Toward Decoupling the Selection of Compression Algorithms from Quality Constraints

Julian Kunkel[1] <kunkel@dkrz.de>, Anastasiia Novikova[2], Eugen Betke[1], Armin Schaare[2]

[1] *Deutsches Klimarechenzentrum (DKRZ)* [2] *Universität Hamburg*

Please see our **standardization effort**: https://www.vi4io.org/std/compression

Abstract

Data intense scientific domains use data compression to reduce the storage space needed. Lossless data compression preserves the original information accurately but usually yields a compression ratio of 2:1. Lossy data compression can achieve much higher compression rates depending on the tolerable error/precision needed. Therefore, the field of lossy compression is still subject to active research. From the perspective of a scientist, the actual algorithm does not matter but the qualitative information about the implied loss of precision of data is the main concern.

With the Scientific Compression Library (SCIL), we are developing a meta-compressor that allows users to set various quantities that define the acceptable error and the expected performance behavior. The library then chooses the appropriate chain of algorithms to yield the users requirements. This approach is a crucial step towards a scientifically safe use of much-needed lossy data compression, because it disentangles the tasks of determining scientific ground characteristics of tolerable noise, from the task of determining an optimal compression strategy given target noise levels and constraints. Without changing applications, it allows these codes to utilize future algorithms once they are integrated into the library.

Contributions of this poster are:

1. Introduction of quantities for acceptable tolerance and performance
2. The analysis of (lossless) compression for climate data (and two new algorithms)

The current version of the library is publicly available under LGPL license:
https://github.com/JulianKunkel/scil

Supported Quantities

SCIL supports to define the tolerable error on data and the expected performance behavior.

Quantities defining the residual (error):

- **absolute tolerance**: compressed can become true value ± absolute tolerance
- **relative tolerance**: percentage the compressed value can deviate from true value
- **relative error finest tolerance**: value defining the absolute tolerable error for relative compression for values around 0
- **significant digits**: number of significant decimal digits
- **significant bits**: number of significant decimals in bits

Defining the performance behavior can be defined for compression and decompression. Each value can be defined according to:

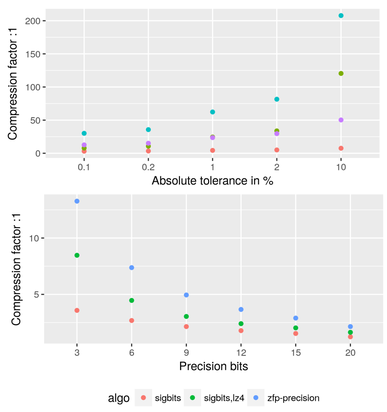
- absolute throughput in MiB or GiB
- relative to network or storage speed

To allow setting these values, the system's performance must be trained initially.

Tolerance-Based Results

The graphs show the mean compression factor for all scientific data files varying the precision for the algorithms ZFP, SZ, SIGBITS (keeps the exponent and significant bits from the mantissa) and ABSTOL (tolerance set to % of the max value in a data set). An optional LZ4 stage increases compression for ABSTOL and SIGBITS.

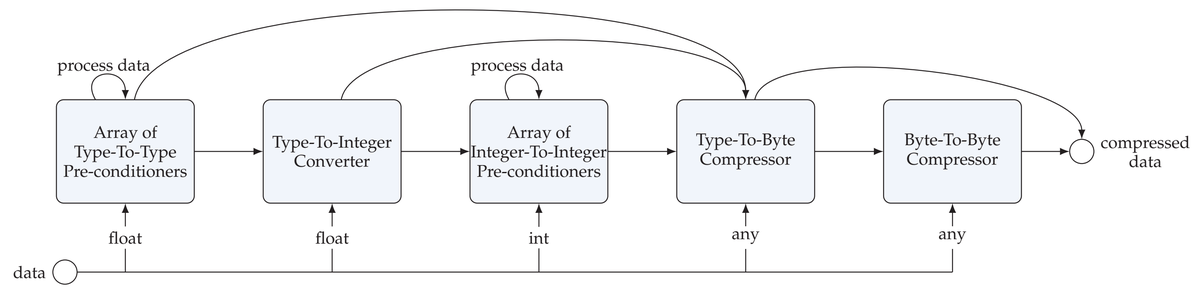
Mean is computed based on the sum of the data size, e.g., after compression a factor of 50:1 means all occupied space is reduced to 2% of the original size.



