Evaluating Lossy Compression on Climate Data*,**

Nathanael Hübbe[1], Al Wegener[2], Julian Martin Kunkel[1], Yi Ling[2], and Thomas Ludwig[3]

[1] University of Hamburg
nathanael.huebbe@informatik.uni-hamburg.de
[2] Samplify
AWegener@samplify.com
[3] German Climate Computing Centre

Abstract. While the amount of data used by today's high-performance computing (HPC) codes is huge, HPC users have not broadly adopted data compression techniques, apparently because of a fear that compression will either unacceptably degrade data quality or that compression will be too slow to be worth the effort. In this paper, we examine the effects of three lossy compression methods (GRIB2 encoding, GRIB2 using JPEG 2000 and LZMA, and the commercial Samplify APAX algorithm) on decompressed data quality, compression ratio, and processing time. A careful evaluation of selected lossy and lossless compression methods is conducted, assessing their influence on data quality, storage requirements and performance. The differences between input and decoded datasets are described and compared for the GRIB2 and APAX compression methods. Performance is measured using the compressed file sizes and the time spent on compression and decompression. Test data consists both of 9 synthetic data exposing compression behavior and 123 climate variables output from a climate model. The benefits of lossy compression for HPC systems are described and are related to our findings on data quality.

Keywords: Data Compression, GRIB2, JPEG 2000, APAX.

1 Introduction

Climate science is a notorius producer of big data. More than 100 climate variables are typically used in modern climate models, a number that cannot meaningfully be reduced, and the simulated time spans are often decades. Data presented in scientific publications must be archived for at least ten years. Consequently, large climate computing facilities like the German Climate Computing Center make significant investments in storage systems. Despite the amount of

* This paper is partly funded by the DFG (GZ: LU 1353/5-1).
** We also thank Luis Kornblueh for providing us with the climate dataset, without which this paper would not have been possible.

J.M. Kunkel, T. Ludwig, and H.W. Meuer (Eds.): ISC 2013, LNCS 7905, pp. 343–356, 2013.

climate data to be stored and transfered, climate scientists have been reluctant to use data compression to reduce dataset volumes. It seems the scientists have vague fears that lossless compression would be too slow to be worthwhile and that lossy compression would unacceptably reduce the quality of the data.

In our previous paper [4], we showed that the slowest lossless compression algorithms actually achieve the best compression. In this paper we measure the effects of two lossy compression algorithms on climate variable quality while also considering lossy compression's benefits in reducing HPC system bottlenecks.

This paper is structured as follows: First, related work and the lossy compression algorithms used are described. We then discuss signal quality requirements of climate scientists and describe the different synthetic and climate variables that are used to evaluate lossy compression. We summarize the metrics to quantify differences in precision. Then our test setup is described. After summarizing our findings, we use two distinct approaches to analyze the results. First, the processing speeds of the competing algorithms is characterized. Second, approaches to obtaining acceptable climate variable quality are described. Finally, future research directions are considered.

2 Related Work

Lossless compression can be profitably used if the costs for compression and decompression are less than the costs of bandwidth and storage. For example, lossless-compressed tarballs are regularly used for source code exchange, and the SLDC [3] algorithm increases both tape drive bandwidth and capacity. However, lossless compression algorithms developed for ASCII text do not compress binary datasets, such as most HPC datasets, very well. In contrast, a small number of targeted algorithms have been developed to compress floating-point HPC data. Current research into compression of scientific data generally takes one of two approaches: Either the performance of available algorithms is evaluated on specific scientific datasets as Woodring et al. [11] have done when they applied JPEG 2000 compression to climate data, or new lossy algorithms are developed that have specific features and/or perform well for specific kinds of data.

An example for such new lossy algorithms is *isabela*, invented by Lakshminarasimhan et al. [6], [7]. Another example is the recent *sengcom* algorithm [9], which has strong similarities to the GRIB2/JPEG 2000 compression described below. Some algorithms take the multidimensionality of scientific datasets into account, such as the work by Lindstrom and Isenburg [8]. Iverson et al. [5] are exploiting data locality on unstructured grids, especially for geo-sciences. Our own last endeavor at lossless scientific data compression [4] handled diverse multidimensional datasets. MAFISC uses the standard lossless compression algorithm LZMA as a compression back-end after the MAFISC front-end transforms the data in a reversible way.

2.1 Lossy Compression in GRIB2

GRIB2 [2] is a format defined by the World Meteorological Organization that is based on self-describing messages using standardized values to identify basic

properties of the data. Examples for such properties are grid types, intended meaning of the data, and data encoding formats.

