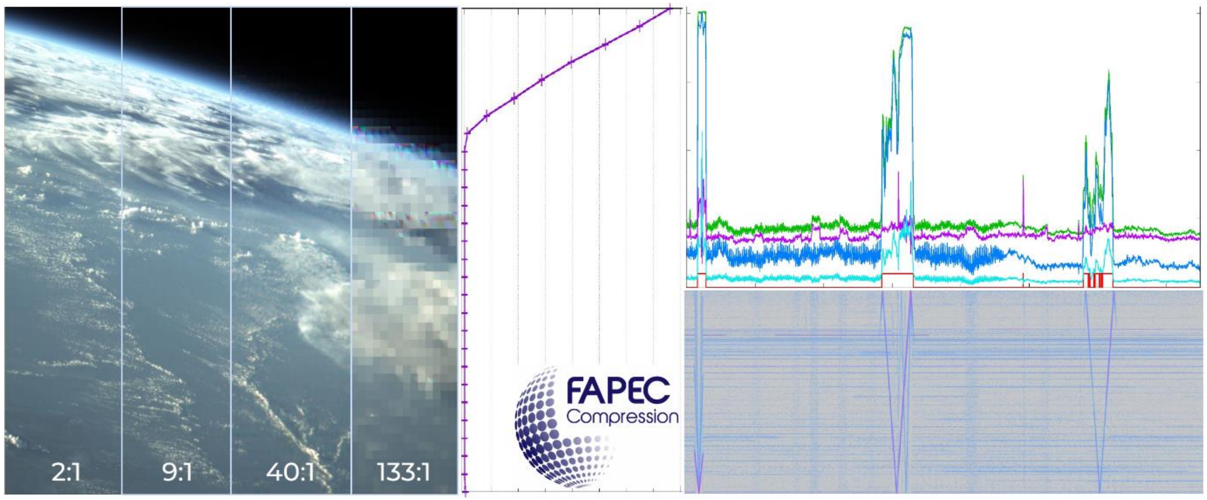
Versatile data compression software for sustained high-throughput in-orbit data acquisition

(a.k.a. **RICSDAC**: RF and Image Compression Software for Demanding Applications in Cubesats)

Final Review

Jordi Portell

on behalf of the FAPEC team at DAPCOM Data Services S.L.: Aniol Martí, Eloi Saus, Riccardo Iudica



Outline

- Overview of the activity

- Image data compression
- Radio-frequency data compression

  For both items:
  - Design updates
  - Tests and results (special focus on these)

- Conclusions

Outline

- Overview of the activity
- Image data compression
- Radio-frequency data compression
- Conclusions

OPS-SAT: an in-orbit laboratory

Photo: Lunghammer - TU Graz

- Technology demonstration cubesat by ESA
  - MityARM 5CSX (**dual core Cortex-A9, 800 MHz**)
  - Linux
  - 226 experiments registered, including FAPEC (exp. #100)
- OPS-SAT camera
  - 2048 x 1944 x 12-bit (raw image size: 8 MB)
  - Bayer colour filter array:
    4 “bands” (Red, Green1, Green2, Blue)
  - JPEG can be used, requiring onboard Bayer demosaicing
  - Up to 5 frames/s → up to 320 Mbps (40 MB/s) raw throughput
- Software Defined Radio
  - 12-bit in-phase & quadrature (I&Q) radio frequency data
  - Some tests at 3 Msamples/s → 96 Mbps (12 MB/s)

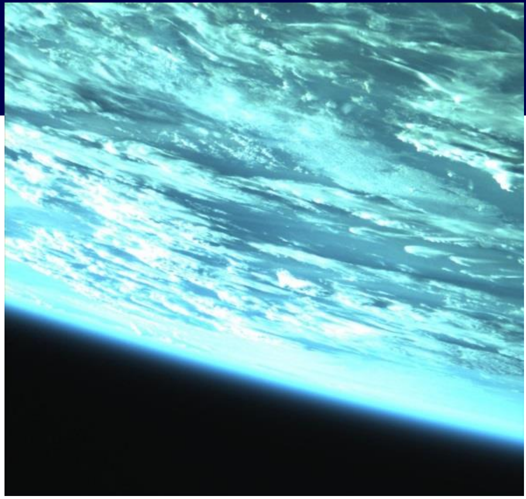
The FAPEC data compressor

Versatile data compression solution (onboard + onground applications)