Architecture and Status of SCIL

SCIL is implemented in C and utilizes existing compression libraries and tools such as LZ4, ZFP, and SZ. There is a prototypical integration into HDF5 and NetCDF available. The implementation for the automatic algorithm selection is ongoing effort.

Compression chain. A chain of algorithms can be constructed that is processed internally. Based on the basic data type that is supplied, the initial stage of the chain is entered. For floating point data and integer data, pre-conditioners or a final LZ4 step can be supplied. Intermediate steps can be skipped.

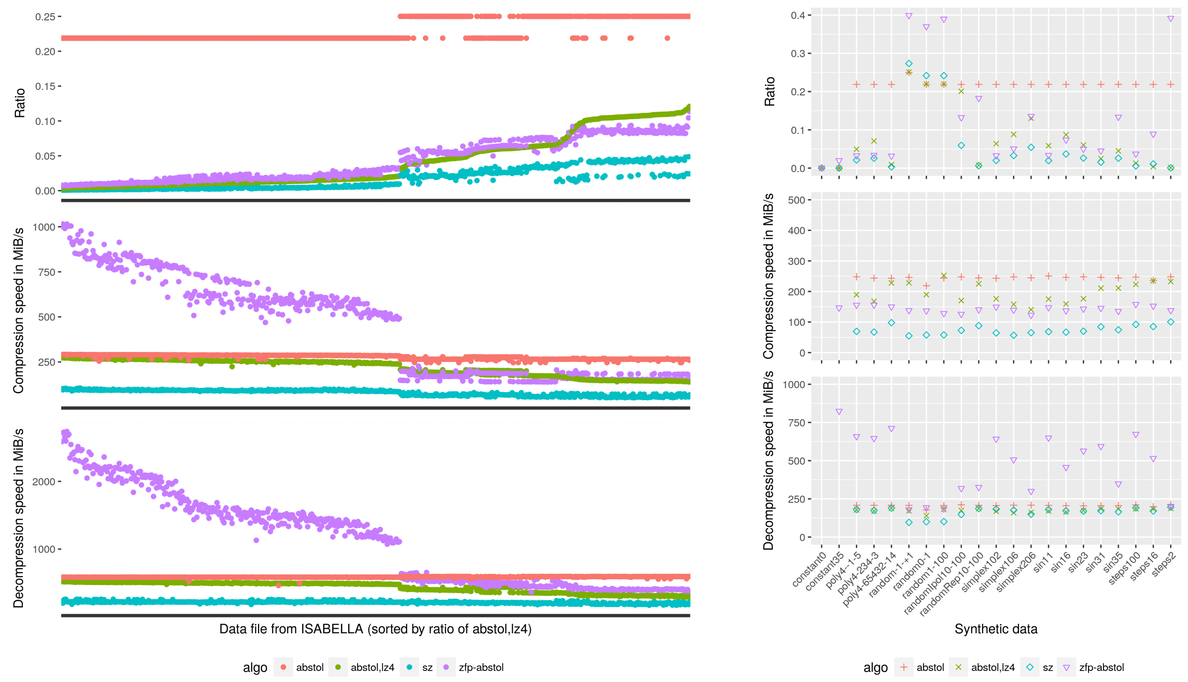


Tools. SCIL comes with additional tools useful for the evaluation:

- Creating several relevant multi-dimensional data patterns of any size
- Adding random noise based on the hint set to existing data
- To evaluate compression on existing CSV and NetCDF data files

Results for Absolute Tolerance

Comparing algorithms using an absolute tolerance of 1% of the maximum value:



The synthetic random patterns can serve as baseline to understand the benefit of the lossy compression. For abstol, a random pattern yields a factor of 4.3:1. Arithmetic mean (ratio, throughput MiB/s)

