Data compression and reduction for Earth System Sciences datasets in practice

original

safe
~25x

excessive
~10,000x

EGU 26, SC 2.5, 08.05.2026

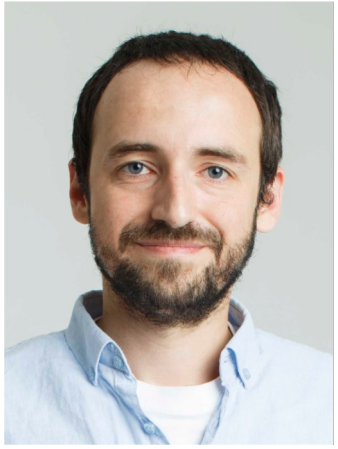
EuroHPC Joint Undertaking

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) under grant agreement No 101093054.

Co-funded by the European Union

Conveners: Juniper Tyree[1], Sara Faghih-Naini[2], Clément Bouvier[1] and Oriol Tintó Prims[3]






Contributors: Martin Burtscher[4], Langwen Huang[5], Milan Klöwer[6], Sam Li[7], Peter Lindstrom[8] and Robert Underwood[9,10]

[1] INAR, University of Helsinki, [2] ECMWF, [3] Barcelona Supercomputing Center, [4] Texas State University, [5] ETH Zurich, [6] University of Oxford, [7] NVIDIA, [8] Lawrence Livermore National Laboratory, [9] Argonne National Laboratory, [10] University of Chicago

EGU 26, SC 2.5, 08.05.2026

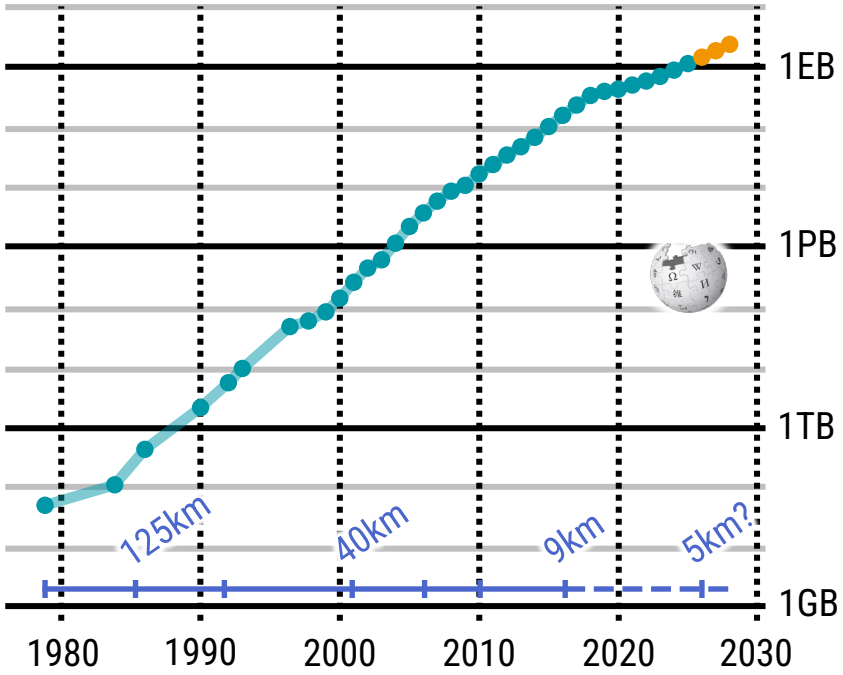
EuroHPC Joint Undertaking

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) under grant agreement No 101093054.

Co-funded by the European Union

Problem: Fast Growth in Earth System Science Data Volumes

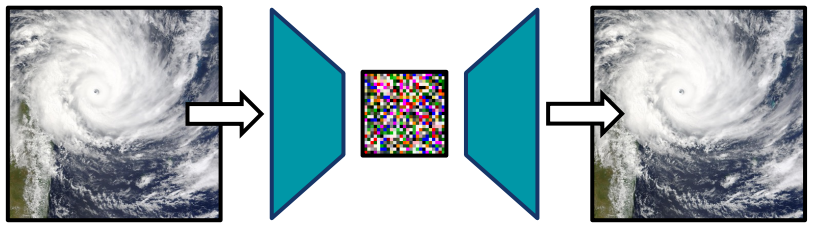
Exp. Growth of the ECMWF Archive Size, log



European Centre for Medium-Range Weather Forecasts archive size
Dataset provided by Mike Hawkins (ECMWF), Figure adapted from Milan Klöwer (Oxford)