As a file format standard, the GRIB2 format itself does not specify how the data is encoded, only how the encoded data should be interpreted. This leaves a number of decisions to the authors of programs producing GRIB2 data.

GRIB data formats are based on fixed point (integer) representation that includes a conversion from the floating point representation that causes a loss of precision. The quantization parameters are selected by the encoding software according to a user choice and are kept in the header of the encoded data. In most cases, the user specifies how many bits of precision should be retained. The encoding software then scans the input data for its value range and adjusts the quantization accordingly. In this way, the user controls both signal quality and file size, resulting in lossy compression.

As provided in current implementations, GRIB2 quantization is time-adaptive since it is performed for each timeslice and elevation level seperately. The resulting quantization error places a tight upper bound on the maximum error. Finally, the quantized GRIB2 integer result can be further compressed by JPEG 2000 [1] in its lossless mode.

2.2 APAX

Figure 1 presents a block diagram of the Samplify APplication AXceleration (APAX) Encoder. The APAX algorithm encodes sequential blocks of input data elements with user-selected block size between 64 and 16,384. The signal monitor tracks the input dataset's center frequency. The attenuator multiplies each input sample by a floating-point value that, in fixed-rate mode, varies from block to block under the control of an adaptive control loop that converges to the user's target compression ratio. The redundancy remover generates derivatives of the attenuated data series and determines which of the derivatives encodes using the fewest bits. The bit packer encodes groups of 4 successive samples using a *joint exponent encoder* (JEE). JEE exploits the fact that block floating-point exponents are highly correlated. Additional APAX details are described in [10].

The APAX encoder uses a software tool called the APAX profiler that provides information about the compressibility of input datasets, and recommends

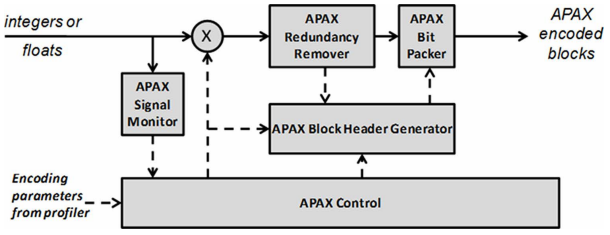


Fig. 1. APAX block diagram

a compression setting that delivers high-quality decompressed samples. Figure 2 illustrates the APAX profiler output on climate variable *ustrl*. The upper left window displays the rate-distortion graph for the input signal being profiled. The profiler suggests a Recommended Operating Point (ROP) where the correlation between the original data x and the APAX-decoded data y is 0.99999 ("five nines"). The upper right window displays metrics comparing input $x(i)$, decoded output $y(i)$, and residual or difference $d(i) = x(i) - y(i)$. The lower left window compares the input spectrum to the residual spectrum and quantifies the spectral margin at the ROP. The lower right window histograms the residuals and calculates the $2 \cdot \sigma$ (95.5%) signal-to-residual margin.

2.3 Comparing GRIB2 and APAX Control

We notice that GRIB2 and APAX are controlled in fundamentally different ways. GRIB2 users must choose the quantization level N, and whether or when to use JPEG2000. GRIB2 parameter choices directly affect the quality of decoded climate variables and the speed with which they are encoded and decoded. The APAX Profiler visually displays the rate-distortion curve on the climate variable being profiled and recommends a compression ratio, helping users in their decision process. Once the user has chosen the compression ratio, that setting is again accessed as APAX encodes that climate variable. The Profiler's default ROP ensures consistent signal quality while allowing sophisticated users to modify the ROP and to visualize and measure the new ROP's effect on signal quality.

3 Quantifying the Uncertainty

3.1 Scientific Requirements

To determine appropriate climate data signal quality requirements, one must understand that climate data will be used in two ways:

- The data is analyzed or visualized for evaluating long-term effects, such as variations in average, variance, frequency, or locality of events.
- Data may also be used to drive another model (or to keep checkpoints).

In the first case, good quality data will not introduce any significant statistical variation. No new effects should be created that were not present in the original data, and no previously visible effects should vanish. To be safe, the maximum error (worst-case scenario) should be monitored in addition to its standard deviation to check whether the error is still guaranteed to be below the required threshold. The second case cannot be evaluated as easily, because climate systems are inherently chaotic. A small change in the input data may either vanish completely or can lead to a completely different state a year later. It is impossible to consistently predict whether an error in the input data caused by lossy compression would cause different simulation effects than a random error would not have caused.