Algorithm	Ratio	Compr.	Decomp.
abstol	0.230	277	587
abstol,lz4	0.041	206	420
sz	0.016	76	213
zfp-abstol	0.042	274	742

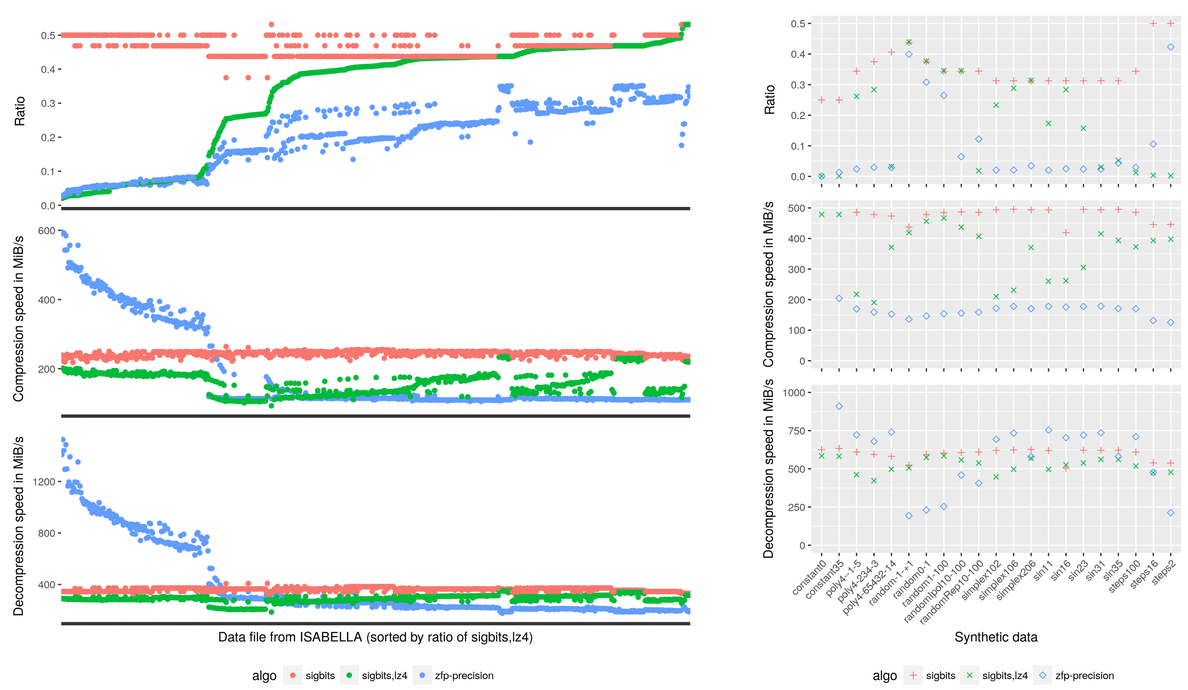
Table 1: For hurricane Isabel data files

Algorithm	Ratio	Compr.	Decomp.
abstol	0.225	240	201
abstol,lz4	0.180	209	169
sz	0.165	64	118
zfp-abstol	0.295	134	224

Table 2: For 5 different random patterns

Results for Precision Bits

Comparing algorithms using 9 precision bits for the mantissa. Note that due to a different understanding of "precision" from ZFP (it defines the number of bits to retain in total), the number for ZFP has been set to be equal to the necessary bits for the exponent and the mantissa in the sigbits algorithm. The achieved precision of ZFP does mostly not hold the defined precision, still the results of sigbits is often better.



