The Journal of Open Source Software

xbitinfo: Compressing geospatial data based on information theory

Hauke Schulz [1,2¶], **Milan Klöwer** [3], and **Aaron Spring** [*4]

1 University of Washington, Seattle, USA **2** eScience Institute, University of Washington, Seattle, USA **3** University of Oxford, Oxford, UK **4** Max Planck Institute for Meteorology, Hamburg, Germany ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Marcel Stimberg

Reviewers:

- @macrocosme
- @tomvothecoder

Submitted: 22 September 2025
Published: unpublished

Summary

Xbitinfo analyses datasets based on their bitwise real information content and applies lossy compression accordingly. Xbitinfo provides additional tools to visualize the information histograms and to make informed decisions on the real information threshold that is subsequently used as the preserved precision during the compression of arrays of floating-point numbers. In contrast, the false information is rounded to zero using bitrounding. Lossless compression subsequently exploits the high compressibility from tailing zero mantissa bits. Xbitinfo's functionality supports xarray datasets to interact with a range of common input and output dataformats, including all numcodecs compression algorithms.

Statement of need

The geosciences, similar to other research fields, are generating more and more data both through simulations and observations. At the same time, data storage solutions have not increased at the same pace. In addition, more and more data is stored in the cloud and egress fees and network speeds are more and more of a concern. Compression algorithms can help to reduce the pressure on these components significantly and are therefore commonly used.

Lossless compressors like Zlib or Zstandard encode datasets exploiting redundancies without losing any information. This is often unnecessarily conservative as not all the bits are meaningful, i.e. they do not contain real information. They often encode unnecessarily high precision of floating-point numbers, several orders of magnitude higher than the uncertainty of the data (arising from e.g. model, numerical, observational or rounding errors) itself. Lossy compression is therefore often used, sacrificing bits with little to no real information, and from image and audio compression, JPEG and MP3 are two promintent examples. Geospatial data lacks a similarly widely accepted compression standard.

JPEG and MP3 use perceptual models of the human visual and auditory system to decide on whether or not to keep information (Standardization, 1993, 1994). Applied to geospatial data, the visual approach is acceptable for the publication of a scientific figure, however, it may not yield a tolerable compression error for the original data that still undergoes mathematical operations, like gradients. Commonly used with geospatial data is linear quantization as it is a standard algorithm supported by the GRIB format. It encodes the min-max range of the data into evenly, or linearly, spaced quanta and enumerates those using integers. The issue with linear quantization is however that it often is not a good mapping for geophysical quantities with a more logarithmical distribution. In practice, the number of preserved mantissa bits in the quantization process is often applied to an entire set of variables and dimensions. As a

*Now at XING, Hamburg, Germany

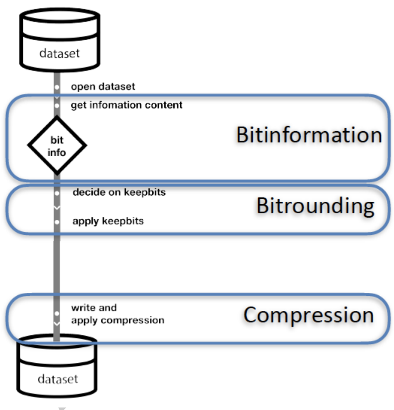
The Journal of Open Source Software

consequence, some variables have too little information preserved while others kept too much (false) information.

Klöwer et al. (2021) has developed an algorithm that can distinguish between real and false information content based on information theory. It further allows to set a threshold for the real information content that shall be preserved in case additional compression is needed beyond the filtering of false information.

As typical for lossy compression, parameters can be set to influence the loss. In case of the bitinformation algorithm, the `inflevel` parameter can be set to decide on the percentage of real information content to be preserved. The compression can therefore be split into three main stages:

- **Bitinformation**: analysing the bitinformation content
- **Bitrounding**:
  - deciding on information content to keep (`inflevel`)
  - translate `inflevel` to mantissa bits to keep (`keepbits`) after rounding
  - bitrounding according to keepbits
- **Compression**:
  - applying lossless compression



All stages are shown in **??**.

