Compressing atmospheric data into its real information content

Milan Kloewer (milan.kloewer@physics.ox.ac.uk)
University of Oxford https://orcid.org/0000-0002-3920-4356
Miha Razinger
European Centre for Medium-Range Weather Forecasts
Juan Dominguez
European Centre for Medium-Range Weather Forecasts
Peter Dueben
European Centre for Medium-Range Weather Forecasts
Tim Palmer
University of Oxford

Article

Keywords: atmospheric data, computational science, weather forecast data, climate forecast data

Posted Date: June 21st, 2021

DOI: https://doi.org/10.21203/rs.3.rs-590601/v1

Version of Record: A version of this preprint was published at Nature Computational Science on November 1st, 2021. See the published version at https://doi.org/10.1038/s43588-021-00156-2.

Compressing atmospheric data into its real information content

Milan Klöwer[1,*], Miha Razinger[2], Juan J. Dominguez[2], Peter D. Düben[2] and Tim N. Palmer[1]

[1]Atmospheric, Oceanic and Planetary Physics, University of Oxford, Oxford, UK
[2]European Centre for Medium-Range Weather Forecasts, Reading, UK

[*]Corresponding author: milan.kloewer@physics.ox.ac.uk

Hundreds of petabytes of data are produced annually at weather and climate forecast centres worldwide. Compression is inevitable to reduce storage and to facilitate data sharing. Current techniques do not distinguish the real from the false information in data. We define the bitwise real information content from information theory for data from the Copernicus Atmospheric Monitoring Service (CAMS). Most variables contain less than 7 bits of real information per value, which are also highly compressible due to spatio-temporal correlation. Rounding bits without real information to zero facilitates lossless compression algorithms and encodes the uncertainty within the data itself. The entire CAMS data is compressed by a factor of 17x, relative to 64-bit floats, while preserving 99% of real information. Combined with 4-dimensional compression to exploit the spatio-temporal correlation, factors beyond 60x are achieved without an increase in forecast errors. A data compression Turing test is proposed to optimize compressibility while minimizing information loss for the end use of weather and climate forecast data.

Many supercomputing centres in the world perform operational weather and climate simulations several times per day[1]. The European Centre for Medium-Range Weather Forecasts (ECMWF) produces 230TB of data on a typical day and most of the data is stored on magnetic tapes in its archive. The data production is predicted to quadruple within the next decade due to an increased spatial resolution of the forecast model[2–4]. Initiatives towards operational predictions with global storm-resolving simulations, such as Destination Earth[5] or DYAMOND[6], with a grid spacing of a couple of kilometres will further increase data volume. This data describes physical and chemical variables of atmosphere, ocean and land in up to 6 dimensions: three in space, time, forecast lead time, and the ensemble dimension. The latter results from calculating an ensemble of forecasts to estimate the uncertainty of predictions[7,8]. Most geophysical and geochemical variables are highly correlated in all of those dimensions, a property that is rarely exploited for climate data compression, although multidimensional compressors are being developed[9–12].

Floating-point numbers are the standard to represent real numbers in binary form. 64-bit double precision floating-point numbers (Float64) consist of a sign bit, 11 exponent

bits representing a power of two, and 52 mantissa bits allowing for 16 decimal places of precision across more than 600 orders of magnitude[13]. Most weather and climate models are based on Float64 arithmetics, which has been questioned as the transition to 32-bit single precision floats (Float32) does not necessarily decrease the quality of forecasts[14,15]. Many bits in Float32 only contain a limited amount of information as even 16-bit arithmetic has been shown to be sufficient for parts of weather and climate applications[16–19]. The information, as defined by Shannon information theory[20,21], for simple chaotic dynamical systems is often zero for many of the 32 bits in Float32[22]. This supports the general concept of low-precision climate modelling for calculations and data storage, as, at least in theory, many rounding errors are entirely masked by other uncertainties in the chaotic climate system[23,24].

Data compression for floating-point numbers often poses a trade-off in size, precision and speed[25–27]: Higher compression factors for smaller file sizes can be achieved with lossy compression, which reduces the precision and introduces rounding errors. Additionally, higher compression requires more sophisticated compression algorithms, which can decrease compression and/or decompression speed. A reduction in precision is not necessarily a loss of real information, as occurring rounding errors are relative to a reference that itself comes with uncertainty. Here, we calculate the bitwise real information content[20–22] of atmospheric data to discard bits that contain no information[28,29] and only compress the real information content. Combined with modern compression algorithms[10,30–32] the multi-dimensional correlation of climate data is exploited for higher compression efficiency[33,34].

Drawbacks of the current compression methods

The Copernicus Atmospheric Monitoring Service[35] (CAMS) is performing operational predictions with an extended version of the Integrated Forecasting System IFS, the global atmospheric forecast model implemented by ECMWF. CAMS includes various atmospheric composition variables, like aerosols, trace and greenhouse gases that are important to monitor global air quality. The system monitors for example the spread of volcanic eruptions or emissions from wildfires. Most variables in CAMS have a multi-modal statistical distribution, spanning many orders of magnitude (Fig. S1).

The current compression technique for CAMS is the linear quantization, widely used in the weather and climate community through the data format GRIB2[36]. CAMS uses the 24-bit version, which encodes values in a data array with integers from 0 to 2^{24}-1. These 24-bit unsigned integers represent values linearly distributed in the min-max range. Unused sign or exponent bits from the floating-point representation are therefore avoided and some of the trailing mantissa bits are discarded in quantization. Choosing the number of

bits for quantization fixes the file size, but the precision follows implicitly, leaving the required precision or amount of preserved information unassessed.

Although linear quantization bounds the absolute error, its linear distribution is unsuited for most variables in CAMS: Many of the available 24 bits are effectively unused as the distribution of the data and the quantized values match poorly (Fig. S2). Alternatively, placing the quantized values logarithmically in the min-max range better resolves the data distribution (Fig. S2). As floating-point numbers are already approximately logarithmically distributed, this motivates compression directly within the float format, which is also used for calculations in a weather or climate model and post-processing.

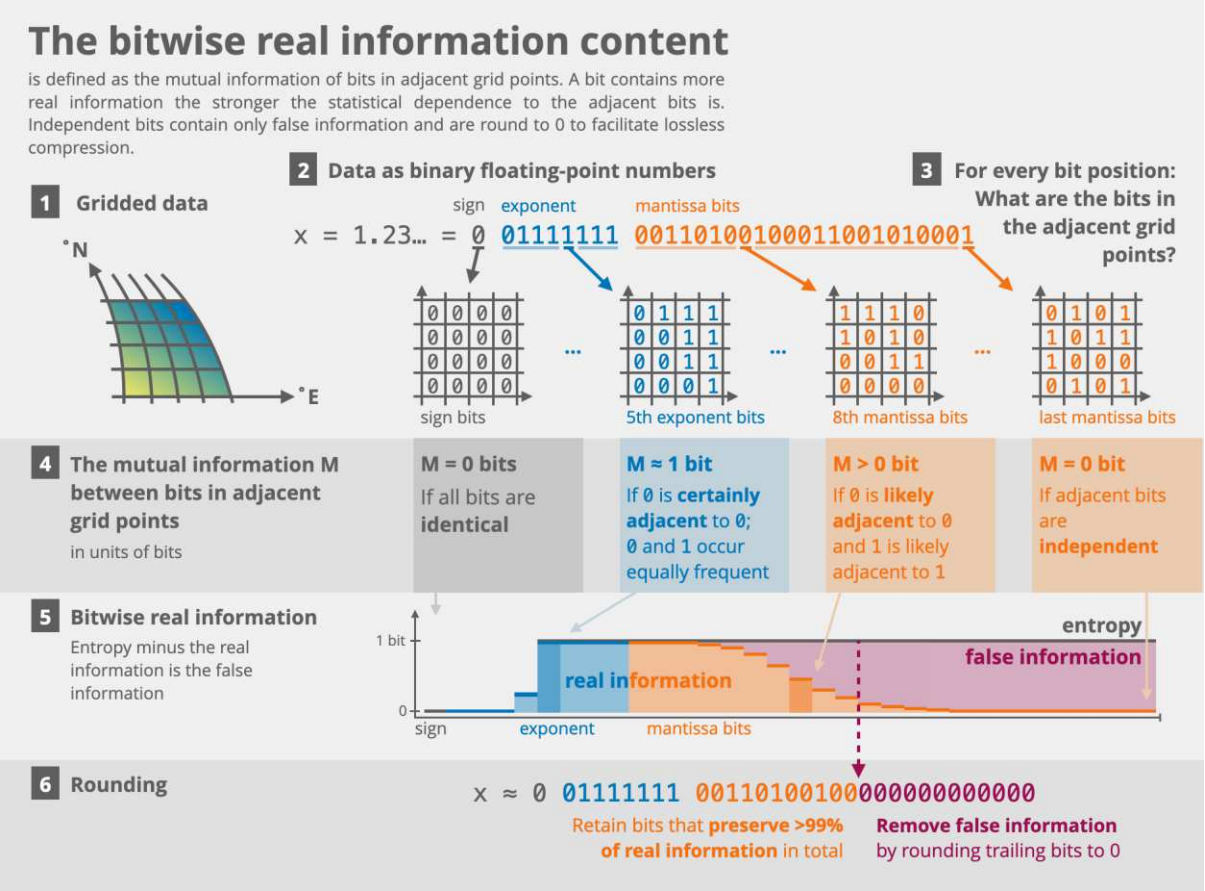

Fig. 1 | The bitwise real information content explained schematically.

Bitwise real information content

Many of the trailing mantissa bits in floating-point numbers occur independently and at similar probability, i.e. with high information entropy[21,22]. These seemingly random bits are incompressible[37–39], reducing the efficiency of compression algorithms. However, they probably also contain a vanishing amount of real information, which has to be analysed to identify bits with and without real information. The former should be conserved while the latter should be discarded to increase compression efficiency.

We define the bitwise *real information* content as the mutual information[20,37,40–42] of bits in adjacent grid points (Fig. 1 and Methods). A bit contains more real information the stronger the statistical dependence to the adjacent bits is. Bits without real information are identified when this dependence is insignificantly different from zero and we regard the remaining entropy in these bits as *false information*. The adjacent bit can be found in any of the dimensions of the data, e.g. in longitude, time or in the ensemble dimension. However, always the same bit position is analysed, e.g. the dependence of the first mantissa bit with other first mantissa bits in adjacent grid points.

In general, this analysis can be applied to any n-dimensional gridded data array when its adjacent elements are also adjacent in physical space. This includes structured and unstructured grids as long as a minimum level of smoothness is present. For example, data with little spatial correlation at the provided resolution will be largely identified as false information due to the independence of adjacent grid points (Fig. S3).

Based on the bitwise real information content, we suggest a new strategy for data compression of climate variables: First, we diagnose the real information for each bit. Afterwards, we round bits with no significant real information to zero, before applying lossless data compression. This allows us to minimise information loss but to maximise the efficiency of compression algorithms. Bits with no or only little real information (but high entropy) are discarded via binary round-to-nearest as defined in the IEEE-754 standard[13] (see Methods). This rounding mode is bias-free and therefore will ensure global conservation of quantities important in climate model data. Rounding is irreversible, but removes the incompressible false information and therefore increases compressibility.

