SZ

SZ: Fast Error-Bounded Scientific Data Compressor

Keywords: floating point data compressor, lossy compressor, error bounded compression

Key developers: Sheng Di, Dingwen Tao, Xin Liang ; Other contributors: Ali M. Gok (Pastri version), Sihuan Li (Time-based compression for HACC simulation) ; Supervisor: Franck Cappello

Today’s HPC applications are producing extremely large amounts of data, thus it is necessary to use an efficient compression before storing them to parallel file systems.

We developed the error-bounded HPC data compressor, by proposing a novel HPC data compression method that works very effectively on compressing large-scale HPC data sets.

The key features of SZ are listed below.

1. **Usage:**

Compression: Input: a data set (or a floating-point array with any dimensions); Output: the compressed byte stream

Decompression: input: the compressed byte stream; Output: the original data set with the compression error of each data point being within a pre-specified error bound .

2. **Environment**: SZ supports C, Fortran, and Java. It has been tested on Linux and Mac, with different architectures (x86, x64, ppc, etc.).

3. **Error control**: SZ supports many types of error bounds. The users can set *absolute error bound, value-range based relative error bound,* or *a combination of the two boundswith* operator *AND* or *OR). The users can also error bound mode to be PNSR-fixed, the point-wise relative error bound, etc. More details can be found in the configuration file (sz.config).*

- The absolute error bound (denoted) is a constant, such as 1E-6. That is, the decompressed data Di must be in the range [Di ,Di +], where Di is referred as the decompressed value and Di is the original data value.
- As for the relative error bound, it is a linear function of the global data value range size, i.e., =r, where ((0,1)) and r refer to *error bound ratio* and range size respectively. For example, given a set of data, the range size r is equal to max (Di) min (Di), and the error bound can be written as (max (Di) min (Di)). The relative error bound allows making sure that the compression error for any data point must be no greater than ×100 percentage of the global data value range size.
- PSNR-fixed compression allows users to set a PSNR value, based on which the compressor will compress the data.

4. SZ supports two compression modes (similar to Gzip): SZ_BEST_SPEED and SZ_BEST_COMPRESSION. SZ_BEST_SPEED results in the fastest compression. The best compression factor will be reached when using SZ_BEST_COMPRESSION and ZSTD_FAST_SPEED meanwhile. The default setting is SZ_BEST_COMPRESSION + Zstd.

5. **User guide:** More detailed usage and examples can be found under the directories doc/user-guide.pdf and example/ respectively, in the package.

6. **Citations**: If you mention SZ in your paper, please cite the following references.

- **SZ 0.1-0.15**: Sheng Di, Franck Cappello, "Fast Error-bounded Lossy HPC Data Compression with SZ," in International Parallel and Distributed Processing Symposium (IEEE/ACM IPDPS 2016), 2016.
- **SZ 1.0-1.4.13**: Dingwen Tao, Sheng Di, Franck Cappello, "A Novel Algorithm for Significantly Improving Lossy Compression of Scientific Data Sets, " in International Parallel and Distributed Processing Symposium (IEEE/ACM IPDPS 2017), Orlando, Florida, 2017.
- **SZ 2.0+**: Xin Liang, Sheng Di, Dingwen Tao, Zizhong Chen, Franck Cappello, "Error-Controlled Lossy Compression Optimized for High Compression Ratios of Scientific Datasets", in IEEE Bigdata2018, 2018.
- As for the point-wise relative error bound mode (i.e., PW_REL), our CLUSTER18 paper describes the key design: Xin Liang, Sheng Di, Dingwen Tao, Zizhong Chen, Franck Cappello, "Efficient Transformation Scheme for Lossy Data Compression with Point-wise Relative Error Bound", in IEEE CLUSTER 2018. (best paper)

7. **Download**

Version SZ 2.0.2.0

-->>> Package Download (including everything) <<<--

-->>> **Github of SZ** <<<--

-->>> **User Guide , hands-on-document** <<<---

(Contact: disheng222@gmail.com or sdi1@anl.gov)

If you download the code, please let us know who you are. We are very keen of helping you using the SZ library.