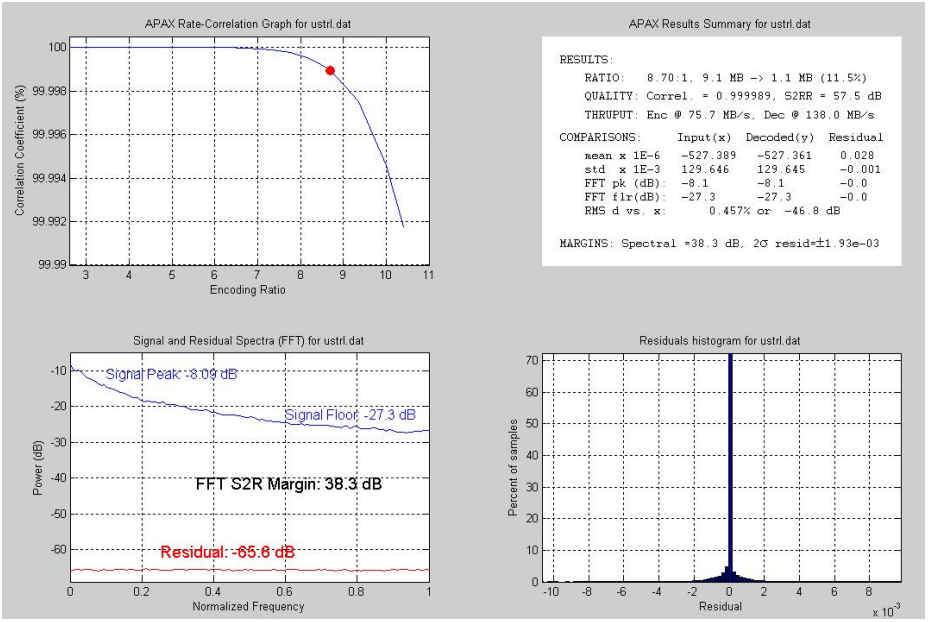


Fig. 2. APAX profiler window

In both cases, errors should not be locally correlated because such correlated errors can give rise to false positives (much in the way that overcompressed JPEG images exhibit block edges that were not present in the original image). When decompressed data drives a climate simulation, locally correlated errors are much more likely to drive the simulation into a state that would not have been reached with uncorrelated errors. In the case of checkpoints, lossy compression is usually not an option, since their goal is to allow a precise restart after a crash. Thus our signal quality metrics will measure both average and worst-case differences between the input and the decompressed datasets, as the compression ratio is varied over a range from 24/32 (75%) to 8/32 (25%).

3.2 Approach

Description of Test Cases. We have tested the the GRIB2 and APAX compressors using 9 synthetic datasets and with 123 climate output variables generated from the ECHAM climate model from the Max-Planck-Institute for Meteorology. The synthetic files are as follows:

bandlim_lowpass 1D, lowpass filtered random data.
bandlim_narrow 1D, bandpass filtered random data.
random 3D random data, flat distribution in the intervall [-1,1].
random_offset 3D random data with an offset of 1 added.

random_correlated 3D random data with 90% correlation to the mean of the three previously generated neighbours.

fractal 3D hierarchically generated data with fractal dimensionality. 2D slices resemble a mountaneous height field.

integrated 3D random data, integrated twice in all directions. Very smooth.

sines_orthogonal 3D cube with one sine per axis in superposition.

sines_random 3D cube with 100 sines of random direction and frequency in superposition.

All the 3D test cases were generated with $257 \times 257 \times 257$ grid points, producing NetCDF files with 68 MB of data. The climate model output consisted of 123 different variables, covering one month at a sample frequency of six hours (124 timesteps). The longitude/latitude grid covers the entire earth with 192×96 grid cells and 47 height levels. Some variables are expressed as 2080 spectral coefficients. The entire climate dataset contained 4.4 GB of data.

Description of Error Metrics. Let us consider the characteristics of the residual signals $r(i)$ generated by GRIB2 and APAX encoding. Since lossy compression always generates non-zero residuals $r(i)$, we should first describe the preferred characteristics of residuals. First, the magnitude of $r(i)$ should be as small as possible, in both a relative sense (minimize $r(i)/x(i)$) and also in an absolute, peak error sense (minimize $r(i)$). Second, residuals should be zero-mean, i.e. $E[r] = 0.0$. Third, residuals should be spectrally white, i.e. the residual's power spectral density $psd(r)$ should be flat from DC to Nyquist (half the sampling rate). Fourth, residuals should be uncorrelated with the signal from which they are generated. Point three and four are generally not met by lossy compression algorithms, including APAX and GRIB2. Both GRIB2 and APAX normalize floating-point input values and then quantize them to integers, so both algorithms generate residuals with similar characteristics. However, the residual characteristics are not identical, since APAX can change the quantization from block to block.