Many exponent bits of the variables in CAMS have a high information content (Fig. 2), but information content decreases to zero within the first mantissa bits for most variables. Exceptions occur for variables like carbon dioxide (CO_2) with mixing ratios varying in a very limited range of 0.5-1.5 mg/kg (equivalent to about 330-990ppmv) globally. Due to the limited range, most exponent bits are unused and the majority of the real information is in mantissa bits 2 to 12.

The sum of real information across all bits is the total information per value, which is less than 7 bits for most variables. Importantly, the last few percent of total information is often distributed across many mantissa bits. This presents a trade-off where for a small tolerance in information loss many mantissa bits can be discarded, resulting in a large increase in compressibility (Fig. S4). Aiming for 99% preserved information is found to be a reasonable compromise.

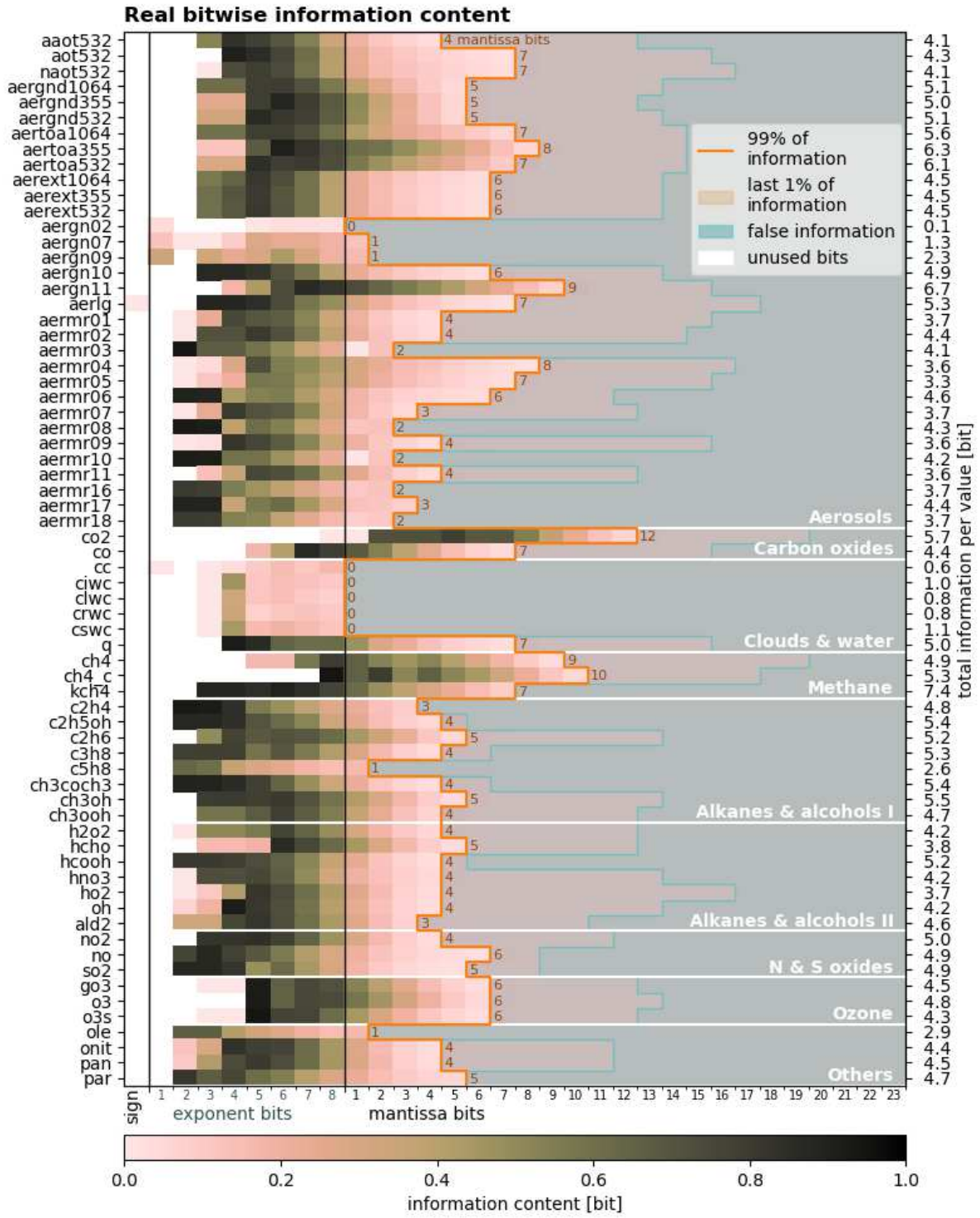

Fig. 2 | Bitwise real information content for all variables in CAMS. The real information is calculated in all three spatial dimensions, revealing false information and unused bits, using the 32-bit encoding of single-precision floats. The bits that should be retained to preserve 99% of real information are enclosed in orange; bits without any real information (see Methods) are shaded in grey-blue. The sum of these real information bits per variable is the total information per value. Variable abbreviations are explained in Table S1.

Compressing only the real information

Lossless compression algorithms can be applied efficiently to rounded floating-point arrays (the *round+lossless* method). Many general-purpose lossless compression algorithms are available[38,39,43–48], which are based on dictionaries and other statistical techniques to remove redundancies. Most algorithms operate on bitstreams and exploit the correlation of data in a single dimension only, we therefore describe this method as 1-dimensional (1D) compression.

The compression of water vapour at 100% preserved information (16 mantissa bits are retained) yields a compression factor of 7x relative to 64-bit floats (Fig. 3a). At 99% of preserved information (7 mantissa bits are retained) the compression factor increases to 39x, which corresponds to our recommended compromise. At this compression level a 15-fold efficiency increase is achieved compared to the current method. Effectively only 1.6bit of information is therefore stored per value.

Compressing all variables in CAMS and comparing error norms reveals the advantages of the 1D round+lossless method compared to the 24-bit linear quantization technique currently in use (Fig. 4). The maximum decimal errors are smaller for many variables due to the logarithmic distribution of floating-point numbers. Some variables are very compressible (>60x) due to many zeros in the data, which is automatically made use of in the lossless compression. Compression factors are between 3x and 60x for most variables, with a geometric mean of 6x when preserving 100% of information. Accepting a 1% information loss the geometric mean reaches 17x, which is the overall compression factor for the entire CAMS data set with this method when compared to data storage with 64 bits per value.

Furthermore, the 24-bit linear quantization could be replaced by a 16-bit logarithmic quantization, as the mean and absolute errors are comparable. The decimal errors are often even lower and naturally bound in a logarithmic quantization despite fewer available bits.

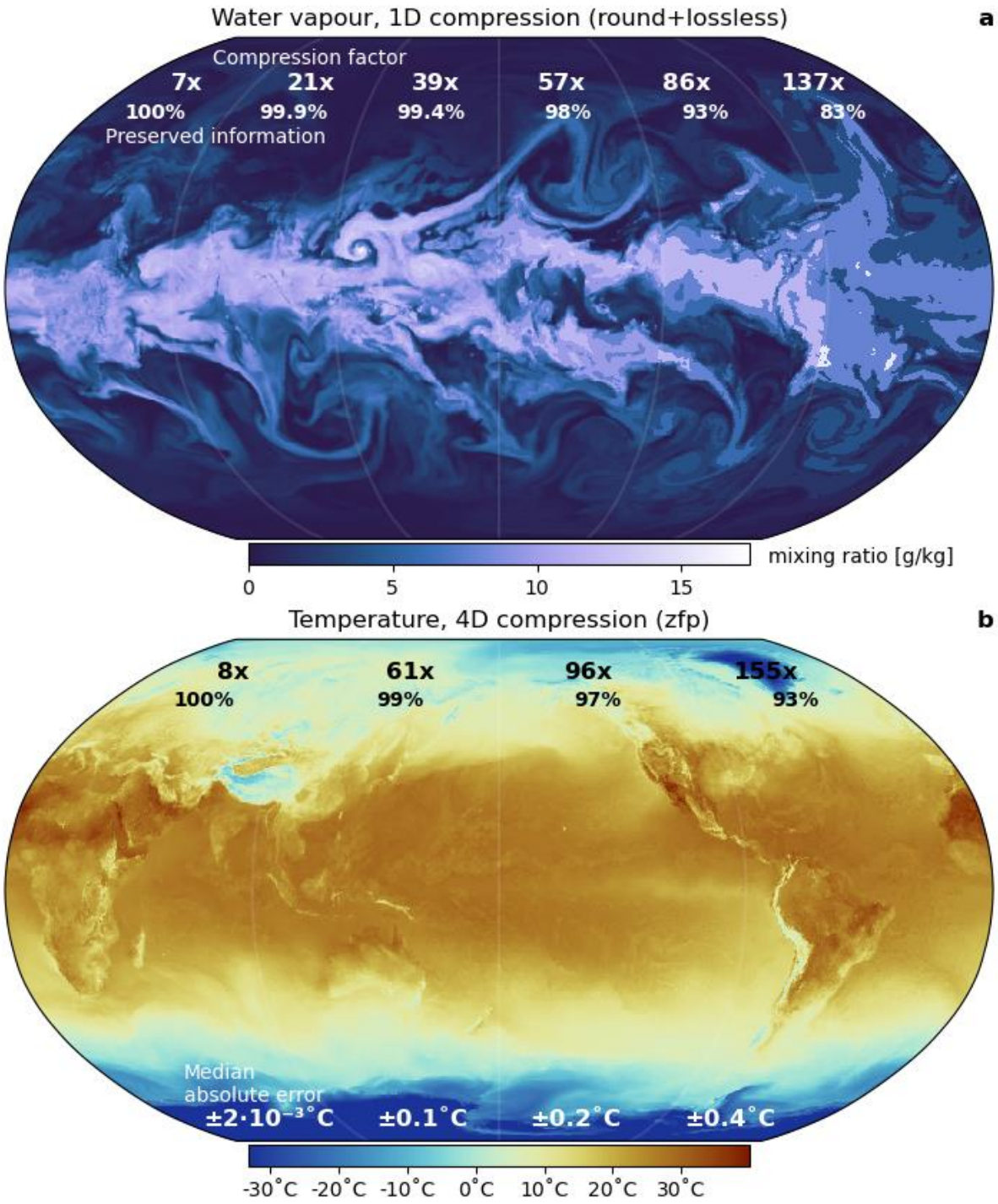

Fig. 3 | Compression at various levels of preserved information. **a** Water vapour (specific humidity) compressed in the longitudinal dimension. The vertical level shown is at about 2 km geopotential altitude, but compression factors include all vertical levels. **b** Surface temperature compressed in the four space-time dimensions at various levels of preserved information with compression algorithm Zfp. Compression factors are relative to 64-bit floats.