The synthetic random patterns can serve as baseline to understand the benefit of the lossy compression. For the new algorithm sigbits, a random pattern yields a factor of 2.6:1. Arithmetic mean (ratio, throughput in MiB/s):

Algorithm	Ratio	Compr.	Decomp.
sigbits	0.467	243	360
sigbits,lz4	0.329	152	285
zfp-precision	0.202	138	281

Table 3: For hurricane Isabel data files

Algorithm	Ratio	Compr.	Decomp.
sigbits	0.369	474	585
sigbits,lz4	0.305	436	550
zfp-precision	0.232	150	276

Table 4: For 5 different random patterns

Test System

- Intel(R) Core(TM) i3-2120 CPU
  - 4 cores @ 3.30GHz (only 1 used)
- 8GB DDR3 (1333 MHz)
- Ubutnu 16.04 LTS with GCC 5.4.0

Experiments

For each test data (CSV or BIN format), the following setups are run:

- Lossless compression
  - Algorithms: memcopy and lz4
- Lossy compression with absolute tolerance
  - Tolerance: 10%, 2%, 1%, 0.2%, 0.1% of the data maximum value
  - Algorithms: zfp, sz, abstol, abstol+lz4
- Lossy compression with significant bits
  - Tolerance: 3, 6, 9, 15, 20 bits
  - Algorithms: zfp, sigbits, sigbits+lz4

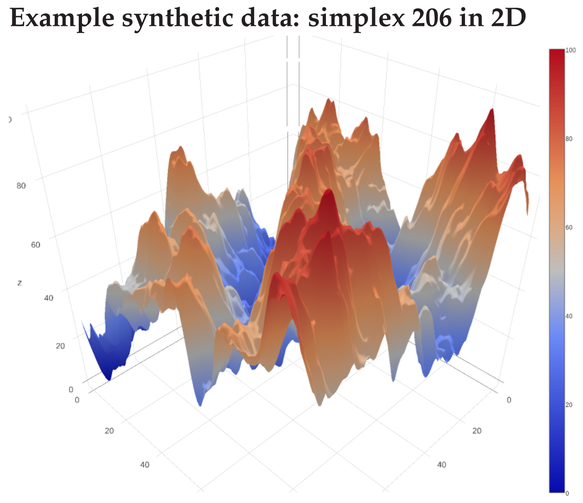
In the test, one thread of the system is used for the compression/decompression. A configuration is run 3x measuring compr/decompression time and compression ratio.

Test Data

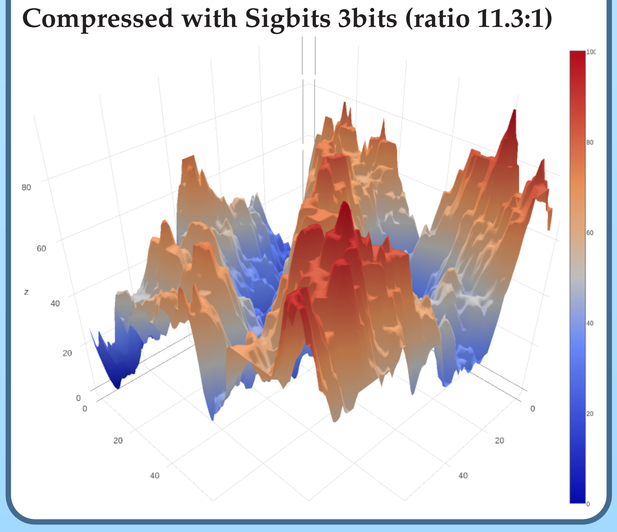
The pool of (double precision floating point) test data is build upon:

- Synthetic, generated by SCIL's pattern lib (e.g., Random, Steps, Sinus, Simplex).
- Hurricane Isabel data produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR, and the U.S. National Science Foundation (NSF).

Example synthetic data: simplex 206 in 2D



Compressed with Sigbits 3bits (ratio 11.3:1)



Summary

This poster provides first results for SCIL and compares novel algorithms implemented with the state-of-the-art compressors. It shows that these algorithms can compete with ZFP/SZ when setting the absolute tolerance or precision bits. In some cases, SZ compresses better than abstol and vice versa. Since SCIL aims to choose the best algorithm, it can take benefit of both algorithms.

Ongoing (future) work:

- A single algorithm honoring all quantities
- Automatic choose for the fitting algorithm

Acknowledgements

This work was supported in part by the German Research Foundation (DFG) through the Priority Programme 1648 "Software for Exascale Computing" (SPPEXA) (GZ: LU 1353/11-1).