- FAPEC **entropy coding core** (outlier-resilient)
- Suite of **pre-processing stages** including images (greyscale, multi/hyperspectral) and wave data (e.g. audio or RF), lossless and near-lossless
- Fast, multi-thread, encryption
- Basic **data analysis** capabilities
- ANSI C software implementation
- CLI + C/Python/Java API
- Currently being used in several earth observation satellites
- Free evaluation licenses: www.dapcom.es/get-fapec

FAPEC
22.0 Merganser

FAPEC in OPS-SAT



- FAPEC being used on-board OPS-SAT since 2020!
  - CILLIC lossy image compression invoked from CLI
  - Lots of images (and “videos”) downloaded
  - Ratios around 10, very good image quality

```
[29-11-2020 17:00:26] COMMAND  Uplink to SEPP: for f in /home/exp1000/toGround/edge/*.ims_rgb; do
  c='/home/exp100/fapec -q -chunk 512K -mt 1 -dtype 16 -cillic 2048 1944 1 x10 12 4 -lev 5
     -ow -o /home/exp100/toGround/'$(basename ${f%.*}.fapec); eval '$c $f >> /home/exp100/f.log'; done
[29-11-2020 17:00:33] DATA: START
[29-11-2020 17:00:33] DATA: STOP
[29-11-2020 17:00:34] COMMAND  Uplink to SEPP: cat /home/exp100/f.log; ls -larthR /home/exp100/toGround
[29-11-2020 17:00:41] DATA: START
[29-11-2020 17:00:41]
[29-11-2020 17:00:41] FAPEC Archiver - 20.0.0 Beta r2280 (2020-11-15)
[29-11-2020 17:00:41] (c) 2013-2020 DAPCOM Data Services S.L. - http://www.dapcom.es
[29-11-2020 17:00:41] 32/32 bit LE Restricted license for:
[29-11-2020 17:00:41]  ESA OPS-SAT
[29-11-2020 17:00:41]
[29-11-2020 17:00:41] Compressing 1 file into /home/exp100/toGround/img_msec_1606601765418_2.fapec...
[29-11-2020 17:00:41]  [1/1] /home/exp1000/toGround/edge/img_msec_1606601765418_2.ims_rgb (7.6 MB)...
[29-11-2020 17:00:41]
[29-11-2020 17:00:41] Done: 7.6 MB compressed to 0.8 MB (ratio 9.6467) in 0.8 seconds (9.0 MB/s)
[29-11-2020 17:00:41]
[29-11-2020 17:00:41] FAPEC Archiver - 20.0.0 Beta r2280 (2020-11-15)
[29-11-2020 17:00:41] (c) 2013-2020 DAPCOM Data Services S.L. - http://www.dapcom.es
[29-11-2020 17:00:41] 32/32 bit LE Restricted license for:
[29-11-2020 17:00:41]  ESA OPS-SAT
[29-11-2020 17:00:41]
[29-11-2020 17:00:41] Compressing 1 file into /home/exp100/toGround/img_msec_1606638723330_2.fapec...
[29-11-2020 17:00:41]  [1/1] /home/exp1000/toGround/edge/img_msec_1606638723330_2.ims_rgb (7.6 MB)...
[29-11-2020 17:00:41]
[29-11-2020 17:00:41] Done: 7.6 MB compressed to 0.8 MB (ratio 9.9437) in 0.8 seconds (9.3 MB/s)
[29-11-2020 17:00:41] /home/exp100/toGround:
[29-11-2020 17:00:41] -rw-r--r--    1 root     root      806.2K Nov 29 17:00 img_msec_1606601765418_2.fapec
[29-11-2020 17:00:41] -rw-r--r--    1 root     root      782.2K Nov 29 17:00 img_msec_1606638723330_2.fapec
```

Motivation of this work

- **ESA OSIP Call**: OPS-SAT Experiments Campaign
  - Call for proposals of additional experiments and developments for OPS-SAT
- FAPEC image and RF compression:
  - CILLIC and Wave algorithms already available as of FAPEC 22.0
  - Some limitations identified: ratios, lossy quality, speed
  - Add support for video compression
  - Add extra features: “Thumbnails” or basic data analysis to support downlink decisions
- Cubesats (and “New Space” in general):
  - Typically Linux-based software solutions
  - Agile developments → use COTS and ready-to-use software as much as possible
  - Zip, JPEG, JPEG2K, PNG, etc. → memory and CPU usage, limitations (e.g.: 16-bit hyperspectral images?)