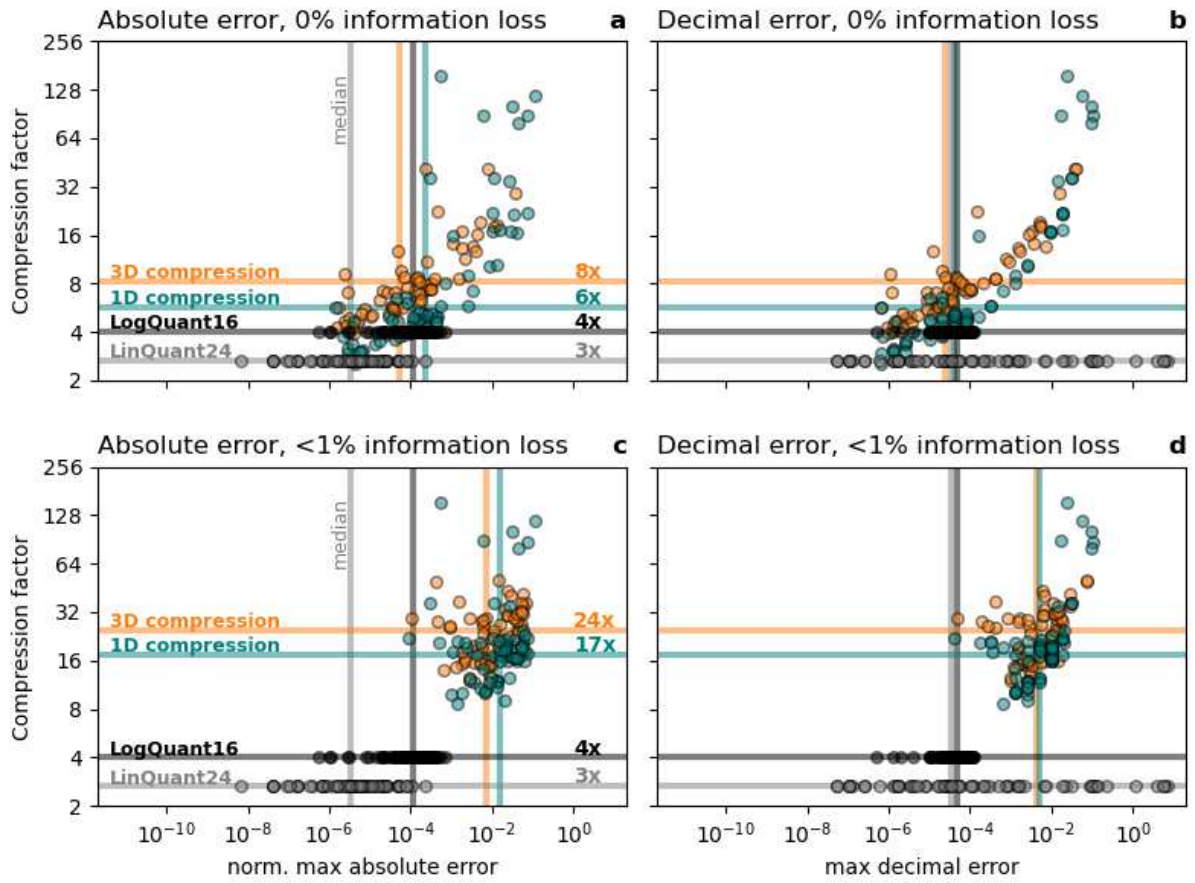

Fig. 4 | Compression factors versus compression errors. The maximum absolute and decimal error for 24-bit linear and 16-bit logarithmic quantisation (LinQuant24, LogQuant16) with 1-dimensional round+lossless and 3-dimensional Zfp compression. Every symbol represents for one variable the global maximum of the **a, c** normalised absolute error, **b, d** decimal error for **a, b** 100% preserved information, and **c, d** 99% preserved information. The geometric mean of compression factors over all variables is given as horizontal lines. The median of the errors across all variables is given as vertical lines.

Multi-dimensional data compression

General-purpose lossless compression algorithms operate on bitstreams and require multi-dimensional data to be unravelled into a single dimension. Modern compressors have been developed for multi-dimensional floating-point arrays[10,30,31], which compress in several dimensions simultaneously. We will compare the 1D lossless compression to Zfp, a modern compression algorithm for two to four dimensions[10]. Zfp divides a d-dimensional array into blocks of 4^d values (i.e. the edge length is 4), which allows to exploit the correlation of climate data in up to 4 dimensions. However, multi-dimensional compression imposes additional inflexibilities: Data is compressed and decompressed in larger chunks, which can increase the load on the data archive.

For 1D compression the compressibility varies with the dimension: Longitude (i.e. in the zonal direction) is more compressible, reaching 25x for temperature at 99% preserved information, than compressing in the vertical which yields only 14x (Fig. 5). This agrees with the predominantly zonal flow of the atmosphere as spatial correlation in the zonal direction is usually highest. Higher resolution in the respective dimensions increases the compressibility as also the correlation in adjacent grid points increases (Fig. S5).

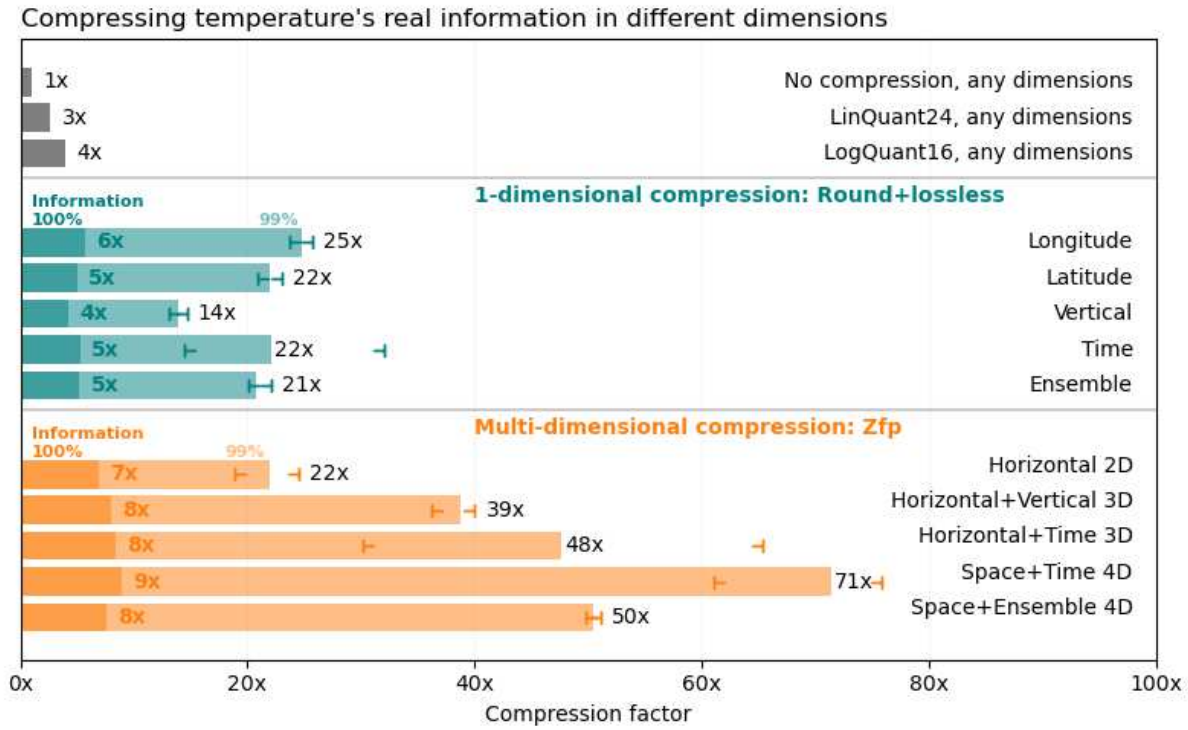

Fig. 5 | Multi-dimensional compression allows for higher compression factors. 1-dimensional compression of temperature reaches at most 25x when preserving 99% of real information with the round+lossless method, whereas 71x is reached with 4-dimensional (4D) space-time compression using Zfp compression. Preserving 100% of information considerably lowers the compression factors to 4-9x. Error brackets represent the min-max range of compression when applied to various data samples.

For multi-dimensional compression it is generally advantageous to include as many highly correlated dimensions as possible. In that sense, including the hourly-resolved forecast lead time instead of the vertical dimension in 3D compression yields higher compression factors. 4D space-time compression is the most efficient, reaching 60-75x at 99% preserved information. For temperature this is equivalent to a median absolute error of 0.1 ˚C (Fig. 3b).

Compressing the entire CAMS dataset in the three spatial dimensions with Zfp while preserving 99% of the information yields an overall compression factor of 24x (Fig. 4). Maximum absolute error and decimal errors are for most variables very similar to 1D

round+lossless (see Methods for a discussion why they are not identical), providing evidence that a multi-dimensional compression is preferable for higher compression factors.

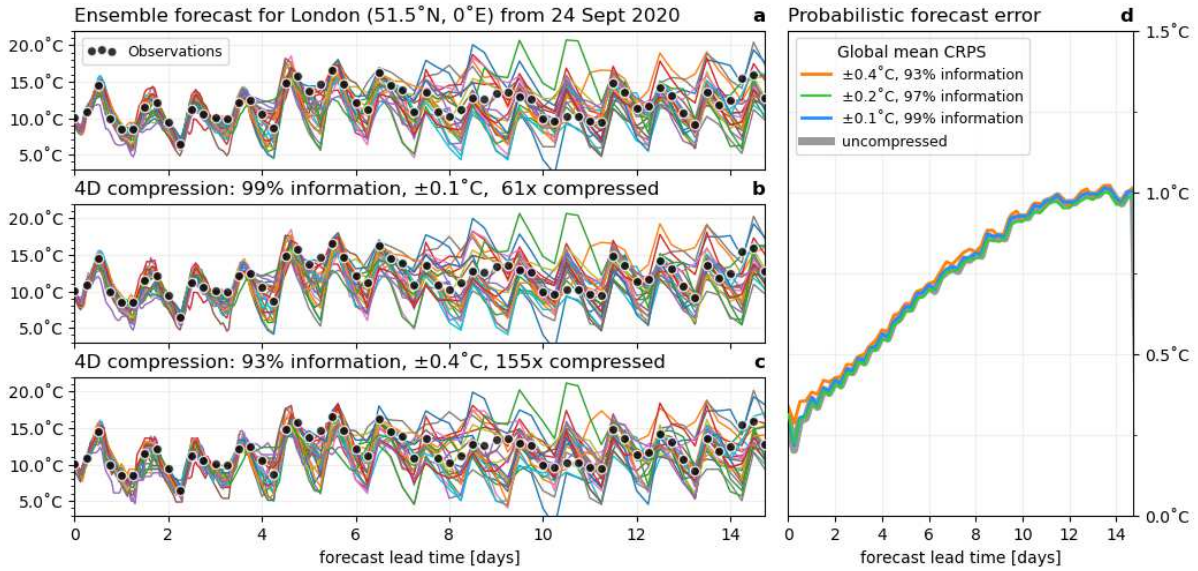

Fig. 6 | Verification of an ensemble forecast with the probabilistic forecast error CRPS (Continuous ranked probability score) with and without compression. a 25-member ensemble forecast of surface temperature in London, UK from 24 Sept 2020 up to 15 days ahead. **b** as **a** but the data was compressed in 4-dimensional (4D) space-time, preserving 99% of real information. **c** as **b** but only preserving 93% of real information. **d** Only for less than 93% of preserved information CRPS increases relative to the uncompressed reference.