8. **Publications:**

- Sheng Di, Franck Cappello, "Fast Error-bounded Lossy HPC Data Compression with SZ," to appear in International Parallel and Distributed Processing Symposium (IEEE/ACM IPDPS 2016), 2016. [download]
- Dingwen Tao, Sheng Di, Franck Cappello, "A Novel Algorithm for Significantly Improving Lossy Compression of Scientific Data Sets, " to appear in International Parallel and Distributed Processing Symposium (IEEE/ACM IPDPS 2017), Orlando, Florida, 2017. [download]

- Dingwen Tao, Sheng Di, Zizhong Chen, and Franck Capello, "Exploration of Pattern-Matching Techniques for Lossy Compression on Cosmology Simulation Data Sets ", Proceedings of the 1st International Workshop on Data Reduction for Big Scientific Data (DRBSD1) in Conjunction with ISC'17, Frankfurt, Germany, June 22, 2017.
- Ian T. Foster, Mark Ainsworth, Bryce Allen, Julie Bessac, Franck Cappello, Jong Youl Choi, Emil M. Constantinescu, Philip E. Davis, Sheng Di, et al., "Computing Just What You Need: Online Data Analysis and Reduction at Extreme Scales", in 23rd International European Conference on Parallel and Distributed Computing (Euro-Par 2017), 2017. pp. 3-19.
- Sheng Di, Franck Cappello, "Optimization of Error-Bounded Lossy Compression for Hard-to-Compress HPC Data," in IEEE Transactions on Parallel and Distributed Systems (IEEE TPDS), 2017.
- Ali Murat Gok, Dingwen Tao, Sheng Di, Vladimir Mironov, Yuri Alexeev, Franck Cappello, "PaSTRI: A Novel Data Compression Algorithm for Two-Electron Integrals in Quantum Chemistry", in IEEE/ACM 29th The International Conference for High Performance computing, Networking, Storage and Analysis (SC2017). [poster]
- Dingwen Tao, Sheng Di, Zizhong Chen, and Franck Cappello, "In-Depth Exploration of Single-Snapshot Lossy Compression Techniques for N-Body Simulations", Proceedings of the 2017 IEEE International Conference on Big Data (BigData2017), Boston, MA, USA, December 11 - 14, 2017. [short paper]
- Dingwen Tao, Sheng Di, Hanqi Guo, Zizhong Chen, and Franck Cappello, "Z-checker: A Framework for Assessing Lossy Compression of Scientific Data", in The International Journal of High Performance Computing Applications (IJHPCA), 2017. [download]
- Sheng Di, Dingwen Tao, Xin Liang, and Franck Cappello, "Efficient Lossy Compression for Scientific Data based on Pointwise Relative Error Bound", in IEEE Transactions on Parallel and Distributed Systems (IEEE TPDS), 2018.
- Dingwen Tao, Sheng Di, Xin Liang, Zizhong Chen and Franck Cappello, "Optimization of Fault Tolerance for Iterative Methods with Lossy Checkpointing", in 27th ACM Symposium on High-Performance Parallel and Distributed Computing (ACM HPDC2018), 2018.
- Ali Murat Gok, Sheng Di, Yuri Alexeev, Dingwen Tao, V. Mironov, Xin Liang, Franck Cappello, "PaSTRI: Error-bounded Lossy Compression for Two-Electron Integrals in Quantum Chemistry", in IEEE CLUSTER 2018, 2018. [best paper award (in the application, algorithms and libraries track]
- Xin Liang, Sheng Di, Dingwen Tao, Zizhong Chen, and Franck Cappello, "Efficient Transformation Scheme for Lossy Data Compression with Point-wise Relative Error Bound", in IEEE CLUSTER 2018. [best paper award (in the Data, Storage, and Visualization track)]
- Dingwen Tao, Sheng Di, Xin Liang, Zizhong Chen, and F. Cappello, "Fixed-PSNR Lossy Compression for Scientific Data", in IEEE CLUSTER 2018. (short paper)
- Xin Liang, Sheng Di, Dingwen Tao, Zizhong Chen, Franck Cappello, "Error-Controlled Lossy Compression Optimized for High Compression Ratios of Scientific Datasets", in IEEE Bigdata2018, 2018.
- Sihuan Li, Sheng Di, Xin Liang, Zizhong Chen, Franck Cappello, "Optimizing Lossy Compression with Adjacent Snapshots for N-body Simulation", in IEEE Bigdata2018, 2018.
- Xin Liang, Sheng Di, Dingwen Tao, Sihuan Li, Zizhong Chen, Franck Cappello, "Improving In-situ Lossy Compression with Spatio-Temporal Decimation based on SZ Model", in Proceedings of the 4th International Workshop on Data Reduction for Big Scientific Data (DRBSD-4), in conjunction with IEEE/ACM 29th The International Conference for High Performance computing, Networking, Storage and Analysis (SC2018).
- Xin-Chuan Wu, Sheng Di, Franck Cappello, Hal Finkel, Yuri Alexeev, Frederic T. Chong, "Amplitude-Aware Lossy Compression for Quantum Circuit Simulation", in Proceedings of the 4th International Workshop on Data Reduction for Big Scientific Data (DRBSD-4), in conjunction with IEEE/ACM 29th The International Conference for High Performance computing, Networking, Storage and Analysis (SC2018).
- Xin-Chuan Wu, Sheng Di, Franck Cappello, Hal Finkel, Yuri Alexeev , Frederic T. Chong, "Memory-Efficient Quantum Circuit Simulation by Using Lossy Data Compression", The 3rd International Workshop on Post-Moore Era Supercomputing (PME) in conjunction with IEEE/ACM 29th The International Conference for High Performance computing, Networking, Storage and Analysis (SC2018).
- Dingwen Tao, Sheng Di, Xin Liang, Zizhong Chen, Franck Cappello, "Optimizing Lossy Compression Rate-Distortion from Automatic Online Selection between SZ and ZFP", in IEEE Transactions on Parallel and Distributed Systems (IEEE TPDS), 2019.
- XiangYu Zou, Tao Lu, Wen Xia, Xuan Wang, Weizhe Zhang, Sheng Di, Dingwen Tao, Franck Cappello, "Accelerating Relative-error Bounded Lossy Compression for HPC datasets with Precomputation-Based Mechanisms", in Proceedings of the 35th International Conference on Massive Storage Systems and Technology (MSST19), 2019.