In the order in which climate dataset values are stored in memory, the standard deviation of residuals, $std(r)$, provides a direct metric of signal quality. When the residuals are spectrally white, $std(r)$ is proportional to $psd(r)$. We calculated the signal-to-residual ratio (SRR), in bits, as given in Equation 1. Since compression users are concerned both with average and worst-case signal quality, we also measured the largest residual magnitude $max(abs(r))$, and compare it to the range of input values, $max(x)$–$min(x)$, calculating our peak error metric, in bits, as given in Equation 2.

$$SRR = log_2\left(\frac{std(x)}{std(r)}\right) \tag{1}$$

$$PrecisionBits = log_2\left(\frac{max(x) - min(x)}{2 \cdot max(abs(r))}\right) \tag{2}$$

In cases where a particular climate variable contains so little numerical variation that the decoded signal is identical to the original signal, it is possible to have an "infinite" SRR. These cases are especially likely at lower compression ratios such as $N = 24$ (75%). In such cases, the residual samples are all zero and both SRR and PrecisionBits are infinite. To avoid calculations using such infinite values, we limit the maximum SRR and PrecisionBits to 50 bits.

4 Evaluation

4.1 Speed

Test Setup. Compression and decompression performance tests were run on GRIB2 and APAX using a 48-core Magny-Cours node with 128 GiB of memory and 1.9 GHz clock frequency. Since the objective was to measure the performance of the algorithms and not the performance of the disks, all input files were first copied to a RAM-disk. The compression/decompression was then performed with both the input and the output file on the RAM-disk and measured using the `time` utility. The measured GRIB2 times are the sum of the system and user times reported by `time`. Timing measurements were automated using a shell script for all measurements. APAX performance was measured using in-process timers measuring only the compression from memory buffer to memory buffer, while the GRIB2 measurements encompas the entire process, including startup times, filesystem calls and library overhead. Thus, the timings are not comparable between GRIB2 and APAX.

Results. Figure 3 compares the performance of the different algorithms. Each point quantifies the performance of one algorithm on one file. Each algorithm is represented using a colored, shaped icon. The x-axis represents compression throughput (sec/GB), while the y-axis represents compression factor.

It is interesting to see that the external LZMA compression of GRIB2 files tends to be faster than the builtin JPEG 2000 compression, which performs as slow as MAFISC in many cases. MAFISC exhibits the slowest processing times on some files. Figure 3 illustrates a roughly linear correlation between the compressability of a file and the time it takes to be encoded by GRIB2. Generally, we see a strong correlation between compressability and speed, the only exception to this is APAX, its speed solely depends on data characteristics and, to a minor degree, target compression ratio. Unfortunately, this is not visible in the graphics. Only the LZMA utility may take considerably more time on some files than for other files with similar compressability, with the LZMA based MAFISC compression this shows even more clearly. APAX throughput is about 152 MB/sec (6.58 sec/GB) for compression and 209 MB/sec (4.79 sec/GB) for decompression, measured from memory buffer to memory buffer, all other measurements include filesystem access to a Ramdisk as well. These averages are calculated from total uncompressed size and total processing time.