Due to the limited meaning of error norms in the presence of uncertainties in the uncompressed reference data, the forecast error is assessed to quantify the quality of compressed atmospheric data. The continuous ranked probability score[49–51] (CRPS), a generalisation of the root-mean-square error for probabilistic forecasts, is evaluated for global surface temperature using observations every 6 hours as truth (Fig. 6). No significant increase of the CRPS forecast error (compared the uncompressed data) occurs for individual locations or globally at 99% and 97% preserved information. The usefulness for the end user of the global temperature forecast is therefore unaltered at these levels of preserved information in the compression. Contrarily, with an information loss larger than 5% the CRPS forecast error starts to increase, while compression factors beyond 150x are achieved.

A data compression Turing test

In numerical weather predictions, progress in the development of global weather forecasts is often assessed using a set of error metrics, summarized in so-called *score cards*. These scores cover important variables in various large scale regions, such as 2m-

temperature over Europe or horizontal wind speed at different vertical levels in the Southern Hemisphere. With a similar motivation as in Baker et al.[52], we suggest assessing the efficiency of climate data compression using similar scores, which have to be passed similar to a Turing test[33,53]. The compressed forecast data should be indistinguishable from the uncompressed data in all of these score tests, or at least indistinguishable from the current compression method while allowing higher compression factors.

Many score tests currently in use represent area-averages (such as Fig. 6d), which would also be passed with coarse-grained data — reducing the horizontal resolution from 10km to 20km, for example, yields a compression factor of 4x. It is therefore important to include resolution-sensitive score tests such as the maximum error in a region. While a compression method either passes or fails such a data compression Turing test, there is additional value in conducting such a test. Evaluating the failures will highlight problems and evaluating the passes may identify further compression potential.

A roadmap for climate data compression

While weather and climate forecast centres produce very large amounts of data, especially for future cloud and storm-resolving models, only the real information content in this data should be stored. We have here presented a methodology to identify real and false information in atmospheric, and more generally, climate data. This novel information-preserving compression relies on the removal of false information via rounding and can then be used in combination with any lossless compression algorithm. This is applied to CAMS data, and we show that a high compressibility can be achieved without increasing the forecast error. The entire data set is 17x smaller in the compressed form when compared to 64-bit values while preserving 99% of real information. This is about 6-times more efficient when compared to the current compression method.

No additional uncertainty measure has to be assumed for the distinction of real and false information presented here. The uncertainty of a variable represented in a data array is directly obtained from the distribution of the data itself. Most lossy compression techniques leave the choice of precision to the user, which may lead to subjective choices or the same precision for a group of variables. Instead, we suggest that the fraction of preserved information (e.g. 99%) may be altered by the user, which will implicitly determine the required precision for each variable (and possibly even subsets of it) individually.

Ideally, climate data compression should exploit correlation in as many dimensions as possible. The most important dimensions to compress along are longitude, latitude and time, which provide the highest compressibility. With Zfp, we achieved factors of 60-75x

for 4D space-time compression of temperature while preserving 99% of real information. Using the three spatial dimensions the entire set of variables in CAMS data can be compressed by 24x equivalently.

To be attractive for large data sets, a compression method should enable compression as well as decompression at reasonable speeds. ECMWF produces data at about 2GB/s, including CAMS which creates about 15 MB/s. Data on ECMWF's archive is compressed once, but downloaded on average at 120 MB/s by different users, such that both high compression and decompression speeds are important. The (de)compression speeds obtained here are all at least 100MB/s single-threaded (Fig. S4), but faster speeds are available in exchange for lower compression factors (see Methods). The real information is only analysed once and ultimately independent of the compressor choice.

Lossy compression inevitably introduces errors compared to the uncompressed data. Weather and climate forecast data, however, already contains uncertainties which are often larger than the compression error. Limiting the precision of surface temperature, for example, to 0.1˚C (as shown in Fig. 4b) is well below the average forecast error (Fig. 6d) and also more precise than the typical precision of 1˚C presented to end users of a weather forecast. Reducing the precision to the real information content does not just increase compressibility but also helps to directly communicate the uncertainty within the data set — an important, often neglected, information by itself.

Satisfying requirements on size, precision and speed simultaneously is an inevitable challenge of data compression. As the precision can be reduced without losing information, we revisit this trade-off and propose an information-preserving compression. While current archives likely use large capacities to store random bits, the analysis of the bitwise real information content is essential towards efficient climate data compression.

Methods

Data CAMS data is analysed for one time step on 01/12/2019 12:00 UTC, bilinearly regridded onto a regular 0.4˚x0.4˚ longitude-latitude grid using climate data operators (cdo) v1.9. All 137 vertical model levels are included. Furthermore, global fields of temperature from ECMWF's ensemble prediction system with 91 vertical levels are used from the first 25 members of a 50-member 15-day ensemble forecast starting on 24 Sept 2020 00:00 UTC. Bilinear regridding onto a regular 0.2˚x0.2˚ longitude-latitude grid, similar as for the CAMS data, is applied. All compression methods here include the conversion from Float64 to Float32.

Only longitude-latitude grids are considered in this paper. However, the methodology can be applied to other grids too. For example, ECMWF's octahedral grid collapses the two horizontal dimensions into a single horizontal dimension which circles on latitude bands around the globe starting at the South Pole till reaching the North Pole[54]. Fewer grid points reduce the size, but the correlation in latitudinal direction cannot be exploited.

Bitpattern entropy An n-bit number format has 2^n bitpatterns available to encode a real number. For most data arrays, not all bitpatterns are used at uniform probability. The bitpattern entropy is the Shannon information entropy H, in units of bits, calculated from the probability of each bitpattern p_i

$$H = -\sum_{i=1}^{2^n} p_i \log_2(p_i) \tag{1}$$

The bitpattern entropy is $H \leq n$ and maximised to n bits for a uniform distribution. The free entropy is the difference $n - H$.

Real information content The Shannon information entropy[20] H in units of bits takes for a bitstream $b = b_1 b_2 ... b_k ... b_l$, i.e. a sequence of bits of length l, the form

$$H(b) = -p_0 \log_2(p_0) - p_1 \log_2(p_1) \tag{2}$$

with p_0, p_1 being the probability of a bit b_k in b being 0 or 1. The entropy is maximised to 1 bit for equal probabilities $p_0 = p_1 = \frac{1}{2}$ in b. To derive the mutual information[40–42] of two bitstreams from adjacent grid points $r = r_1 r_2 ... r_k ... r_l$ and $s = s_1 s_2 ... s_k ... s_l$ are now considered. The mutual information is defined via the joint probability mass function p_{rs} which here takes the form of a 2x2 matrix

$$p_{rs} = \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix} \tag{3}$$

with p_{ij} being the probability that the bits are in the state $r_k = i$ and $s_k = j$ simultaneously and $p_{00} + p_{01} + p_{10} + p_{11} = 1$. The marginal probabilities follow as column or row-wise additions in p_{rs}, e.g. the probability that $r_k = 0$ is $p_{r=0} = p_{00} + p_{01}$. The mutual information $M(r, s)$ of the two bitstreams r, s is then

$$M(r,s) = \sum_{r=0}^{1} \sum_{s=0}^{1} p_{rs} \log_2 \left(\frac{p_{rs}}{p_{r=r} p_{s=s}} \right) \tag{4}$$

We now consider the two bitstreams r, s being the preceding and succeeding bits (for example in space or time) in a single bitstream b, i.e. $r = b_1 b_2 ... b_{l-1}$ and $s = b_2 b_3 ... b_l$. The (unconditional) entropy is then effectively $H = H(r) = H(s)$ as in Eq. (2) and for l being very large. The conditional entropies H_0, H_1 are conditioned on the state of the previous bit b_{j-1} being 0 or 1

$$\begin{aligned} H_0 &= -p_{00}\log_2(p_{00}) - p_{01}\log_2(p_{01}) \\ H_1 &= -p_{10}\log_2(p_{10}) - p_{11}\log_2(p_{11}) \end{aligned} \tag{5}$$

The conditional entropy is maximised to 1 bit for bitstreams where the probability of a bit being 0 or 1 does not depend on the state of the previous bit, which is here defined as *false information*. With the conditional and unconditional entropies and p_0, p_1 as in Eq. (2) the mutual information M of succeeding bits can be written as

$$I(b) = H - p_0 H_0 - p_1 H_1 \tag{6}$$

which is the *real information* content I. This definition is similar to Jeffres et al. (2017) but avoids an additional assumption of an uncertainty measure[22]. Eq. (5) defines the real information as the entropy minus the false information. For bitstreams with either $p_0 = 1$ or $p_1 = 1$, i.e. all bits are either 0 or 1, the entropies are zero $H = H_0 = H_1 = 0$ and we may refer to the bits in the bitstream as being *unused*. In the case where $H > p_0 H_0 + p_1 H_1$, the preceding bit is a predictor for the succeeding bit which means that the bitstream contains real information ($I > 0$).

The multi-dimensional real information content The real information content I_m for an m-dimensional array A is the sum of the real information along the m dimensions. Let b_j be a bitstream obtained by unravelling a given bitposition in A along its j-th dimension. Although the unconditional entropy H is unchanged along the m-dimensions, the conditional entropies H_0, H_1 change as the preceding and succeeding bit is found in another dimension, e.g. b_2 is obtained by re-ordering b_1. Normalization by $\frac{1}{m}$ is applied to I_m such that the maximum information is 1 bit in I_m^*

$$I_m^* = H - \frac{p_0}{m}\sum_{j=1}^{m} H_0(b_j) - \frac{p_1}{m}\sum_{j=1}^{m} H_1(b_j) \tag{7}$$

Due to the presence of periodic boundary conditions for longitude and given the large grid sizes generally, a succeeding bit might be found across the bounds of A. This simplifies the calculation as the bitstreams are obtained from permuting the dimensions of A and subsequent unravelling into a vector.

Preserved information We define the preserved information in a bitstream s when approximating r (e.g. after a lossy compression) via the symmetric normalised mutual information $M(r,s)$

$$R(r,s) = \frac{2M(r,s)}{H(r)+H(s)} \tag{8}$$

R is the redundancy of information of r in s. The preserved information is then the redundancy-weighted mutual information in r.

$$P(r,s) = R(r,s)M(r,s) \tag{9}$$

The information loss L is $1-P$ and represents the unpreserved information of r in s. In most cases we are interested in the preserved information of an array $X=(x_1,x_2,...,x_q,...,x_n)$ of bitstreams x when approximated by a previously compressed array $Y=(y_1,y_2,...,y_q,...,y_n)$. For an array A of floats with $n=32$ bit, for example, x_1 is the bitstream of all sign bits unravelled along a given dimension (e.g. longitudes) and x_{32} is the bitstreams of the last mantissa bits. The redundancy $R(X,Y)$ and the real information $I(X)$ is then calculated for each bit position q individually, and the preserved information is the redundancy-weighted mean of the real information in X

$$P(X,Y) = \frac{\sum_{q=1}^{n} R(x_q,y_q)I(x_q)}{\sum_{q=1}^{n} I(x_q)} \tag{10}$$

The quantity $\Sigma_{q=1}^{n} I(x_q)$ is the total information in X and therefore also in A. The redundancy is $R=1$ for bits that are unchanged during rounding and $R=0$ for bits that are round to zero. The preserved information with bitshave or halfshave[28,29] (i.e. replacing mantissa bits without real information with either 00...00 or 10...00, respectively) is therefore equivalent to truncating the bitwise real information for the (half)shaved bits. For round to nearest, however, the carry bit depends on the state of bits across several bit positions. To account for interdependency of bitpositions the mutual information has to be extended to include more bit positions in the joint probability p_{rs}, which will then be a m x 2 matrix. For computational simplicity, we truncate the real information as the rounding errors of round to nearest and halfshave are equivalent.