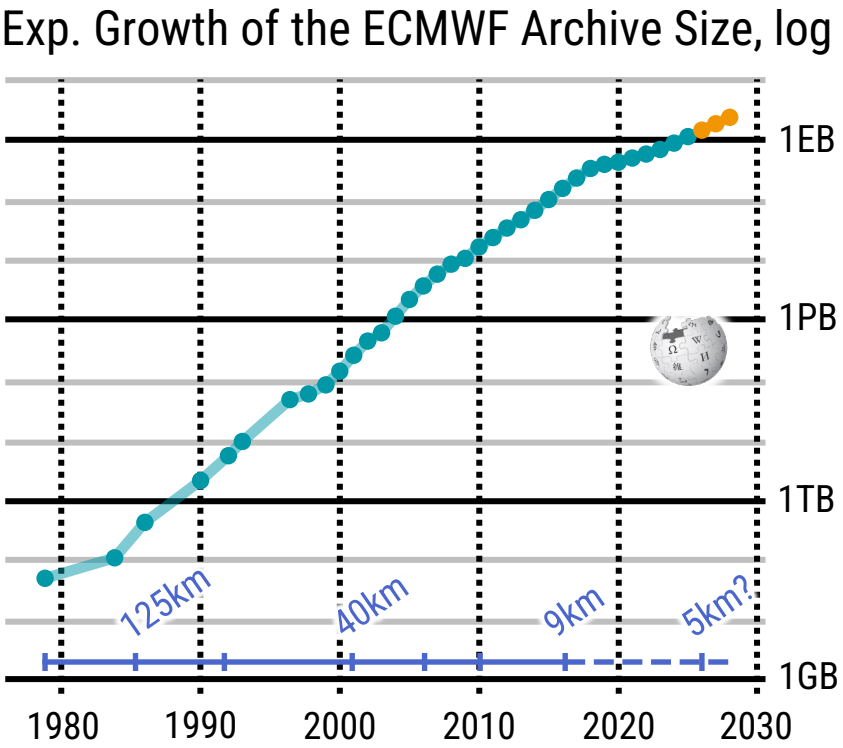
Wikipedia Logo v2 by Nohat. Available from https://en.wikipedia.org/wiki/File:Wikipedia-logo-v2.svg (CC BY-SA 3.0)

We need lossless $\times 2\text{–}6$ **compression**



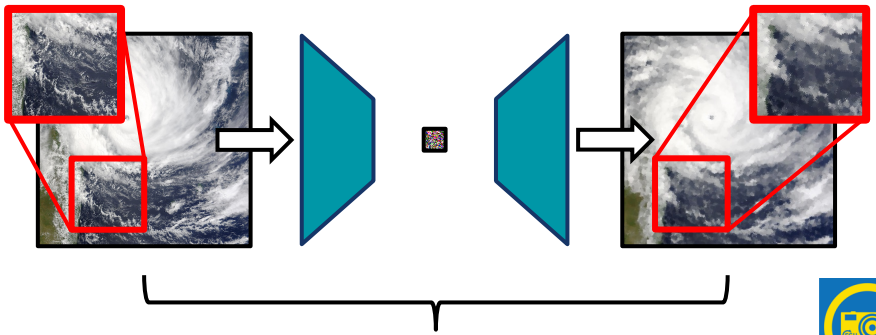
lossless = same bits

Problem: Fast Growth in Earth System Science Data Volumes



European Centre for Medium-Range Weather Forecasts archive size
Dataset provided by Mike Hawkins (ECMWF), Figure adapted from Milan Klöwer (Oxford)

We need **lossy compression** $\times 1000$



lossy = approximation

Solution: Some ready-to-use Scientific Compressors

Bit Rounding Milan Klöwer	**EBCC** Langwen Huang	**LibPressio** Robert Underwood
ZFP Peter Lindstrom*	**LC** Martin Burtscher*	**Safeguards** Juniper Tyree
SPERR Sam Li	**SZ** Robert Underwood	

* materials presented by Juniper Tyree

Quick Glossary:

Compression

- Lossless
- Lossy

Codec

- Filter
- Compressor
- **Meta-Compressor**

Error Bound

- Pointwise Absolute
- Pointwise Relative
- Range-Relative

IEEE 754 Float: $(-1)^{s} \cdot 2^{e-b} \cdot 1.m \ldots$ **with bit pattern** seeeeemmmmmmmmm

SC2.5 Overview:

Compressor Pitch Presentations

Introduction to Compression with numcodecs

Try out Compression Yourself

Discussion on Community Wishes and Concerns

SC2.5 Overview:

Compressor Pitch Presentations

Introduction to Compression with numcodecs

Try out Compression Yourself

Discussion on Community Wishes and Concerns

numcodecs: Simple Compression Framework in Python

```python
import numcodecs
import numpy as np

codec = numcodecs.Zlib(level=6)

config = codec.get_config()
codec = numcodecs.get_codec(config)

data = np.linspace(-10, 10, 41)

encoded = codec.encode(data)

decoded = codec.decode(encoded)
decoded = codec.decode(encoded, out=np.empty_like(data))
```

Creating codecs

Recreating from config

Encode / Compress

Decode / Decompress

numcodecs-wasm: **Bit-Reproducible** Cross-Platform **Compression**

Why: Outreach, Interactive Documentation, Reproducible Science

Expose pre-compiled scientific compressors to numcodecs with WASM

numcodecs-*	**numcodecs-wasm-***	**numcodecs-wasm**
compile e.g. SZ3 and adapt to numcodecs	WASM artifact package on PyPi	WASM engine executes codecs

Cross-platform bytecode with limited instruction set but 2-3x slower [1]
Deterministic single-threaded ISA subset with e.g. only canonical NaNs
Compressors run in **isolated 4GB sandboxes**: no integration hell

[1] Jangda, A., Powers, B., Berger, E., & Guha, A. (2019). Not So Fast: Analyzing the Performance of WebAssembly vs. Native Code. *arXiv.org*. Available from: doi:10.5555/3358807.3358817.

numcodecs-wasm

`numcodecs` compression for codecs compiled to WebAssembly.

`numcodecs_wasm` : provides the `WasmCodecMeta` meta class to load a codec from a WebAssembly component into a fresh Python class.

The following Python packages, all published independently on PyPi, use `numcodecs_wasm` to provide their codecs:

- `numcodecs_wasm_asinh` : $\operatorname{asinh}(x)$ codec
- `numcodecs_wasm_bit_round` : bit rounding codec
- `numcodecs_wasm_ebcc` : EBCC codec
- `numcodecs_wasm_fixed_offset_scale` : $\frac{x-o}{s}$ codec
- `numcodecs_wasm_fourier_network` : fourier feature neural network codec
- `numcodecs_wasm_identity` : identity codec
- `numcodecs_wasm_jpeg2000` : JPEG 2000 codec
- `numcodecs_wasm_lc` : LC codec
- `numcodecs_wasm_linear_quantize` : linear quantization codec
- `numcodecs_wasm_log` : $\ln(x)$ codec
- `numcodecs_wasm_pco` : pcodec
- `numcodecs_wasm_qpet_sperr` : QPET-SPERR codec
- `numcodecs_wasm_random_projection` : random projection codec
- `numcodecs_wasm_reinterpret` : binary reinterpret codec
- `numcodecs_wasm_round` : rounding codec
- `numcodecs_wasm_sperr` : SPERR codec
- `numcodecs_wasm_stochastic_rounding` : stochastic rounding codec
- `numcodecs_wasm_swizzle_reshape` : array axis swizzle and reshape codec
- `numcodecs_wasm_sz3` : SZ3 codec
- `numcodecs_wasm_tthresh` : Tthresh codec
- `numcodecs_wasm_uniform_noise` : uniform noise codec
- `numcodecs_wasm_zfp` : ZFP codec
- `numcodecs_wasm_zlib` : zlib codec
- `numcodecs_wasm_zstd` : Zstdandard codec

https://numcodecs-wasm.rtfd.io/

SC2.5 Overview:

Compressor Pitch Presentations

Introduction to Compression with numcodecs

Try out Compression Yourself

Discussion on Community Wishes and Concerns

Try Compression Yourself: Getting Started

(a) Local Installation

```
git clone https://github.com/climet-eu/egu26-compression-sc2-5.git
cd egu26-compression-sc2-5
git lfs install
git lfs fetch --all
uv sync
uv run jupyter lab
```

Please ask,
if you need any help!

(b) Online Laboratory

Multi-memory (WebAssembly)

