i$.

The implementation features careful data-access arrangement, including (1) coalescing load, global- to shared-memory, (2) coarsening by assigning multiple items to a thread, (3) in-warp shuffle for partial-sum[2], and (4) coalescing access to shared memory for out-of-warp data exchange, (5) coalescing store to global memory.

3 Evaluation

Our evaluation setup is listed below,

platform A100 of ALCF-ThetaGPU, V100 of TACC-Longhorn
dataset 1D HACC; 2D CESM; 3D Hurricane, Nyx, QMC.

3.1 Evaluation of Reduction Rate

	cuSZ	cuSZ(+)		cuSZ(+)	
	VLE	RLE	gain	RLE+VLE	gain
FSDSC	23.88	26.10	1.09×	71.35	2.99×
FSDTOA	26.10	43.65	1.67×	119.17	4.57×
ODV_bcar1	25.83	37.28	1.44×	110.51	4.28×
ODV_bcar2	25.83	30.71	1.19×	89.98	3.48×
ODV_dust1	26.10	22.91	-	67.72	2.59×
ODV_dust2	26.37	24.02	-	70.98	2.69×
ODV_dust3	26.10	33.29	1.28×	98.22	3.76×
ODV_dust4	26.10	46.81	1.79×	139.27	5.34×
ODV_ocar1	24.11	41.17	1.71×	121.59	5.04×
ODV_ocar2	24.11	33.79	1.40×	98.63	4.09×
PRECSC	25.83	19.50	-	58.92	2.28×
SNOWHLND	25.57	21.18	-	63.33	2.48×
SOLIN	26.10	43.65	1.67×	119.17	4.57×

Table 1: cuSZ(+) gain in reduction rate over cuSZ-VLE under 1e-2 error bound. The selection features either the gain from RLE is greater than 1.0× or that from RLE+VLE is greater than 2.0×.

[2] NVIDIA::cub is used in 1D case, but there is no 2D or 3D partial-sum from that.

Table 1 shows several cases that RLE outperforms in reduction rate than cuSZ-VLE. An additional VLE can get up to 5.3× reduction rate improvements over cuSZ.

3.2 Evaluation of Performance

V100		HACC	CESM	Hurr	Nyx	QMC
Lorenzo	cuSZ	207.7	252.1	175.8	200.2	189.6
construct	cuSZ(+)	307.4	273.9	229.9	296	298.6
		1.48×	1.09×	1.31×	1.48×	1.57×
Huffman	cuSZ	54.1	57.2	55.2	58.8	61
encode	cuSZ(+)	58.3	107.7	111.2	120.5	110.8
		1.08×	1.88×	2.01×	2.05×	1.82×
Lorenzo	cuSZ	16.8	58.5	43.9	29.7	22.4
reconstruct	cuSZ(+)	313.1	254.2	218.4	238.1	255.5
		18.64×	4.35×	4.97×	8.02×	11.41×

Table 2: Performance comparison of cuSZ(+) and Huffman encoding kernels in cuSZ(+) and cuSZ on V100. The unit is in GB/s.

size in MB		1071.8	24.7	95.4	512.0	601.5
		HACC	CESM	Hurr	Nyx	QMC
Lorenzo	V100	328.3	273.9	199.0	296.0	298.6
construct	A100	501.1	466.8	429.0	481.3	492.9
		1.53×	1.70×	2.16×	1.63×	1.65×
Huffman	V100	58.3	107.7	111.2	120.5	110.8
encode	A100	174.6	121.6	206.0	217.2	198.4
		2.99×	1.13×	1.85×	1.80×	1.79×
Lorenzo	V100	308.7	267.0	200.1	251.7	255.5
reconstruct	A100	504.4	495.3	345.5	398.6	384.0
		1.63×	1.86×	1.73×	1.58×	1.50×

Table 3: Evaluation of cuSZ(+) using default compression workflow (Lorenzo and VLE) with relative error bound of 10^{-4} on V100 and A100: breakdown throughput of compression subprocedures.

With the baseline from [14], Table 2 illustrates that the performance improvements of cuSZ(+)'s Lorenzo construction kernels are 1.48× for 1D data, 1.09× for 2D data, and 1.45× for 3D data on average over cuSZ. Moreover, we increase the lowest throughput from 175.8 GB/s to 229.9 GB/s (+30.7%) on the tested datasets. Table 3 shows that the optimized kernels scale well from V100 to A100. The scaling is by 1.53× to 2.16× for Lorenzo construction, 1.13× to 2.99× for Huffman encoding, and 1.50× to 1.86× for Lorenzo reconstruction.