Significance of real information For an entirely independent and approximately equal occurrence of bits in a bitstream of length l, the probabilities p_0, p_1 of a bit being 0 or 1 approach $p_0 \approx p_1 \approx \frac{1}{2}$, but they are in practice not equal for $l < \infty$. Consequently, the entropy is smaller than 1, but only insignificantly. The probability p_1 of successes in the

binomial distribution (with parameter $p = \frac{1}{2}$) with l trials (using the normal approximation for large l) is

$$p_1 = \frac{1}{2} + \frac{z}{2\sqrt{l}} \tag{11}$$

where z is the $1 - \frac{1}{2}(1-c)$ quantile at confidence level c of the standard normal distribution. For $c = 0.99$, corresponding to a 99%-confidence level which is used as default here, $z = 2.58$ and for $l = 5.5 \cdot 10^7$ (the size of a 3D array from CAMS) a probability $\frac{1}{2} \le p \le p_1 = 0.5002$ is considered insignificantly different from equal occurrence $p_0 = p_1$. The associated free entropy H_f in units of bits follows as

$$H_f = 1 - p_1 \log_2(p_1) - (1-p_1)\log_2(1-p_1) \tag{12}$$

And we consider real information below H_f as insignificantly different from 0 and set the real information $I = 0$.

Linear and logarithmic quantization The n-bit linear quantization compression for each element a in an array A is

$$\tilde{a} = \mathrm{round}\left(2^{n-1}\frac{a - \min(A)}{\max(A) - \min(A)}\right) \tag{13}$$

with round a function that rounds to the nearest integer in $0, \ldots, 2^{n-1}$. Consequently, every compressed element $\tilde{a}$ can be stored with n bits. The n-bit logarithmic quantization compression for every element $a \ge 0$ in A is

$$\tilde{a} = \begin{cases} 0 & \text{if } a = 0, \\ \mathrm{round}\left(c + \Delta^{-1}\log(a)\right) + 1 & \text{else.} \end{cases} \tag{14}$$

which reserves the 0-bit pattern to encode 0. The logarithmic spacing is

$$\Delta = \frac{\log(\max(A)) - \log(\min^+(A))}{2^n - 2} \tag{15}$$

with a constant $c = 1/2 - \Delta^{-1}\log(\min^+(A)(\exp(\Delta)+1)/2)$ which is chosen to implement round-to-nearest in linear space instead of in logarithmic space, for which $c = -\Delta^{-1}\log(\min^+(A))$. The function $\min^+(A)$ is the minimum of all positive elements in A.

Rounding With round-to-nearest a full-precision number is replaced by the nearest representable float with fewer mantissa bits by rounding the trailing bits to zero. Representing π as the 32-bit float `f`, for example, can then be round to 6 mantissa bits (orange) as

```
      f = 0 10000000 10010010000111111011011 = 3.1415927
round(f) = 0 10000000 10010100000000000000000 = 3.15625
```
(16)

where the 6th mantissa bit flips due to the carry bit, i.e. `f` is round up, `f < round(f)`. Alternative rounding modes have been proposed for data compression[28,29], but many suffer from some bias or introduce larger rounding errors.

Error norms The normalised absolute error $\mathcal{E}^*_{\mathrm{abs}}$ of an element $\tilde{a}$ from a compressed array $\tilde{A}$ relative to the respective element a from full-precision array A is

$$\mathcal{E}^*_{\mathrm{abs}} = \frac{|\tilde{a} - a|}{\mathrm{mean}(|A|)} \tag{17}$$

where $|A|$ denotes the element-wise absolute value of A. The normalisation with $\mathrm{mean}(|A|)$ is therefore the same for all element pairs across A and $\tilde{A}$, which distinguishes it from a relative error. It used to make the absolute errors between variables with different value ranges comparable. The decimal error $\mathcal{E}_{\mathrm{dec}}$ is[55]

$$\mathcal{E}_{\mathrm{dec}} = |\log_{10}\left(\frac{\tilde{a}}{a}\right)| \tag{18}$$

Special cases are $\mathcal{E}_{\mathrm{dec}} = \infty$ when a or $\tilde{a}$ is 0 or the signs do not match $\mathrm{sign}(a) \neq \mathrm{sign}(\tilde{a})$, unless $\tilde{a} = a = 0$ in which case $\mathcal{E}_{\mathrm{dec}} = 0$. The decimal error is used to better highlight when lossy data compression changes the sign (with $\mathrm{sign}(0) = 0$) of a value. Bounding the absolute or relative error does not enforce that. The maximum normalised absolute and the decimal error are then the maximum of all $\mathcal{E}^*_{\mathrm{abs}}$ and $\mathcal{E}_{\mathrm{dec}}$, respectively, computed for all element pairs across A and $\tilde{A}$.

Lossless compression Zstandard is a modern compression algorithm that combines many techniques to form a single compressor with tunable 22 compression levels that allow large trade-offs between compression speed and factors[46,48]. Here, we use compression level 10, as it presents a reasonable compromise between speed and size. Zstandard outperforms other tested algorithms (deflate, LZ4, LZ4HC and Blosc) in our applications and is also found to be among the best in the lzbench compression benchmark[46] and other studies have focused on comparisons[43]. Lossless compressors

are often combined with reversible transformations that preprocess the data. The so-called bitshuffle[43] transposes an array on the bit-level, such that bitpositions (e.g. the sign bit) of floating-point numbers are stored next to each other in memory. Another example is the bitwise XOR-operation[56] with the preceding floating-point value, which sets subsequent bits that are identical to 0. Neither bitshuffle nor XOR significantly increased the compression factors in our applications.

Matching preserved bits to Zfp's precision The Zfp compression algorithm divides a d-dimensional array into blocks of size 4^d to exploit correlation in every dimension of the data. Within each block a transformation of the data is applied with specified absolute error tolerance or precision, which bounds a local relative error. Due to the rather logarithmic distribution of CAMS data (Fig. S1), a log-preprocessing of the data is applied to prevent sign changes (including a flushing to zero) within the compression. The error introduced by Zfp is approximately normally distributed (Fig. S7) and therefore usually yields higher maximum errors compared to round-to-nearest in float arithmetic. In order to find an equivalent error level between the two methods we therefore chose the precision level of Zfp to yield median absolute and decimal errors that are at least as small as those from rounding.

Compressor performances Although different compressors and their performance are not within the central focus of this study, we analyse the compression and decompression speeds as a sanity check (Fig. S6). In order to find a data compression method that can be used operationally, a certain minimum data throughput should be achieved. The 24-bit linear quantisation method currently reaches compression speeds of almost 800 MB/s single-threaded on an Intel i7 (Kaby Lake) CPU in our application, excluding writing to disk. For the logarithmic quantisation, this would decrease to about 200 MB/s due to the additional evaluation of a logarithm for every value. For Zstandard the user can choose between 22 compression levels, providing a trade-off between compression speed (highest for level 1) and compression factor (highest for level 22). Compression speed reduces from about 700 MB/s at compression level 1 to 2 MB/s at level 22, such that for high compression factors about a thousand cores would be required in parallel to compress in real time the 2GB/s data production at ECMWF. For Zstandard at compression level 10 speeds of at least 100MB/s are achieved, but at the cost of about 50% larger file sizes. We use compression level 10 throughout this study as a compromise. The decompression speed is independent of the level. The additional performance cost of binary rounding is with 2 GB/s negligible. Zfp reaches compression speeds of about 200 MB/s (single-threaded, including the log-preprocessing) in our application, enough to compress ECMWF's data production in real time with a small number of processors in parallel.

Acknowledgements

MK gratefully acknowledges funding from the Copernicus Programme within the ECMWF summer of weather code 2020 and 2021 and from the Natural Environmental Research Council under grant number NE/L002612/1. MK and TNP gratefully acknowledge funding from the European Research Council under grant number 74112. MR and JJD acknowledge funding from ECMWF and the Copernicus Programme. PDD gratefully acknowledges funding from the Royal Society for his University Research Fellowship as well as funding from the ESIWACE2 project (EU Horizon 2020 under grant number 823988).

Author contributions

Conceptualization: MK, MR, JJD. Data curation: MR, JJD, MK. Formal Analysis: MK. Methodology: MK. Visualization: MK. Writing – original draft: MK. Writing – review & editing: MK, PDD, MR, JJD, TNP

Conflict of interest

The authors declare no conflict of interest.

Data and software availability

Software that was developed for this study is available in the published Julia packages BitInformation.jl (v0.2), LinLogQuantization.jl (v0.2) and ZfpCompression.jl (v0.2). All scripts to reproduce the analysis are available in https://github.com/esowc/Elefridge.jl (will be converted to a DOI upon acceptance). Relevant software and a subset of the data will also be made available on Code Ocean for reproducibility. The entire CAMS data set is freely available to download from ECMWF or on the Copernicus Atmosphere Data Store at https://atmosphere.copernicus.eu/data. Full precision data that is was not subject to lossy compression before, which was used here, is available in

Copernicus Atmosphere Monitoring Service, 2021. CAMS forecast experiment using GRIB IEEE data encoding. doi: 10.21957/56gh-9y86
European Centre for Medium Range Weather Forecasts, 2021, Ensemble temperature forecast experiment using GRIB IEEE data encoding. doi: 10.21957/phgf-bv34