Limited availability

(c) WiFi Laboratory

2.4G 5G

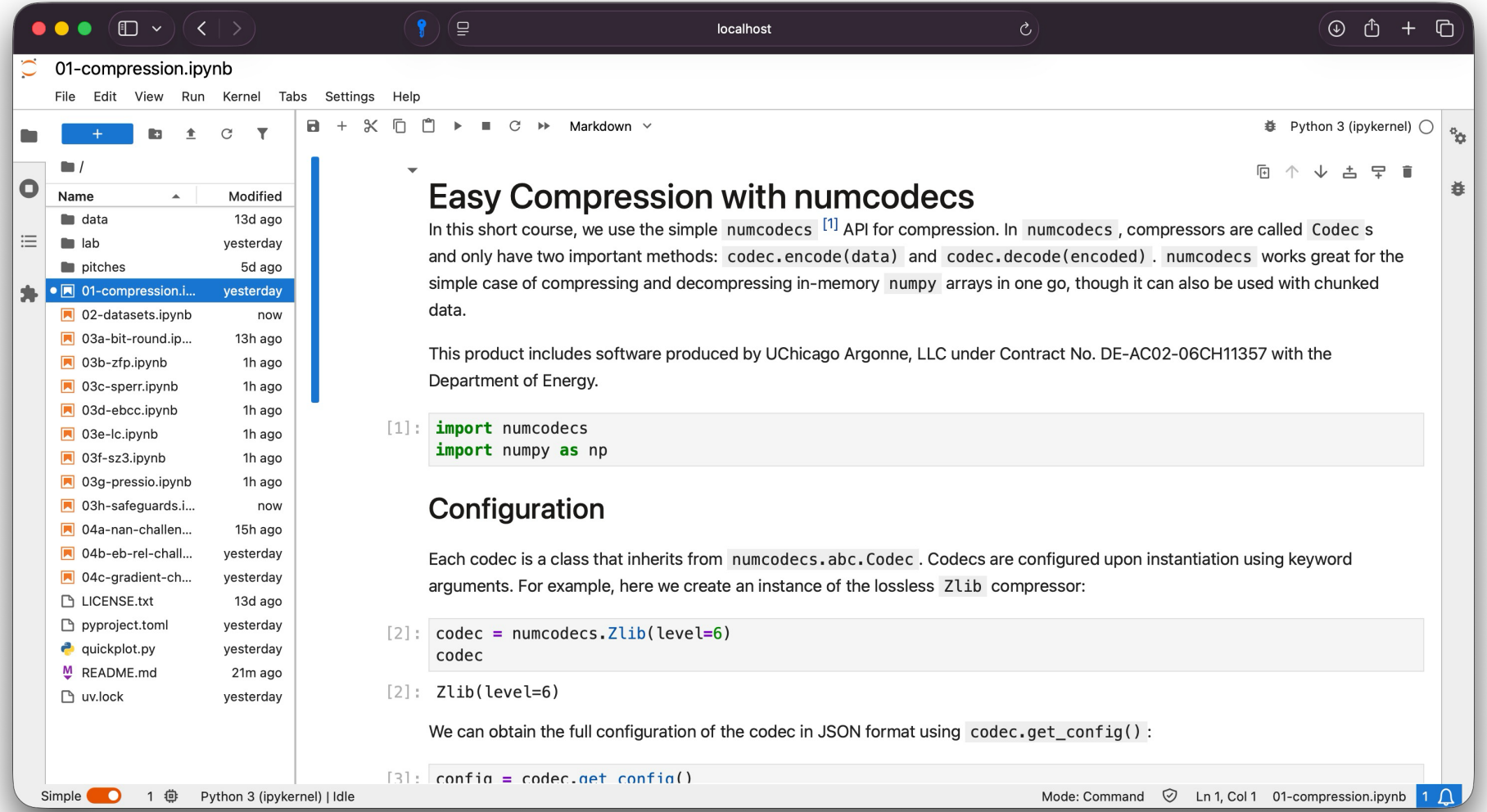
SSI: EGU26-SC2-5-Guest
SSI: EGU26-SC2-5-Guest-5G
Password: compression