Conclusion

In this work, we propose cuSZ(+), a compressibility-aware GPU-based lossy compressor for NVIDIA GPU architectures, which can (1) make use of data smoothness to boost reduction rate and (2) improve the compression throughput over cuSZ. The optimization also shows good scalability of kernels from V100 to A100.

Acknowledgments

This research was supported by the Exascale Computing Project (ECP), Project Number: 17-SC-20-SC, a collaborative effort of two DOE organizations—the Office of Science and the National Nuclear Security Administration, responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, to support the nation's exascale computing imperative. The material was supported by the U.S. Department of Energy, Office of Science, under contract DE-AC02-06CH11357. This work was also supported by the National Science Foundation under Grants OAC-2042084, OAC-2034169, OAC-2003709, and CCF-1619253.

References

[1] X. Liang *et al.*, "Error-controlled lossy compression optimized for high compression ratios of scientific datasets," in *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA: IEEE, 2018, pp. 438–447.

[2] X. Liang, S. Di, D. Tao, Z. Chen, and F. Cappello, "An efficient transformation scheme for lossy data compression with point-wise relative error bound," in *IEEE International Conference on Cluster Computing (CLUSTER)*, Belfast, UK: IEEE, 2018, pp. 179–189.

[3] S. Habib *et al.*, "HACC: Extreme scaling and performance across diverse architectures," *Communications of the ACM*, vol. 60, no. 1, pp. 97–104, 2016.

[4] S. C. V. Vishwanath and K. Harms, *Parallel i/o on mira*, https://www.alcf.anl.gov/files/Parallel_IO_on_Mira_0.pdf, Online, 2019.

[5] D. Meister, J. Kaiser, A. Brinkmann, T. Cortes, M. Kuhn, and J. Kunkel, "A study on data deduplication in HPC storage systems," in *SC '12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, Salt Lake City, UT, USA: IEEE, 2012, p. 7.

[6] S. W. Son, Z. Chen, W. Hendrix, A. Agrawal, W.-k. Liao, and A. Choudhary, "Data compression for the exascale computing era-survey," *Supercomputing Frontiers and Innovations*, vol. 1, no. 2, pp. 76–88, 2014.

[7] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[8] D. Taubman and M. Marcellin, *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*. Boston, MA, USA: Springer Science & Business Media, 2012, vol. 642.

[9] S. Di and F. Cappello, "Fast error-bounded lossy HPC data compression with SZ," in *2016 IEEE International Parallel and Distributed Processing Symposium*, Chicago, IL, USA: IEEE, 2016, pp. 730–739.

[10] 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," in *2017 IEEE International Parallel and Distributed Processing Symposium*, Orlando, FL, USA: IEEE, 2017, pp. 1129–1139.

[11] P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2674–2683, 2014.

[12] P. Lindstrom and M. Isenburg, "Fast and efficient compression of floating-point data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1245–1250, 2006.

[13] F. Cappello *et al.*, "Use cases of lossy compression for floating-point data in scientific data sets," *The International Journal of High Performance Computing Applications*, vol. 33, no. 6, pp. 1201–1220, 2019.

[14] J. Tian *et al.*, "Cusz: An efficient gpu-based error-bounded lossy compression framework for scientific data," in *Proceedings of the ACM International Conference on Parallel Architectures and Compilation Techniques*, 2020, pp. 3–15.

[15] cuZFP, https://github.com/LLNL/zfp/tree/develop/src/cuda_zfp, Online, 2019.

[16] Scientific Data Reduction Benchmarks, https://sdrbench.github.io/, Online, 2019.

[17] A. H. Robinson and C. Cherry, "Results of a prototype television bandwidth compression scheme," *Proceedings of the IEEE*, vol. 55, no. 3, pp. 356–364, 1967. DOI: 10.1109/PROC.1967.5493.

[18] N. Cressie and D. M. Hawkins, "Robust estimation of the variogram: I," *Journal of the International Association for Mathematical Geology*, vol. 12, no. 2, pp. 115–125, 1980.