References

1. Bauer, P., Thorpe, A. & Brunet, G. The quiet revolution of numerical weather prediction.

Nature **525**, 47–55 (2015).
2. Bauer, P. *et al.* The ECMWF Scalability Programme: Progress and Plans. (2020) doi:10.21957/gdit22ulm.
3. Voosen, P. Europe is building a 'digital twin' of Earth to revolutionize climate forecasts. *Sci. AAAS* (2020) doi:10.1126/science.abf0687.
4. Schär, C. *et al.* Kilometer-Scale Climate Models: Prospects and Challenges. *Bull. Am. Meteorol. Soc.* **101**, E567–E587 (2020).
5. Bauer, P., Stevens, B. & Hazeleger, W. A digital twin of Earth for the green transition. *Nat. Clim. Change* **11**, 80–83 (2021).
6. Stevens, B. *et al.* DYAMOND: the DYnamics of the Atmospheric general circulation Modeled On Non-hydrostatic Domains. *Prog. Earth Planet. Sci.* **6**, 61 (2019).
7. Molteni, F., Buizza, R., Palmer, T. N. & Petroliagis, T. The ECMWF Ensemble Prediction System: Methodology and validation. *Q. J. R. Meteorol. Soc.* **122**, 73–119 (1996).
8. Palmer, T. The ECMWF ensemble prediction system: Looking back (more than) 25 years and projecting forward 25 years. *Q. J. R. Meteorol. Soc.* **145**, 12–24 (2019).
9. Ballester-Ripoll, R., Lindstrom, P. & Pajarola, R. TTHRESH: Tensor Compression for Multidimensional Visual Data. *IEEE Trans. Vis. Comput. Graph.* **26**, 2891–2903 (2020).
10. Lindstrom, P. Fixed-Rate Compressed Floating-Point Arrays. *IEEE Trans. Vis. Comput. Graph.* **20**, 2674–2683 (2014).
11. von Larcher, T. & Klein, R. On identification of self-similar characteristics using the Tensor Train decomposition method with application to channel turbulence flow. *Theor. Comput. Fluid Dyn.* **33**, 141–159 (2019).
12. Zhao, K. *et al.* Significantly Improving Lossy Compression for HPC Datasets with Second-Order Prediction and Parameter Optimization. in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing* 89–100 (Association for Computing Machinery, 2020). doi:10.1145/3369583.3392688.
13. IEEE Standard for Binary Floating-Point Arithmetic. *ANSIIEEE Std 754-1985* 1–20 (1985) doi:10.1109/IEEESTD.1985.82928.
14. Váňa, F. *et al.* Single Precision in Weather Forecasting Models: An Evaluation with the IFS. *Mon. Weather Rev.* **145**, 495–502 (2017).
15. Tintó Prims, O. *et al.* How to use mixed precision in ocean models: exploring a potential reduction of numerical precision in NEMO 4.0 and ROMS 3.6. *Geosci. Model Dev.* **12**, 3135–3148 (2019).
16. Hatfield, S., Chantry, M., Düben, P. & Palmer, T. Accelerating High-Resolution Weather Models with Deep-Learning Hardware. in *Proceedings of the Platform for Advanced Scientific Computing Conference* 1–11 (Association for Computing Machinery, 2019). doi:10.1145/3324989.3325711.
17. Klöwer, M., Düben, P. D. & Palmer, T. N. Number formats, error mitigation and scope for 16-bit arithmetics in weather and climate modelling analysed with a shallow water model. *J. Adv. Model. Earth Syst.* **n/a**, e2020MS002246 (2020).
18. Dawson, A. Reliable low precision simulations in land surface models. 10.
19. Ackmann, J., Düben, P. D., Palmer, T. N. & Smolarkiewicz, P. K. Mixed-precision for Linear Solvers in Global Geophysical Flows. *ArXiv210316120 Phys.* (2021).
20. Shannon, C. E. A mathematical theory of communication. *Bell Syst. Tech. J.* **27**, 623–656

(1948).
21. Kleeman, R. Information Theory and Dynamical System Predictability. *Entropy* **13**, 612–649 (2011).
22. Jeffress, S., Düben, P. & Palmer, T. Bitwise efficiency in chaotic models. *Proc. R. Soc. Math. Phys. Eng. Sci.* **473**, 20170144 (2017).
23. Palmer, T. Modelling: Build imprecise supercomputers. *Nat. News* **526**, 32 (2015).
24. Palmer, T. Climate forecasting: Build high-resolution global climate models. *Nat. News* **515**, 338 (2014).
25. Silver, J. D. & Zender, C. S. The compression–error trade-off for large gridded data sets. *Geosci. Model Dev.* **10**, 413–423 (2017).
26. Kuhn, M., Kunkel, J. M. & Ludwig, T. Data compression for climate data. *Supercomput. Front. Innov.* **3**, 75–94 (2016).
27. Hübbe, N., Wegener, A., Kunkel, J. M., Ling, Y. & Ludwig, T. Evaluating Lossy Compression on Climate Data. in *Supercomputing* (eds. Kunkel, J. M., Ludwig, T. & Meuer, H. W.) 343–356 (Springer, 2013). doi:10.1007/978-3-642-38750-0_26.
28. Zender, C. S. Bit Grooming: statistically accurate precision-preserving quantization with compression, evaluated in the netCDF Operators (NCO, v4.4.8+). *Geosci. Model Dev.* **9**, 3199–3211 (2016).
29. Kouznetsov, R. A note on precision-preserving compression of scientific data. *Geosci. Model Dev. Discuss.* 1–9 (2020) doi:https://doi.org/10.5194/gmd-2020-239.
30. Di, S. & Cappello, F. Fast Error-Bounded Lossy HPC Data Compression with SZ. in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)* 730–739 (2016). doi:10.1109/IPDPS.2016.11.
31. Lindstrom, P. & Isenburg, M. Fast and Efficient Compression of Floating-Point Data. *IEEE Trans. Vis. Comput. Graph.* **12**, 1245–1250 (2006).
32. Fan, Q., Lilja, D. J. & Sapatnekar, S. S. Using DCT-based Approximate Communication to Improve MPI Performance in Parallel Clusters. in *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)* 1–10 (IEEE, 2019). doi:10.1109/IPCCC47392.2019.8958720.
33. Baker, A. H. *et al.* Evaluating lossy data compression on climate simulation data within a large ensemble. *Geosci. Model Dev.* **9**, 4381–4403 (2016).
34. Woodring, J., Mniszewski, S., Brislawn, C., DeMarle, D. & Ahrens, J. Revisiting wavelet compression for large-scale climate data using JPEG 2000 and ensuring data precision. in *2011 IEEE Symposium on Large Data Analysis and Visualization* 31–38 (2011). doi:10.1109/LDAV.2011.6092314.
35. Inness, A. *et al.* The CAMS reanalysis of atmospheric composition. *Atmospheric Chem. Phys.* **19**, 3515–3556 (2019).
36. WMO. Guide to the WMO Table Driven Code Form Used for the Representation and Exchange of Regularly Spaced Data In Binary Form: FM 92 GRIB Edition 2. (2003).
37. MacKay, D. *Information Theory, Inference and Learning Algorithms*. (Cambridge University Press, 2003).
38. Ziv, J. & Lempel, A. A universal algorithm for sequential data compression. *IEEE Trans. Inf. Theory* **23**, 337–343 (1977).
39. Huffman, D. A. A Method for the Construction of Minimum-Redundancy Codes. *Proc. IRE*

40, 1098–1101 (1952).
40. Schreiber, T. Measuring Information Transfer. *Phys. Rev. Lett.* **85**, 461–464 (2000).
41. Kraskov, A., Stögbauer, H. & Grassberger, P. Estimating mutual information. *Phys. Rev. E* **69**, 066138 (2004).
42. Pothapakula, P. K., Primo, C. & Ahrens, B. Quantification of Information Exchange in Idealized and Climate System Applications. *Entropy* **21**, 1094 (2019).
43. Delaunay, X., Courtois, A. & Gouillon, F. Evaluation of lossless and lossy algorithms for the compression of scientific datasets in netCDF-4 or HDF5 files. *Geosci. Model Dev.* **12**, 4099–4113 (2019).
44. Ziv, J. & Lempel, A. Compression of individual sequences via variable-rate coding. *IEEE Trans. Inf. Theory* **24**, 530–536 (1978).
45. Deutsch, P. DEFLATE Compressed Data Format Specification version 1.3. (1996).
46. Skibinski, P. *inikep/lzbench*. (2020).
47. Alted, F. Why Modern CPUs Are Starving and What Can Be Done about It. *Comput. Sci. Eng.* **12**, 68–71 (2010).
48. Collet, Y. & Kucherawy, M. Zstandard Compression and the application/zstd Media Type. https://tools.ietf.org/html/rfc8478.
49. Matheson, J. E. & Winkler, R. L. Scoring Rules for Continuous Probability Distributions. *Manag. Sci.* **22**, 1087–1096 (1976).
50. Hersbach, H. Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems. *Weather Forecast.* **15**, 559–570 (2000).
51. Zamo, M. & Naveau, P. Estimation of the Continuous Ranked Probability Score with Limited Information and Applications to Ensemble Weather Forecasts. *Math. Geosci.* **50**, 209–234 (2018).
52. Baker, A. H., Hammerling, D. M. & Turton, T. L. Evaluating image quality measures to assess the impact of lossy data compression applied to climate simulation data. *Comput. Graph. Forum* **38**, 517–528 (2019).
53. TURING, A. M. I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind* **LIX**, 433–460 (1950).
54. Malardel, S. *et al.* A new grid for the IFS. *ECMWF Newsl.* (2016).
55. Klöwer, M., Düben, P. D. & Palmer, T. N. Posits as an alternative to floats for weather and climate models. in *Proceedings of the Conference for Next Generation Arithmetic 2019 on - CoNGA'19* 1–8 (ACM Press, 2019). doi:10.1145/3316279.3316281.
56. Pelkonen, T. *et al.* Gorilla: a fast, scalable, in-memory time series database. *Proc. VLDB Endow.* **8**, 1816–1827 (2015).

Supplementary materials

Name	Code	Unit	Name	Code	Unit
Aerosols			**Carbon oxides**		
Aerosol optical thickness 532nm	aot532	1	Carbon dioxide	co2	kg/kg
Anthropogenic aot532	aaot532	1	Carbon monoxide	co	kg/kg
Natural aot532	naot532	1	**Clouds and water**		
Backscatter from ground at 1064nm	aergnd1064	$m^{-1}sr^{-1}$	Fraction of cloud cover	cc	1
Backscatter from ground at 355nm	aergnd355	$m^{-1}sr^{-1}$	Cloud ice water content	ciwc	kg/kg
Backscatter from ground at 532nm	aergnd532	$m^{-1}sr^{-1}$	Cloud liquid water content	clwc	kg/kg
Backscatter from top of atm at 1064nm	aertoa1064	$m^{-1}sr^{-1}$	Specific rain water content	crwc	kg/kg
Backscatter from top of atm at 532nm	aertoa355	$m^{-1}sr^{-1}$	Specific snow water content	cswc	kg/kg
Backscatter from top of atm at 532nm	aertoa532	$m^{-1}sr^{-1}$	Specific humidity	q	kg/kg
Aerosol extinction coefficient at 1064nm	aerext1064	m^{-1}	**Methane**		
Aerosol extinction coefficient at 355nm	aerext355	m^{-1}	Methane	ch4	kg/kg
Aerosol extinction coefficient at 532nm	aerext532	m^{-1}	Methane (chemistry)	ch4_c	kg/kg
Aerosol type 2 source/gain accumulated	aergn02	kg/m^2	Methane loss rate	kch4	s^{-1}
Aerosol type 7 source/gain accumulated	aergn07	kg/m^2	**Alkanes or alcohols**		
Aerosol type 9 source/gain accumulated	aergn09	kg/m^2	Ethene	c2h4	kg/kg
Aerosol type 10 source/gain accumulated	aergn10	kg/m^2	Ethanol	c2h5oh	kg/kg
Aerosol type 11 source/gain accumulated	aergn11	kg/m^2	Ethane	c2h6	kg/kg
Aerosol large mode mixing ratio	aerlg	kg/kg	Propane	c3h8	kg/kg
Sea salt (0.03-0.5μm)	aermr01	kg/kg	Isoprene	c5h8	kg/kg
Sea salt (0.5-5μm)	aermr02	kg/kg	Acetone	ch3coch3	kg/kg
Sea salt (5-20μm)	aermr03	kg/kg	Methanol	ch3oh	kg/kg
Dust aerosol (0.03-0.55μm)	aermr04	kg/kg	Methyl peroxide	ch3ooh	kg/kg
Dust aerosol (0.55-0.9μm)	aermr05	kg/kg	Hydrogen peroxide	h2o2	kg/kg
Dust aerosol (0.9-20μm)	aermr06	kg/kg	Formaldehyde	hcho	kg/kg
Hydrophilic organic matter	aermr07	kg/kg	Formic acid	hcooh	kg/kg
Hydrophobic organic matter	aermr08	kg/kg	Nitric acid	hno3	kg/kg
Hydrophilic black carbon	aermr09	kg/kg	Hydroperoxy radical	ho2	kg/kg
Hydrophobic black carbon	aermr10	kg/kg	Hydroxyl radical	oh	kg/kg
Sulphate aerosol	aermr11	kg/kg	Aldehyde	ald2	kg/kg
Nitrate fine mode	aermr16	kg/kg	**Nitrogen and sulphur oxides**		
Nitrate coarse mode	aermr17	kg/kg	Nitrogen dioxide	no2	kg/kg
Ammonium aerosol	aermr18	kg/kg	Nitrogen monoxide	no	kg/kg
Others			Sulphur dioxide	so2	kg/kg
Olefins	ole	kg/kg	**Ozone**		

Organic nitrates	onit	kg/kg	Ozone mass mixing ratio 2	go3	kg/kg
Peroxyacetyl nitrate	pan	kg/kg	Ozone mass mixing ratio 1	o3	kg/kg
Paraffins	par	kg/kg	Stratospheric ozone	o3s	kg/kg

Table S1: List of atmospheric variables in CAMS with name, variable code and unit.