Improve image and RF decorrelation algorithms for FAPEC

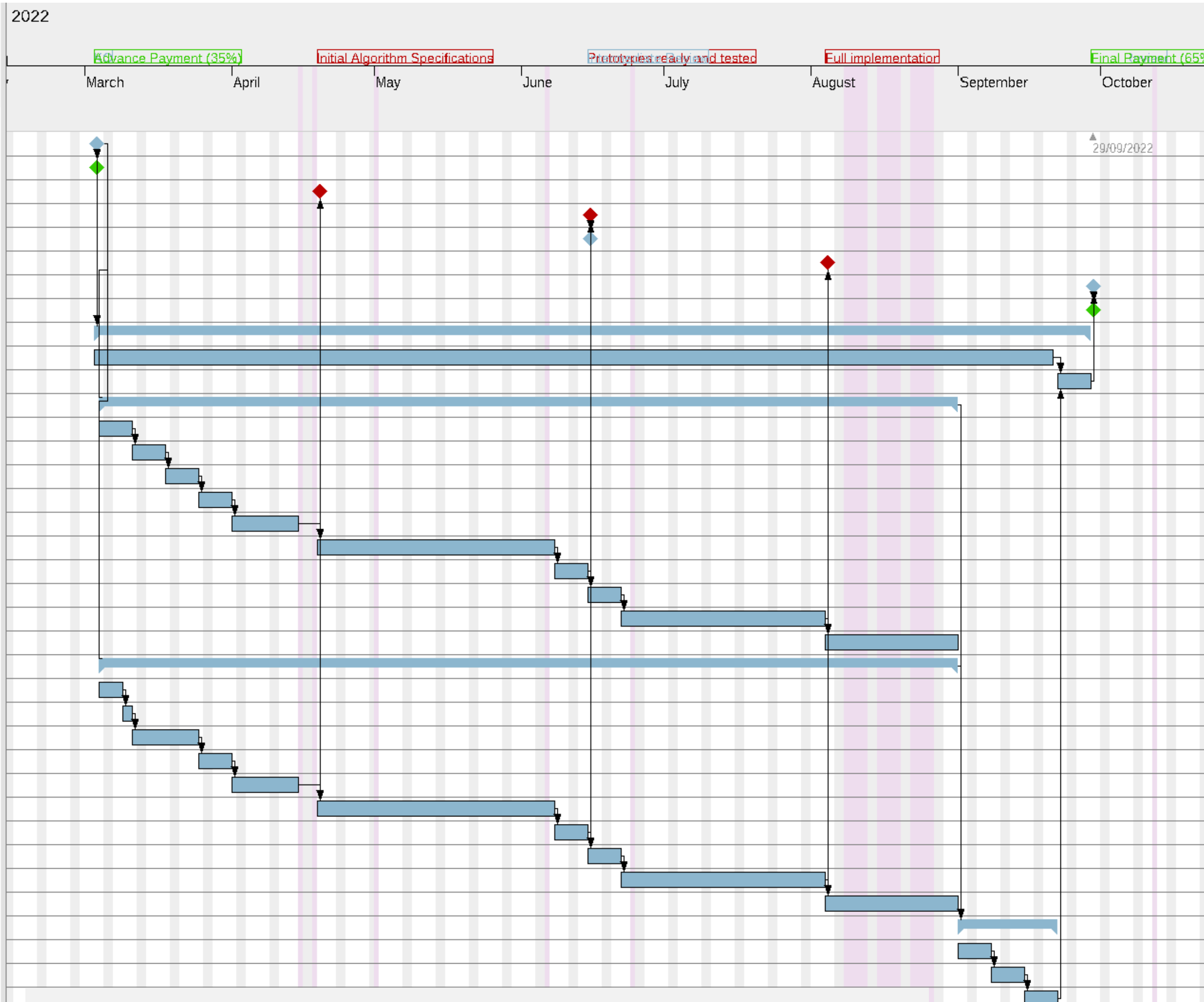
Provide a “de facto” standard for data compression in cubesats