Then egu.lab.climet.eu

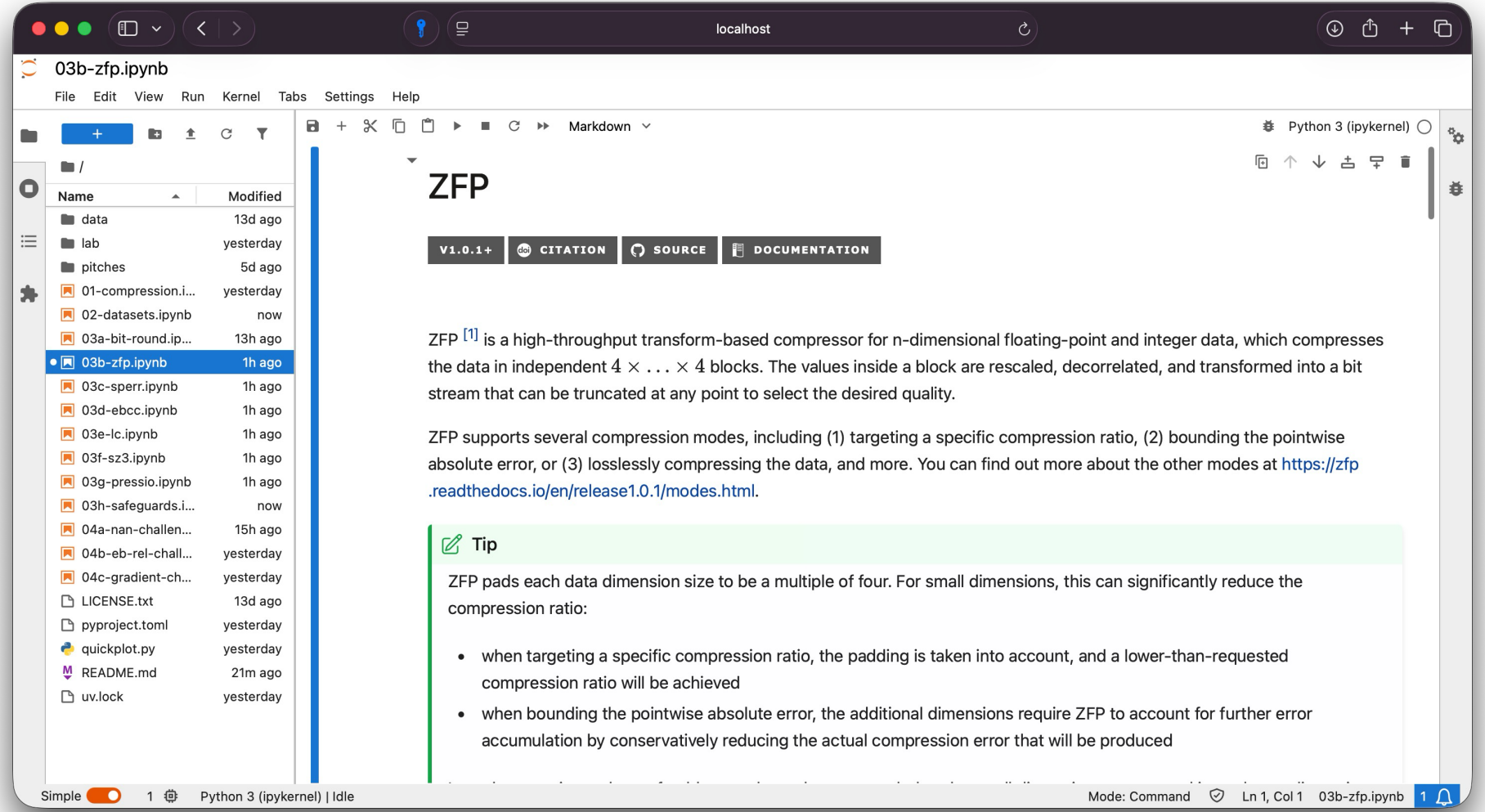
iOS: turn off private relay
HTTPS certificate is self-signed

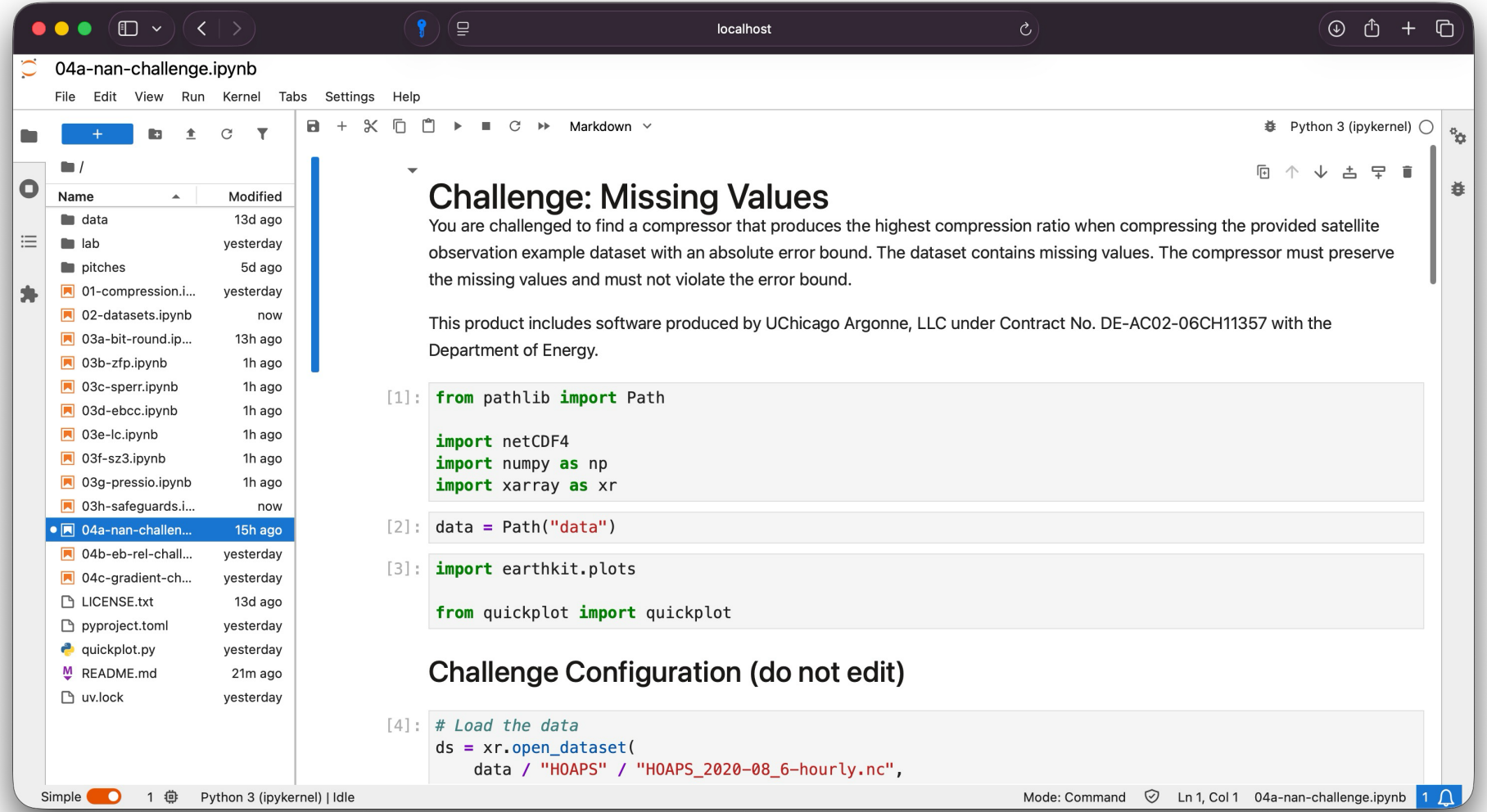
Materials on GitHub: https://github.com/climet-eu/egu26-compression-sc2-5 or ask for our USB