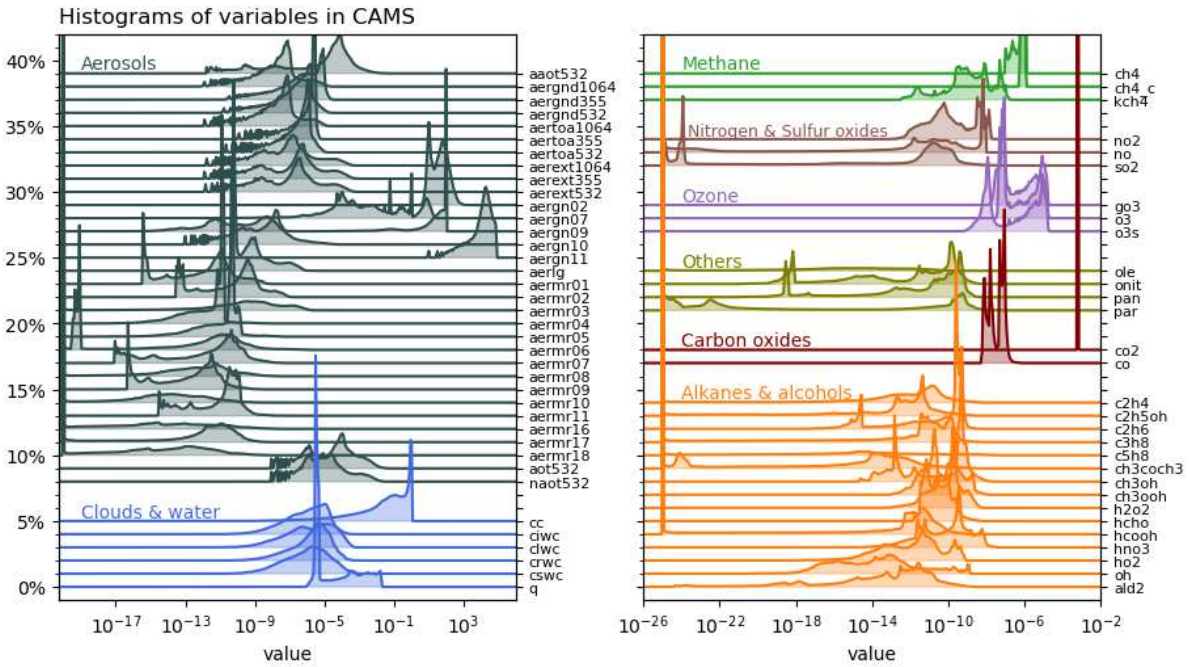

Fig. S1 | Statistical distributions of all variables in CAMS. Histograms use a logarithmic binning and are staggered vertically for clarity. The variable abbreviations are explained in Table S1.

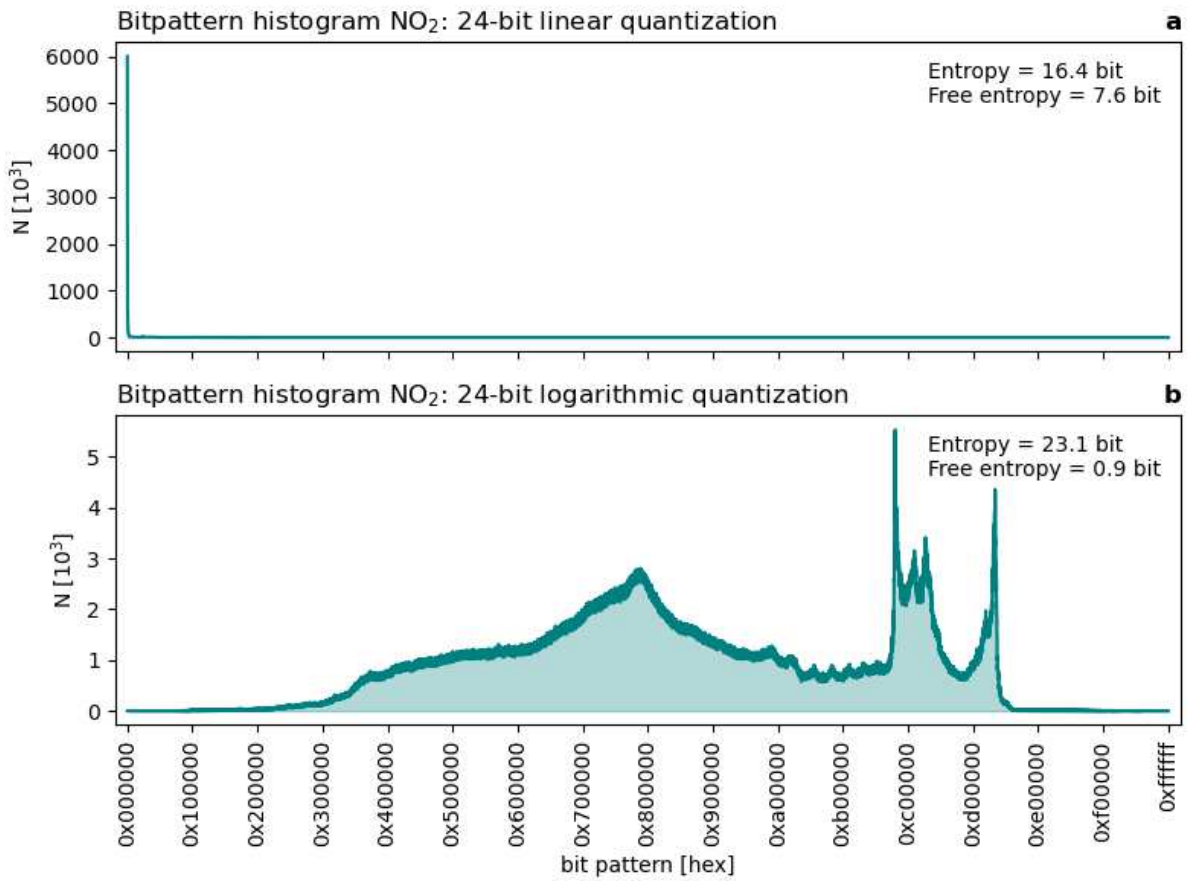

Fig. S2 | Bitpattern histogram for linear and logarithmic quantization. a linear 24-bit quantization and **b** 24-bit logarithmic quantization of nitrogen dioxide NO_2 mixing ratio [kg/kg]. All grid points and all vertical levels are used, consisting of $5.6 \cdot 10^7$ values with a range of $2 \cdot 10^{-14}$ to $2 \cdot 10^{-7}$ kg/kg. Bitpatterns are denoted in 24-bit hexadecimal. The free entropy is the difference between the available 24 bit and the bitpattern entropy (see methods) and quantifies the number of effectively unused bits.

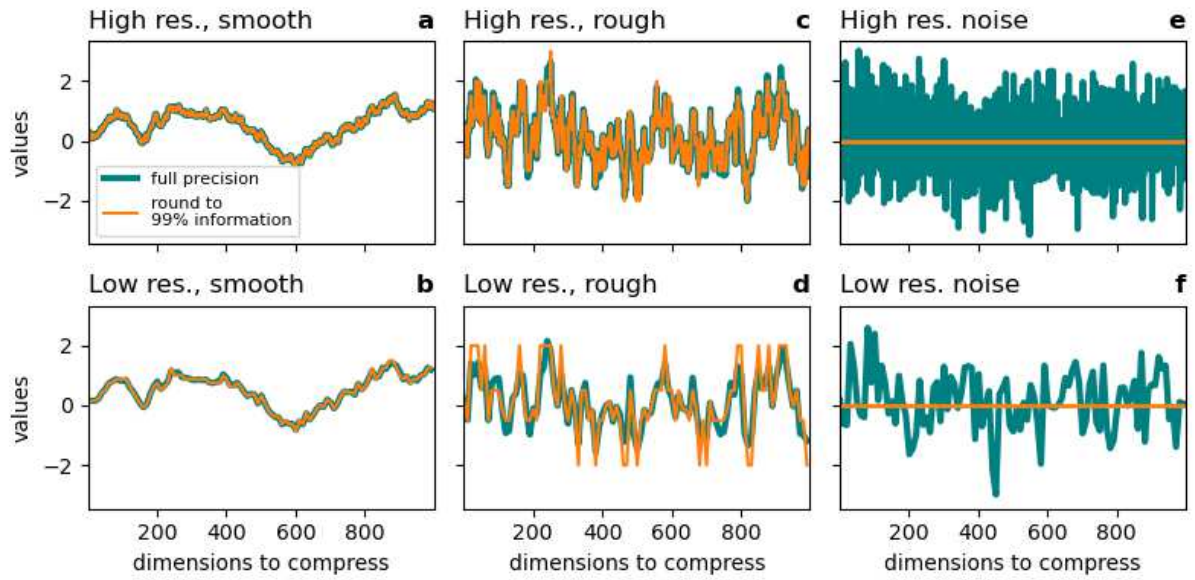

Fig. S3 | Resolution and smoothness dependence of the information-preserving compression. a,b Highly auto-correlated data (1st order auto-regressive process with correlation r=0.999) will have many mantissa bits preserved, at high and low resolution. **c,d** Many mantissa bits in data with less auto-correlation (r=0.95) will be independent at low resolution and therefore round to zero. **e,f** All bits in random data (r=0) drawn from a standard normal distribution are fully independent so that removing the false information rounds this data to zero. Low resolution data (**b,d,f**) is obtained from high resolution (**a,c,e**) by subsampling every 10th data point.