Enable new use cases: continuous monitoring, smart downlink

Activity schedule

DAPCOM Data services

Name	Begin date	End date
KO	03/03/2022	03/03/2022
Advance Payment (35%)	03/03/2022	03/03/2022
Initial Algorithm Specifications	19/04/2022	19/04/2022
Prototypes ready and tested	15/06/2022	15/06/2022
Intermediate Review	15/06/2022	15/06/2022
Full implementation	04/08/2022	04/08/2022
Final Review	29/09/2022	29/09/2022
Final Payment (65%)	29/09/2022	29/09/2022
RIC-100: Management	03/03/2022	28/09/2022
105: Management tasks	03/03/2022	20/09/2022
110: Prep. Final Report	22/09/2022	28/09/2022
RIC-200: RF	04/03/2022	31/08/2022
205: Identify typ. RF signals	04/03/2022	10/03/2022
210: Get OPS-SAT data	11/03/2022	17/03/2022
215: Research RF methods	18/03/2022	24/03/2022
220: Review FAPEC Wave	25/03/2022	31/03/2022
225: Initial RF spec	01/04/2022	14/04/2022
230: Implement prototype	19/04/2022	07/06/2022
235: Initial tests on x86	08/06/2022	14/06/2022
240: Revise RF spec	15/06/2022	21/06/2022
245: Final implementation	22/06/2022	03/08/2022
250: Final tests	04/08/2022	31/08/2022
RIC-300: Image	04/03/2022	31/08/2022
305: Identify camera features	04/03/2022	08/03/2022
310: Get OPS-SAT images	09/03/2022	10/03/2022
315: Research image methods	11/03/2022	24/03/2022
320: Review FAPEC CILLIC	25/03/2022	31/03/2022
325: Initial Image spec	01/04/2022	14/04/2022
330: Implement prototype	19/04/2022	07/06/2022
335: Initial tests on x86	08/06/2022	14/06/2022
340: Revise Image spec	15/06/2022	21/06/2022
345: Final implementation	22/06/2022	03/08/2022
350: Final tests	04/08/2022	31/08/2022
RIC-400: Integ+Test+Val	01/09/2022	21/09/2022
405: Fast coder option	01/09/2022	07/09/2022
410: Final onground tests	08/09/2022	14/09/2022
415: Final onboard tests	15/09/2022	21/09/2022



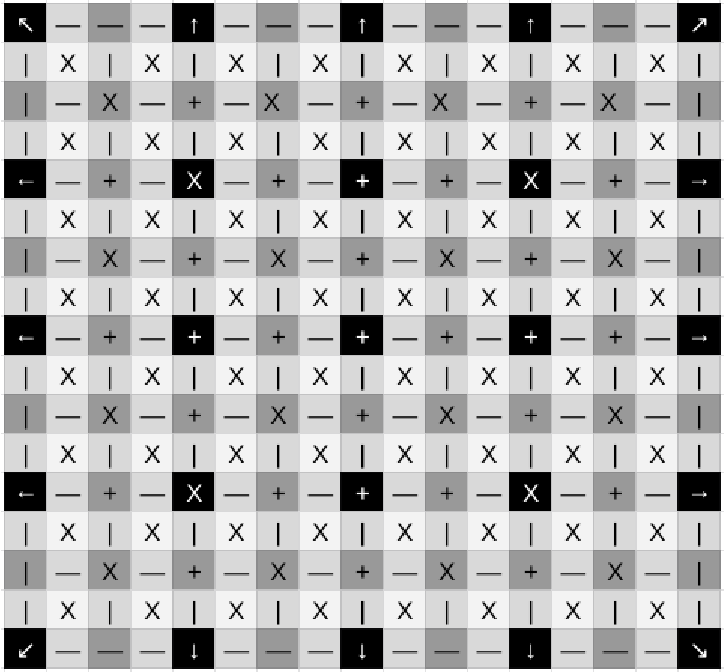
All tasks can be considered done (except onboard tests)

Some delays with respect to the original plan

Outline

- Overview of the activity
- **Image data compression**
- Radio-frequency data compression
- Conclusions

Improvements in the CILLIC algorithm



- Larger blocks: 17 x 17 pixels
  - More SIMD friendly
  - Thumbnails: 1/289th resolution
- Different pixel types
  - 9 x 9 "lattice" pixels (types 1 and 2)
  - 208 internal pixels
- Spatial and spectral decorrelators
  - Interpolation and inter-block estimators
  - Simplistic inter-band decorrelator, for speed
- Multi-band adaptiveness:
  - Determine best inter-band decorrelator once every few blocks
- Near-lossless and lossy options:
  - Revised approach to achieve higher ratios and better quality
- Flat blocks:
  - Smaller variations than quantization step
- Misalignment / motion estimation:
  - Brute force for now, to be optimized. Hill Climbing algorithm? Phase Correlation?
- Fast coding option:
  - FASEC entropy coding instead of the FAPEC core

FASEC algorithm

- Fast and simplistic signed-to-unsigned conversion
  - Equiv. to ”mapper” in CCSDS 121.0-B-2 (Laplacian → Geometric distrib.)
  - Quickly determine the maximum number of bits required to code a value
  - Just 5 arithmetic operations needed; no branching