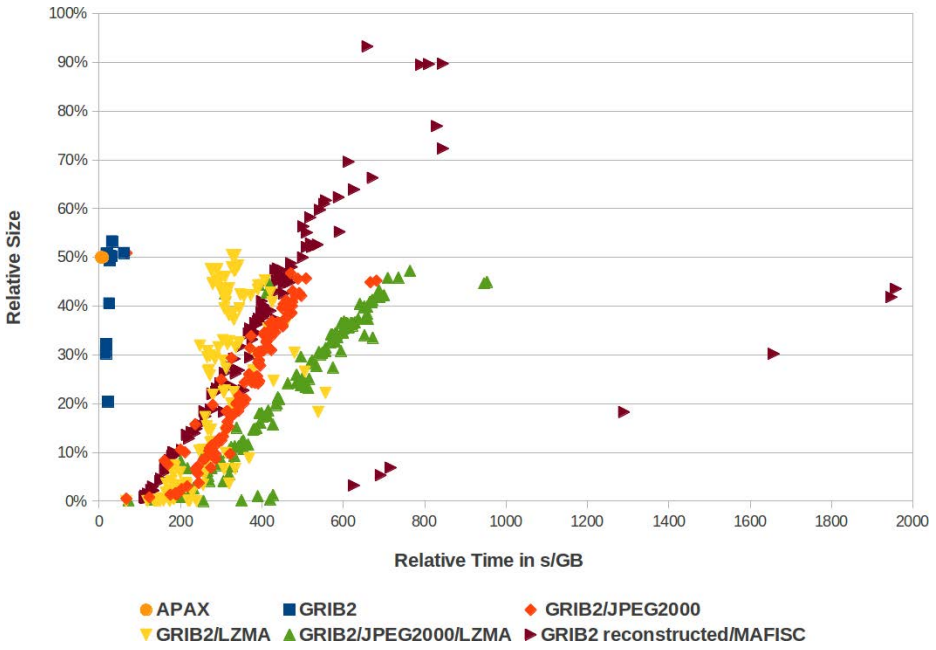


Fig. 3. Comparison of algorithm throughput (sec/GB) vs. achieved compression. Note that the measurement conditions for APAX were not the same as for the other methods.

It is clear that APAX and GRIB2 are the fastest algorithms in the field. GRIB2 and APAX have comparable compression and decompression throughput, and even though GRIB2 performs fewer calculations, APAX appears to be the faster of the two.

4.2 Compression

Apart from the compression time, the plot in Figure 3 also reveals the compression ratio of the files. It is clear that a lot of the redundancy of the data remains in the GRIB2 encoded files. This is demonstrated by strong additional lossless compression that can be achieved using JPEG 2000 or LZMA. As shown in Figure 3, the additional lossless compression benefits from JPEG2000 and LZMA are achieved at the expense of slower processing speed. Both JPEG 2000 and LZMA add between 100 and 500 s/GB to the GRIB2 encoding time for $N = 16$, in most cases, however, LZMA takes less time than JPEG 2000 compression.

Figure 4 compares the compression ratio of GRIB2/JPEG 2000, GRIB2/LZMA and GRIB2/JPEG 2000/LZMA The files were sorted according to their GRIB2/LZMA compression, the GRIB2 quality was set to 22 bits, but the results for other sizes are comparable. Figure 4 is interesting for a number of reasons:

- Neither JPEG 2000 nor LZMA can be said to be better than the other, each significantly outperforms the other on a large number of files.
- Whether JPEG 2000 or LZMA provides more compression is strongly correlated with the LZMA compressability.
- Even though the JPEG 2000 output is next to incompressible for most files, in some cases, LZMA applied on GRIB2/JPEG 2000 output still has a very strong effect. These are data sets, where a number of timeslices/height levels are identical or differ only by an offset and a factor. In these cases, the JPEG 2000 output is identical because its input is identical. LZMA finds these repeated regions in the file and achieves additional compression.

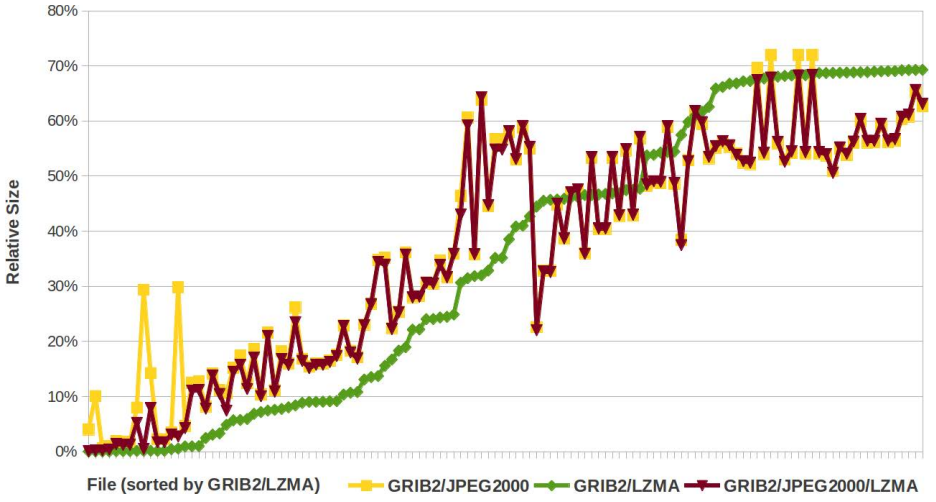
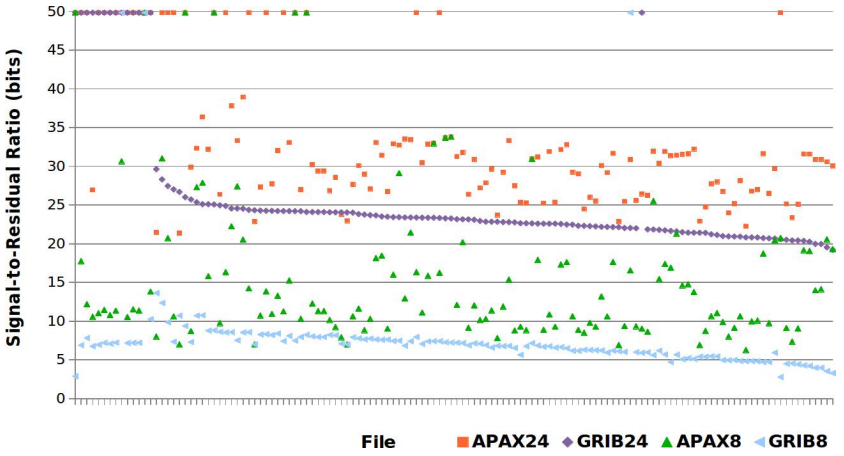