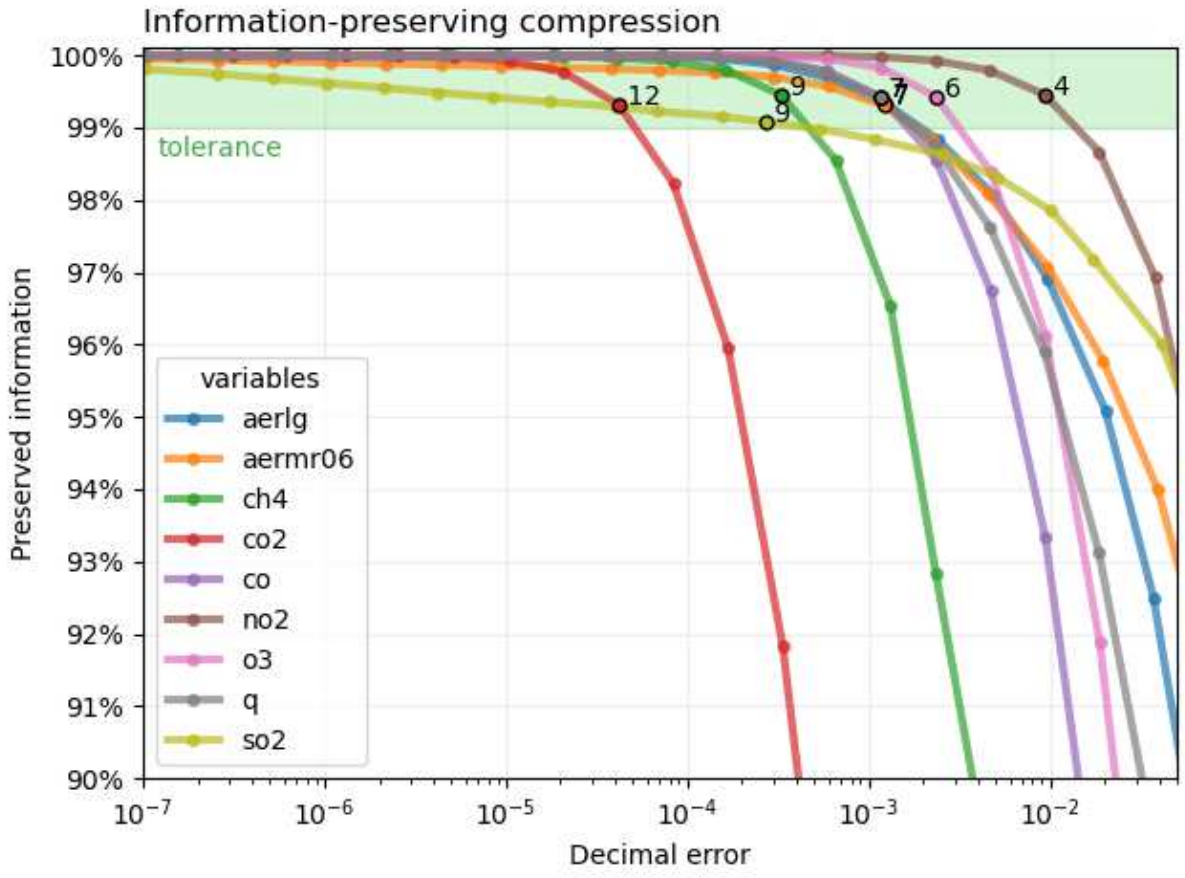

Fig. S4 | Information-preserving compression aims to discard as many bits as possible while limiting information loss. The last 1% of information tends to be distributed across many mantissa bits such that a trade-off arises where a large increase in compressibility is achieved for a small tolerance in information loss. For several variables in CAMS the preserved information is presented as a function of the decimal error, which itself increases linearly for every additional bit (small circles) that is discarded due to rounding. Denoted circles present the number of mantissa bits that have to be retained during compression to preserve at least 99% of information.

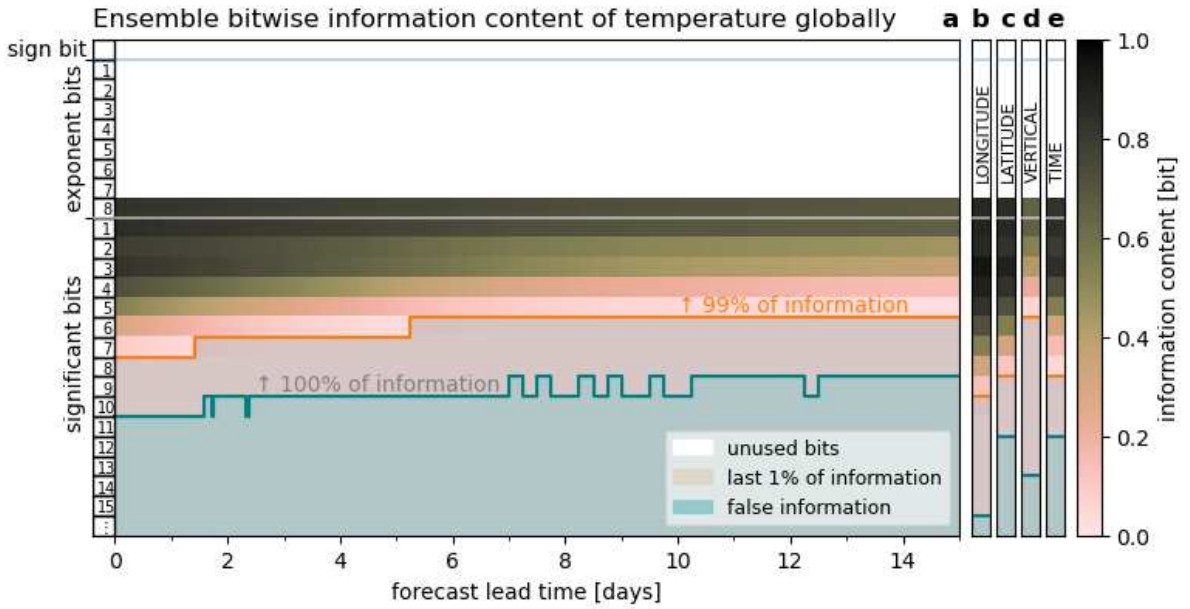

Fig. S5 | Bitwise real information content for temperature in various dimensions. a ensemble, **b** longitude, **c** latitude, **d** vertical and **e** forecast lead time. The ensemble information effectively encodes the ensemble mean, which is less information than in most other dimensions. Longitude, latitude and forecast lead time have the highest total information which should be preserved in compression. The ensemble information decreases over time as the ensemble spread increases.

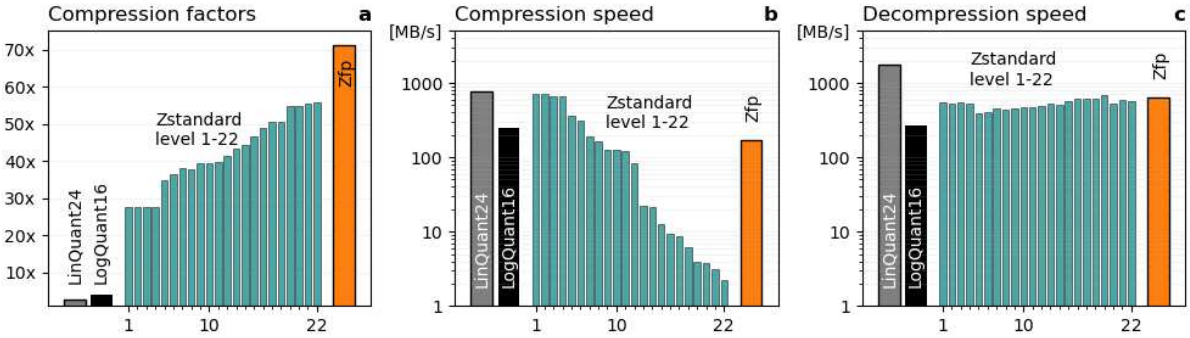

Fig. S6 | Compressor performances. Compressing water vapour (specific humidity, variable code q) (3 mantissa bits retained, as in Fig. 3) with 24-bit linear quantisation (LinQuant24), 16-bit logarithmic quantisation (LogQuant16), round+lossless (Zstandard, compression level 1-22) and Zfp (precision-mode, including log-preprocessing): **a** Compression factors, **b** compression speed, **c** decompression speed. Timings are single-threaded on an Intel Core™ i7 (Kaby Lake) and do not include the writing to disk.

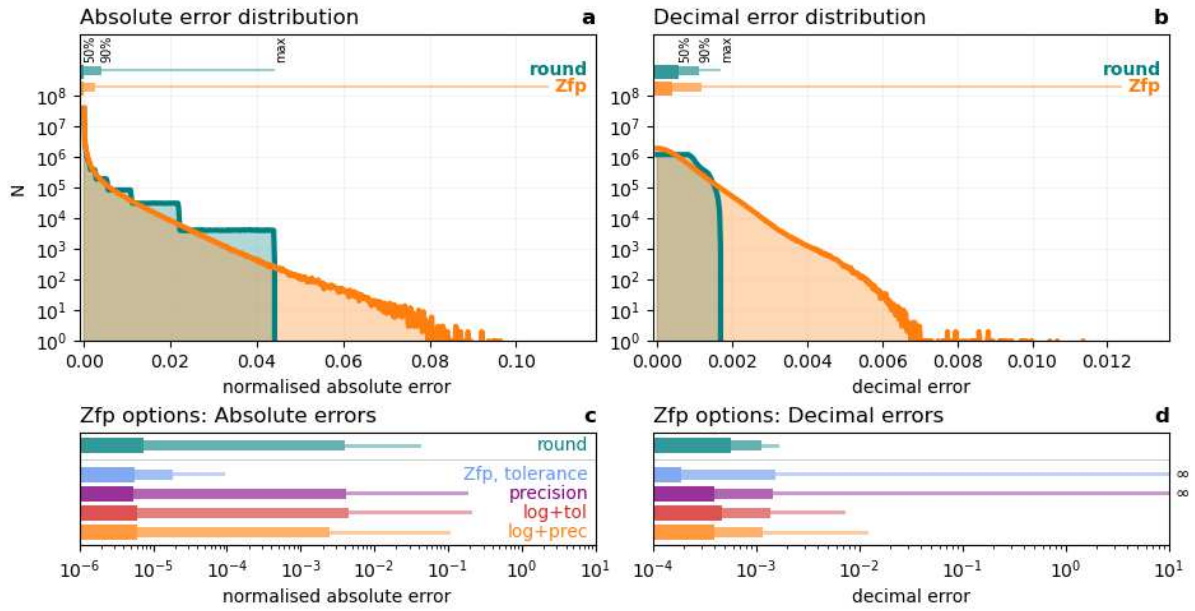

Fig. S7 | Error distribution of binary rounding compared to Zfp compression. IEEE round-to-nearest and Zfp compression of water vapour (specific humidity) in the three spatial dimensions. **a, c** normalised absolute errors **b, d** decimal errors. 7 mantissa bits are retained for rounding corresponding to 99% preserved information. The precision parameter of Zfp is chosen to yield median errors that are at least as small as those obtained by rounding. **c, d** Zfp via specifying tolerance (tol) or precision (prec) with and without log-preprocessing. Some decimal errors reached infinity in **d** due to sign changes.