The Bitrounding is supported by many libraries (e.g. CDO, netCDF, numcodecs). One can also set the `inflevel` and get the according number of keepbits with the Julia implementation provided by Klöwer et al. (2021). However, for a user with a workflow that is otherwise Python-based this is not convenient. In practice, the decision on how much real information to keep needs testing with the downstream tools and is often an iterative process to ensure consistent behaviour with the original dataset. The gathering of the bitinformation and the decision on the bitrounding parameters are therefore often not immediately following each other and are interrupted by visual inspection and testing (see Figure 1).

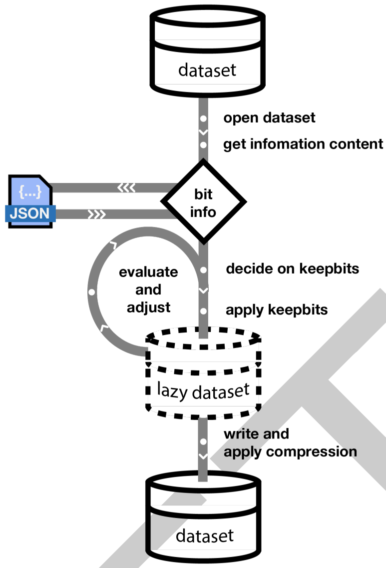


Figure 1: Xbitinfo workflow with the addition of storing the computational expensive retrieval of the bitinformation content in a JSON file for later reference and the ability to evaluate and adjust the keepbits on subsets of the original dataset.

Xbitinfo therefore provides additional convenience functions over Klöwer et al. (2021) to analyse, filter and visualize the bitwise real information content. Because Xbitinfo operates on xarray datasets it can also handle a large variety of input and output formats, like netCDF and Zarr and naturally fits into other scientific workflows. Thanks to the xarray-compatibility it can also make use of a wide range of modern lossless compression algorithms that are implemented for the specific output data formats to utilize the additional compression gains due to reduced information.

Xbitinfo provides two backends for the calculation of the real bitwise information content, one wraps the latest Julia implementation in BitInformation.jl provided with Klöwer et al. (2021) for consistency and the other uses numpy to be dask compatible and therefore is more performant when compressing in parallel.

Example

To compress a dataset based on its real bitwise information content with xbitinfo follows the following steps:

```python
import xarray as xr
import xbitinfo as xb

ds = xr.open_dataset("/path/to/input/file")
bitinfo = xb.get_bitinformation(ds)
keepbits = xb.get_keepbits(bitinfo, inflevel=0.95)
ds = xb.xr_bitround(ds, keepbits)
ds.to_compressed_zarr("/path/to/output/file")
```

Remarks

It should be noted that the BitInformation algorithm relies on uncompressed data that hasn't been manipulated beforehand. A common issue is that climate model output has been linearly quantized during its generation, e.g. because it has been written to the GRIB format. Such datasets should be handeled with care as the bitinformation often contains artificial information resulting in too many keepbits. Filters to capture those cases are currently developed within xbitinfo to warn the user.

Acknowledgements

We acknowledge all GitHub contributors that helped and continue to help to improve Xbitinfo and its dependencies. We also thank the ECMWF Code4Earth and Google Summer of Code programs for providing funds to support students to work on new features of Xbitinfo.

References

Klöwer, M., Razinger, M., Dominguez, J. J., Düben, P. D., & Palmer, T. N. (2021). Compressing atmospheric data into its real information content. *Nature Computational Science*, *1*(11, 11), 713–724. https://doi.org/10.1038/s43588-021-00156-2

Standardization, I. O. for. (1993). *ISO/IEC 11172-3:1993*. ISO. https://www.iso.org/standard/22412.html

Standardization, I. O. for. (1994). *ISO/IEC 10918-1:1994*. ISO. https://www.iso.org/standard/18902.html