9. **Version history**: We recommend the latest version.

- SZ 2.1: Significantly improve the compression rate and decompression rate for point-wise relative error bounded compression. See our paper published in MSST19 for details.
- SZ 2.0.1.0: Significantly improve rate-distortion (compression ratio vs. PSNR) for many datasets in high-compression cases.
- SZ 1.4.13: (1) support openMP version for both single-precision and double-precision floating-point data compression. (2) support Pastri algorithm customized for GAMESS data (two-electron integral data)
- SZ 1.4.12.1: (1) Fix the bug the segmentation fault may happen when the error bound is greater than the value range size.
- SZ 1.4.12: (1) Support thresholding-based strategy for 1D data compression based on point-wise relative error bound. (In order to test it, please select errBoundMode = PW_REL, and set the point-wise relative error bound using the parameter pw_relBoundRatio in the sz.config.) For other dimensions of data, point-wise relative error based compression is using block-based strategy (see our DRBSD-2 paper for details) (2) fix the bug in the callZlib.c (previously, segmentation fault might happen when using best_compression mode). (2) Fix a small bug that happened when the data size is extremely huge (nbEle>4G) and the compression mode is SZ_BEST_COMPRSSION. Specifically, the previous call to zlib functions has one potential bug that may lead to segmentation fault, which has been fixed.
- SZ 1.4.11-beta: (1) Support HDF5 (using HDF5 filter : id (32017)); (2) Support integer data compression (see testint_compress.c in example/ for details).
- SZ 1.4.10-beta: (1) Support direct sub-block data compression; (2) Support compression of large data file directly (i.e., the number of data points could be up to as large as LONG size, unlike the previous version that can only compress 2^{32} data points each time); (3) separate the internel functions from the sz.h;
- SZ 1.4.9-beta: allows users to switch off/on the Fortran compilation on demand (Fortran compile is off by default). Support lossy compression with 'point-wise relative error bound ratio'. For example, given a relative error bound ratio (such as 0.001), SZ can make sure the compression error for each data point be limited within {the relative error ratio}*{the data point's value} (e.g., err_bound=0.001*{data_value}). For details, please set the errBoundMode to PW_REL in the configuration file.
- SZ 1.4.8-beta Increase the max number of quantization intervals (from 65536 to 2^30), which will lead to better compression ratio on high-precision data compression. This version also allows users to specify the maximum number of quantization intervals in the configuration file. Fix the issue of possible non-identical compression output with multiple runs.
- SZ 1.4.7-beta Fix some memory leakage bugs (related to Huffman encoding). Fix the bugs about memory crash or segmentation faults when the number of data points is pretty large. Fix the sementation fault bug happening when the data size is super small. Fix the issue that decompressed data may be largely skewed from the original data in some special cases (especially when the data are not smooth at all). - Dec. 17th, 2016.
- SZ 1.4.6-beta The compression ratio and speed are further improved than SZ 1.3 in most cases. We also provide three compression modes: SZ_BEST_SPEED, SZ_DEFAULT_COMPRESSION, and SZ_BEST_COMPRESSION. Please read the user guide for details.
- SZ 1.3 The compression ratio and speed are further improved than SZ 1.2.
- SZ 1.2 The compression ratio is improved significantly compared with SZ 1.1.