Fig. 4. Compression factor of lossless algorithms on GRIB2 data for 22 bits

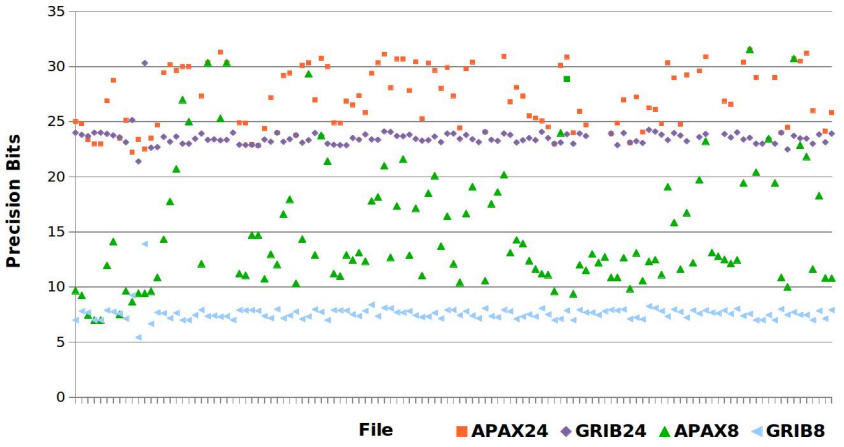
4.3 Errors

In this analysis we compare the loss in precision of compressed values between APAX and GRIB for 8 bits (4 : 1) and 24 bits (1.33 : 1) precision. Since LZMA, MAFISC and JPEG2000 are lossless, these algorithms do not alter the quality of the compressed data.

Figure 5a illustrates the SRR average signal quality metric for each of the 132 datasets. Larger SRR values indicate better decompressed signal quality. Visually, Figure 5a demonstrates that APAX SRR values generally exceed GRIB2 SRR values, with a few exceptions. Figure 5b illustrates the PrecisionBits peak error metric, measured under the same conditions as Figure 5a's SRR metric. Larger PrecisionBits values indicate better decompressed signal quality. Visually, Figure 5b demonstrates that APAX PrecisionBits values generally exceed GRIB2 values, with a few exceptions.



(a) Signal-to-Residual Ratios



(b) PrecisionBits

Fig. 5. Comparison of GRIB and APAX at N=8 and N=24

Table 1 compares the resulting signal quality on 10 climate files, when comparing GRIB2 quantization at 22 bits followed by JPEG 2000 encoding of the GRIB2 output, to APAX at the equivalent compression ratio. For instance, while GRIB2 compression of the climate variable alcov achieves a compression ratio of 68.75% (22/32), JPEG 2000 further compresses the 22-bit GRIB2 values to a compression ratio of 64%. APAX was directed to achieve the same compression ratio (64%) as GRIB2/JPEG 2000, and the quality of both results was compared. Unfortunately APAX could not be instructed to operate at a compression ratio of less than 2.9% on the alsom file, so the following considerations do not take this file into account.

Table 1. GRIB2/JPEG2000 and APAX Signal Quality Metrics at the same compression ratios. N=22 for GRIB2/JPEG2000 files.