Sharing is encouraged


localhost
01-compression.ipynb
File Edit View Run Kernel Tabs Settings Help
Markdown
Python 3 (ipykernel)
/
Name
Modified
data 13d ago
lab yesterday
pitches 5d ago
01-compression.i... yesterday
02-datasets.ipynb now
03a-bit-round.ip... 13h ago
03b-zfp.ipynb 1h ago
03c-sperr.ipynb 1h ago
03d-ebcc.ipynb 1h ago
03e-lc.ipynb 1h ago
03f-sz3.ipynb 1h ago
03g-pressio.ipynb 1h ago
03h-safeguards.i... now
04a-nan-challen... 15h ago
04b-eb-rel-chall... yesterday
04c-gradient-ch... yesterday
LICENSE.txt 13d ago
pyproject.toml yesterday
quickplot.py yesterday
README.md 21m ago
uv.lock yesterday
Easy Compression with numcodecs
In this short course, we use the simple numcodecs [1] API for compression. In numcodecs, compressors are called Codecs and only have two important methods: codec.encode(data) and codec.decode(encoded). numcodecs works great for the simple case of compressing and decompressing in-memory numpy arrays in one go, though it can also be used with chunked data.
This product includes software produced by UChicago Argonne, LLC under Contract No. DE-AC02-06CH11357 with the Department of Energy.
[1]:
import numcodecs
import numpy as np
Configuration
Each codec is a class that inherits from numcodecs.abc.Codec. Codecs are configured upon instantiation using keyword arguments. For example, here we create an instance of the lossless Zlib compressor:
[2]:
codec = numcodecs.Zlib(level=6)
codec
[2]:
Zlib(level=6)
We can obtain the full configuration of the codec in JSON format using codec.get_config():
[3]:
config = codec.get_config()
Simple
1
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
01-compression.ipynb
1