- SZ 1.1 This version improved the compression performance by 50% compared to SZ1.0. A few bugs that may make the compression disrespect the error-bound are also fixed.
- SZ 1.0 This version is coded in C programming language, unlike the previous version coded in Java. It also allows setting the endianType for the data to compress.
- SZ 0.5.14 fixed a design bug, which improves the compression ratio further.
- SZ 0.5.13 improves compression performance, by replacing the implementation with classes by that of primitive data types.
- SZ 0.5.12 allows users to set "offset" parameter in the configuration file sz.config. The value of the offset is an integer in [1,7]. Generally, we recommend offset=2 or 3, while we also find that some other settings (such as offset=7) may lead to better compression ratios in some cases. How to automize/optimize the selection of offset value would be the future work. In addition, the compression speed is improved, by replacing java List by array implementation in the code.
- SZ 0.5.11 improved SZ 0.5.10 on the level of guaranteeing user-specified error bounds. In very few cases, SZ 0.5.10 cannot guarantee the error-bounds to a certain user-specified level. For example, when absolute error bound = 1E-6, the maximum decompression error may be 0.01(>>1E-6) because of the huge value range even in the optimized segments such that the normalized data cannot reach the required precision even storing all of the 64 or 32 mantissa bits. SZ 0.5.11 fixed the problem well, with compression ratio degraded by less than 1% in that case.
- SZ 0.5.10 optimizes the offset by using the optimized formula of computing the median_value based on optimized right-shifting method. Anyway, this version improves compression ratio a lot for hard-to-compress datasets. (Hard-to-compress datasets refer to the cases whose compression ratios are usually very limited)
- SZ 0.5.9 optimize the offset by using the simple right-shifting method. Experiments show that this cannot improve compression ratio actually because simple right-shifting actually make each data be multiplied by 2^{-k}, where k is # right-shifting bits. The pros is to save bits because of more leading-zero bytes, but the cons are much more required bits to save. See SZ 0.5.10 for the better solution on this issue!
- SZ 0.5.8 Refine the leading-zero granularity (change it from byte to bits based on the distribution). For example, in SZ0.5.7, the leading-zero is always in bytes, 0, 1, 2, or 3. In SZ0.5.8, the leading-zero part could be xxxx xxxx xx xx xx xx xxxx xxxx (where each x means a bit in the leading-zero part)
- SZ 0.5.7 improve the decompression speed for some cases
- SZ 0.5.6 improve compression ratio for some cases (when the values in some segmentation are always the same, this segment will be merged forward)
- SZ 0.5.5 runtime memory is shrunk (by changing int xxx to byte xxx in the codes. The bug that writing decompressed data may encounter exceptions is fixed. Memory leaking bug for ppc architecture is fixed.
- SZ 0.5.4 Gzip_mode: default --> fast_mode ; Support reserved value
- SZ 0.5.3 Integrate with the dynamic segmentation support
- SZ 0.5.2 finer compression granularity for unpredictable data, and also remove redundant Java storage bytes.
- SZ 0.5.1 Support version checking
- SZ 0.2-0.4 Compression ratio is the same as SZ 0.5. The key difference is different implementation ways, such that SZ 0.5 is much faster than SZ 0.2-0.4.

10. Other versions are available upon request