		SRR		PrecisionBits	
file	rel. size	APAX	GRIB2/JPEG 2000	APAX	GRIB2/JPEG 2000
trads	65.4%	20.7	20.8	20.3	21.1
aclcov	64%	23.2	22.1	21.0	21.0
trafl	56.5%	20.8	21.4	20.4	22.0
trflwac	53.2%	21.5	20.9	20.2	21.8
soflwac	35.8%	21.0	22.0	18.7	22.0
wsmx	29.3%	23.7	21.6	21.8	21.8
ahfllac	28%	21.7	19.0	21.1	21.7
vdisgw	22.9%	24.6	19.6	22.9	21.9
srad0d	22.6%	13.5	21.6	12.2	21.3
alsom	2.9%/1.9%	lossless	22.8	lossless	21.5

Looking at the SRR metric, each of the two compression methods achieves lower errors for some files than the other. This changes, however, if the maximal error is taken into account, which is used for the PrecisionBits metric. With this metric, APAX only has an advantage over GRIB2/JPEG 2000 on one of nine files, for the rest it achieves at most the same precision level. This one file, however, is a file on which GRIB2/LZMA results in output only half as big as the GRIB2/JPEG 2000 result.

In addition to comparing GRIB2 and APAX signal quality at the same compression ratio, we also examined compression ratio at comparable SRR quality levels. Using APAX's fixed quality mode, APAX compression ratios vary from dataset to dataset, because APAX encodes the most compressible derivative from among three alternatives, for each input block. With a fixed quality of SRR=14 bits, APAX averaged 1.6x more compression than GRIB2 with N=16 (wich yields SRR values around 15 bits). For certain files, GRIB2 compression is improved by as much as 60:1 by JPEG 2000 post-processing, and by as much as 3000:1 by LZMA post-processing.

5 Analysis of the Differences

APAX appears to be the fastest algorithm. GRIB2 can be fast as well, but how fast it actually is depends heavily on its precision parameter: N = 8, 16 or 24 bits yields much better performance than any other setting for obvious reasons. Even though GRIB2 is the simpler algorithm, APAX appears to be faster; we believe that this is due to the fact that GRIB2 has to scan the input data twice (once to compute the data range and once to do the conversion) while APAX is a single pass algorithm, which leads to better cache usage. The GRIB2 encoder available from the Max-Planck-Institute for Meteorology is said to be a much more optimized encoder than the WMO GRIB2 encoder we used in this paper, but time did not allow us to verify these claims.

Comparing these two fast algorithms at the same compression level without LZMA or JPEG 2000 post-processing, we find that APAX signal quality exceeded GRIB2 signal quality for more than 85% of all variables in our dataset. At N = 24 / 16 / 8, APAX SRR signal quality averages 7 / 11 / 8 bits better than GRIB2, respectively. In some cases, however, GRIB2 generates better SRR and PrecisionBits metrics than APAX on the least compressible datasets that exhibit very few repeated values, including the datasets random, random_offset, sines_orthogonal, lsp, st and az0w. Since APAX needs more control bits than GRIB2 to describe what the encoder has done, these bits are consequently not available to encode the data itself, and which provide no benefit when all alternatives are equally bad. So, even if GRIB2 were as fast or faster than APAX, in this speed class APAX is the better encoding for most climate variables due to its supperior compression or data quality.

If execution speed is not as much of a concern, the GRIB2/JPEG 2000 and GRIB2/LZMA combinations come into play. While neither can outperform the other on the majority of variables, both profit from the tight error guarantees of the GRIB2 format, indicated by their better PrecisionBits results compared to APAX, and for most variables one combination clearly outperforms the other. So, while it is clear, that one of these combinations should be used when good compression is more important than speed, the decision which of the two to use should be made on a per variable basis.

6 Compression Use Cases and Benefits for HPC

The most easily obtained benefit from lossy compression of climate datasets is a significant reduction in disk file size and a corresponding increase in disk bandwidth. Compared to lossless compression, both GRIB2 and APAX lossy compression can achieve significantly higher compression ratios with acceptable quality, as described in Section 4. Both GRIB2 and APAX are fast enough to saturate typical filesystems, in HPC settings with high throughput parallel filesystems, several cores might be neccessary, though. For archiving applications where processing speed is not critical, the combination of GRIB2 and JPEG 2000 provides slightly better signal quality than APAX and is thus the preferred solution.

As climate simulation resolution improves, and as HPC core counts continue to increase, lossy compression could also be used to reduce other system bottlenecks, including PCIe, Infiniband and Ethernet links for data exchange, and DDR memory bottlenecks that store increasingly large climate datasets. In these cases, compression performance (sec/GB) becomes critical. Bus and networks speeds can reach 56 Gbps (FDR Infiniband), while sustained HPC server DDR3 memory throughput now achieves 5 GB/sec. If lossy compression could be used, simulations could complete faster as if they were using faster network and memory. As HPC core counts have increased, overall core utilization has decreased (sometimes to below 20% of peak MIPS), so significant CPU cycles could be used for compressing and decompressing datasets in DDR memory. This, of course,

would require a very tight coupling between the simulation code and the compression/decompression code.