localhost
02-datasets.ipynb
File Edit View Run Kernel Tabs Settings Help
Markdown
Python 3 (ipykernel)
/
Name
Modified
data 13d ago
lab yesterday
pitches 5d ago
01-compression.i... yesterday
02-datasets.ipynb now
03a-bit-round.ip... 13h ago
03b-zfp.ipynb 1h ago
03c-sperr.ipynb 1h ago
03d-ebcc.ipynb 1h ago
03e-lc.ipynb 1h ago
03f-sz3.ipynb 1h ago
03g-pressio.ipynb 1h ago
03h-safeguards.i... now
04a-nan-challen... 15h ago
04b-eb-rel-chall... yesterday
04c-gradient-ch... yesterday
LICENSE.txt 13d ago
pyproject.toml yesterday
quickplot.py yesterday
README.md 21m ago
uv.lock yesterday
Datasets
We have provided several small example datasets for this short course, which have all been preprocessed to be in NetCDF4 format and on a regular latitude-longitude grid. The original datasets and instructions [1] are published [2] as part of the Online Laboratory for Data Compression in Climate Science and Meteorology [3].
Here, we load all datasets using xarray and visualise them using our quickplot(da) helper function.
[1]:
from pathlib import Path
import humanize
import netCDF4
import numpy as np
import xarray as xr
[2]:
data = Path("data")
[3]:
import earthkit.plots
from quickplot import quickplot
[4]:
DATASET_PATHS = [
data / "HOAPS" / "HOAPS_2020-08_6-hourly.nc",
data / "ICON-XPP" / "ICONXPP_regridded_t_025deg_steps_2026-05_06.nc",
data / "ICON-XPP" / "ICONXPP_regridded_tp_025deg_steps_2026-05_06.nc",
data / "NextGEMS_EW3_ICON_ngc4008" / "NextGEMS_regridded_hus_025deg_steps_44_45.nc",
data / "NextGEMS_EW3_ICON_ngc4008" / "NextGEMS_regridded_pr_025deg_steps_44_45.nc",
Simple
1
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
02-datasets.ipynb
1


localhost
03b-zfp.ipynb
File Edit View Run Kernel Tabs Settings Help
Markdown
Python 3 (ipykernel)
/
Name
Modified
data 13d ago
lab yesterday
pitches 5d ago
01-compression.i... yesterday
02-datasets.ipynb now
03a-bit-round.ip... 13h ago
03b-zfp.ipynb 1h ago
03c-sperr.ipynb 1h ago
03d-ebcc.ipynb 1h ago
03e-lc.ipynb 1h ago
03f-sz3.ipynb 1h ago
03g-pressio.ipynb 1h ago
03h-safeguards.i... now
04a-nan-challen... 15h ago
04b-eb-rel-chall... yesterday
04c-gradient-ch... yesterday
LICENSE.txt 13d ago
pyproject.toml yesterday
quickplot.py yesterday
README.md 21m ago
uv.lock yesterday
ZFP
V1.0.1+
CITATION
SOURCE
DOCUMENTATION
ZFP [1] is a high-throughput transform-based compressor for n-dimensional floating-point and integer data, which compresses the data in independent $4 \times \ldots \times 4$ blocks. The values inside a block are rescaled, decorrelated, and transformed into a bit stream that can be truncated at any point to select the desired quality.
ZFP supports several compression modes, including (1) targeting a specific compression ratio, (2) bounding the pointwise absolute error, or (3) losslessly compressing the data, and more. You can find out more about the other modes at https://zfp.readthedocs.io/en/release1.0.1/modes.html.
Tip
ZFP pads each data dimension size to be a multiple of four. For small dimensions, this can significantly reduce the compression ratio:
when targeting a specific compression ratio, the padding is taken into account, and a lower-than-requested compression ratio will be achieved
when bounding the pointwise absolute error, the additional dimensions require ZFP to account for further error accumulation by conservatively reducing the actual compression error that will be produced
Simple
1
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
03b-zfp.ipynb
1


localhost
04a-nan-challenge.ipynb
File Edit View Run Kernel Tabs Settings Help
Markdown
Python 3 (ipykernel)
/
Name
Modified
data 13d ago
lab yesterday
pitches 5d ago
01-compression.i... yesterday
02-datasets.ipynb now
03a-bit-round.ip... 13h ago
03b-zfp.ipynb 1h ago
03c-sperr.ipynb 1h ago
03d-ebcc.ipynb 1h ago
03e-lc.ipynb 1h ago
03f-sz3.ipynb 1h ago
03g-pressio.ipynb 1h ago
03h-safeguards.i... now
04a-nan-challen... 15h ago
04b-eb-rel-chall... yesterday
04c-gradient-ch... yesterday
LICENSE.txt 13d ago
pyproject.toml yesterday
quickplot.py yesterday
README.md 21m ago
uv.lock yesterday
Challenge: Missing Values
You are challenged to find a compressor that produces the highest compression ratio when compressing the provided satellite observation example dataset with an absolute error bound. The dataset contains missing values. The compressor must preserve the missing values and must not violate the error bound.
This product includes software produced by UChicago Argonne, LLC under Contract No. DE-AC02-06CH11357 with the Department of Energy.
[1]: from pathlib import Path
import netCDF4
import numpy as np
import xarray as xr
[2]: data = Path("data")
[3]: import earthkit.plots
from quickplot import quickplot
Challenge Configuration (do not edit)
[4]: # Load the data
ds = xr.open_dataset(
data / "HOAPS" / "HOAPS_2020-08_6-hourly.nc",
Simple
1
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
04a-nan-challenge.ipynb
1

04b-eb-rel-challenge.ipynb

File Edit View Run Kernel Tabs Settings Help

/

Name	Modified
data	13d ago
lab	yesterday
pitches	5d ago
01-compression.i...	yesterday
02-datasets.ipynb	now
03a-bit-round.ip...	13h ago
03b-zfp.ipynb	1h ago
03c-sperr.ipynb	1h ago
03d-ebcc.ipynb	1h ago
03e-lc.ipynb	1h ago
03f-sz3.ipynb	1h ago
03g-pressio.ipynb	1h ago
03h-safeguards.i...	now
04a-nan-challen...	15h ago
04b-eb-rel-chall...	yesterday
04c-gradient-ch...	yesterday
LICENSE.txt	13d ago
pyproject.toml	yesterday
quickplot.py	yesterday
README.md	21m ago
uv.lock	yesterday

Markdown

Python 3 (ipykernel)

Challenge: Pointwise Relative Error Bound

You are challenged to find a compressor that produces the highest compression ratio when compressing the provided precipitation example dataset with a pointwise relative error bound. The compressor must not violate the error bound.

[1]:
```
from pathlib import Path

import netCDF4
import numpy as np
import xarray as xr
```

[2]:
```
data = Path("data")
```

[3]:
```
import earthkit.plots

from quickplot import quickplot
```

Challenge Configuration (do not edit)

[4]:
```
# Load the data
ds = xr.open_dataset(
    data / "hplp" / "hplp_sfc_regridded_tp_025deg_steps_228_240.nc",
    engine="netcdf4",
```

04c-gradient-challenge.ipynb

File Edit View Run Kernel Tabs Settings Help

/

Name	Modified
data	13d ago
lab	yesterday
pitches	5d ago
01-compression.i...	yesterday
02-datasets.ipynb	now
03a-bit-round.ip...	13h ago
03b-zfp.ipynb	1h ago
03c-sperr.ipynb	1h ago
03d-ebcc.ipynb	1h ago
03e-lc.ipynb	1h ago
03f-sz3.ipynb	1h ago
03g-pressio.ipynb	1h ago
03h-safeguards.i...	now
04a-nan-challen...	15h ago
04b-eb-rel-chall...	yesterday
04c-gradient-ch...	yesterday
LICENSE.txt	13d ago
pyproject.toml	yesterday
quickplot.py	yesterday
README.md	21m ago
uv.lock	yesterday

Markdown

Python 3 (ipykernel)

Challenge: Absolute Error Bound over Spatial Gradient

You are challenged to find a compressor that produces the highest compression ratio when compressing the provided specific humidity example dataset with an absolute error bound over the gradient along the periodic longitude axis. The compressor must not violate the error bound.

[1]:
```
from pathlib import Path

import netCDF4
import numpy as np
import xarray as xr
```

[2]:
```
data = Path("data")
```

[3]:
```
import earthkit.plots

from quickplot import quickplot
```

Challenge Configuration (do not edit)

[4]:
```
# Load the data
ds = xr.open_dataset(
    data / "NextGEMS_EW3_ICON_ngc4008" / "NextGEMS_regridded_hus_025deg_steps_44_45.nc",
```

Try Compression Yourself: Getting Started

(a) Local Installation

```
git clone https://github.com/climet-eu/egu26-compression-sc2-5.git
cd egu26-compression-sc2-5
git lfs install
git lfs fetch --all
uv sync
uv run jupyter lab
```

Please ask,
if you need any help!

(b) Online Laboratory

Multi-memory (WebAssembly)

Limited availability

(c) WiFi Laboratory

2.4G 5G

SSI: EGU26-SC2-5-Guest
SSI: EGU26-SC2-5-Guest-5G
Password: compression

Then egu.lab.climet.eu

iOS: turn off private relay
HTTPS certificate is self-signed

Materials on GitHub: https://github.com/climet-eu/egu26-compression-sc2-5 or ask for our USB

Sharing is encouraged

SC2.5 Overview:

Compressor Pitch Presentations

Introduction to Compression with numcodecs

Try out Compression Yourself

Discussion on Community Wishes and Concerns

Discussion with You and our Presenters:

What do You *need* from Lossy Compression?

What *concerns* You about Lossy Compression?

What would you like to tell compression developers?

Data compression and reduction for Earth System Sciences datasets in practice

Thank You for Your Participation!

Learn more about **High-performance computation and Data Compression** in Session **ESSI2.2** in -2.33 at 16:15–18:00

Join us at the Data Compression **ESiWACE3 Hackathon** on **15.09.-17.09.26** at ECMWF in Bonn, Germany or remotely

Sharing is encouraged

EuroHPC Joint Undertaking

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) under grant agreement No 101093054.

Co-funded by the European Union