Climate scientists will have to trade off where such *in situ* compression would add the most benefits while maintaining climate simulation results. Due to the statistical nature of climate research, input and output datasets could probably be lossy compressed. For intermediate climate simulation results, climate scientists will want to carefully monitor the numerical stability and consistency of results as they evaluate compression's artifacts. Most intermediate results will likely require lossless handling. However, even lossless compression schemes might accelerate simulations if they are fast and applied to the more compressible variables.

7 Summary and Future Work

We have compared GRIB2 (with and without optional JPEG2000 and LZMA post-processors) and the APAX lossy compression algorithms on synthetic and climate datasets. At equivalent compression ratios, APAX signal quality exceeds GRIB2 signal quality for most climate variables. On some climate datasets, GRIB2 compression ratios are improved by 60x (JPEG2000) to 3800x (LZMA). GRIB2 and APAX processing speeds are comparable to each other, and both are at least 10x faster, and often 100x faster, than when GRIB2 includes JPEG2000 or LZMA post-processing. In the future, we plan to investigate how much APAX compression would be improved by JPEG2000 and LZMA post-processing. We also plan to involve climate scientists in quantifying the acceleration in "time to results" that the fast GRIB2 and APAX algorithms could provide and analyze the required precision, by increasing HPC memory and disk capacity and bandwidth.

References

1. Christopoulos, C., Skodras, A., Ebrahimi, T.: The JPEG2000 still image coding system: an overview. IEEE Transactions on Consumer Electronics 46(4), 1103–1127 (2000)
2. Dey, C., et al.: Guide to the WMO Table Driven Code Form Used for the Representation and Exchange of Regularly Spaced Dat. In: Binary Form: FM 92 GRIB Edition 2. Tech. rep., World Meteorological Organization (2007), http://www.wmo.int/pages/prog/www/WMOCodes/Guides/GRIB/GRIB2_062006.pdf
3. ECMA: Streaming lossless data compression algorithm - (sldc), ECMA Standart 321 (2001)
4. Hübbe, N., Kunkel, J.: Reducing the HPC-Datastorage Footprint with MAFISC - Multidimensional Adaptive Filtering Improved Scientific data Compression. In: Computer Science - Research and Development. Executive Committee. Springer, Heidelberg (2012), doi:http://dx.doi.org/10.1007/s00450-012-0222-4

5. Iverson, J., Kamath, C., Karypis, G.: Fast and effective lossy compression algorithms for scientific datasets. In: Kaklamanis, C., Papatheodorou, T., Spirakis, P.G. (eds.) Euro-Par 2012. LNCS, vol. 7484, pp. 843–856. Springer, Heidelberg (2012)
6. Lakshminarasimhan, S., Shah, N., Ethier, S., Klasky, S., Latham, R., Ross, R., Samatova, N.F.: Compressing the incompressible with ISABELA: In-situ reduction of spatio-temporal data. In: Jeannot, E., Namyst, R., Roman, J. (eds.) Euro-Par 2011, Part I. LNCS, vol. 6852, pp. 366–379. Springer, Heidelberg (2011)
7. Lakshminarasimhan, S., Shah, N., Ethier, S., Ku, S.H., Chang, C.S., Klasky, S., Latham, R., Ross, R., Samatova, N.F.: Isabela for effective in situ compression of scientific data. Concurrency and Computation: Practice and Experience (2012)
8. Lindstrom, P., Isenburg, M.: Fast and efficient compression of floating-point data. IEEE Transactions on Visualization and Computer Graphics 12(5), 1245–1250 (2006)
9. Sullivan, S.: Wavelet compression for floating point data–sengcom. Tech. rep., University Corporation for Atmospheric Research (2012), `http://www.unidata.ucar.edu/software/netcdf/papers/sengcom.pdf`
10. Wegener, A.: Adaptive compression and decompression of bandlimited signals. US patent 7,009,533 (2006)
11. Woodring, J., Mniszewski, S., Brislawn, C., DeMarle, D., Ahrens, J.: Revisiting wavelet compression for large-scale climate data using JPEG2000 and ensuring data precision. In: 2011 IEEE Symposium on Large Data Analysis and Visualization (LDAV), pp. 31–38 (2011), doi:10.1109/LDAV.2011.6092314