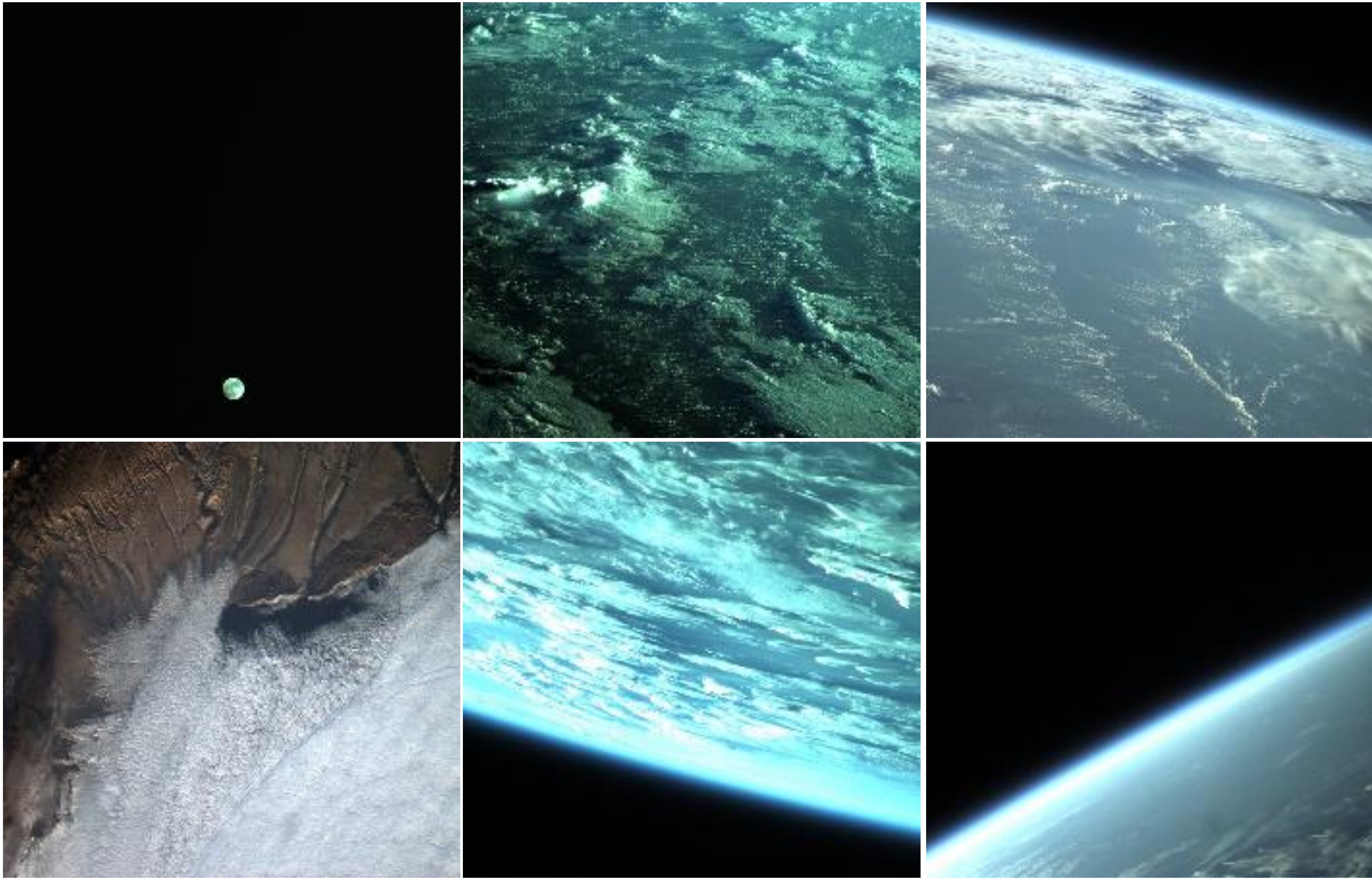
Residual	Code	Bits
0	0	0
+1	2	2
-1	3	2
+2	4	3
-2	5	3
…		
+255	510	9
-255	511	9

- Block-based operation
  - Fixed length: 8 samples per block
  - Simple determination of largest code required per block (just 7 logical operations needed)
  - Flag of 3 bits per block: “zeroes”, 2b/value, 3b/value, … 7b/value, “uncompressible”
  - Coding sets of 8 blocks: 3 bytes/set with flags + 8 blocks (each 0, 2, 3, … 7, 8-9 bytes)
  - Composition of 64-bit output code (written in 2 steps for some CPUs; e.g. some ARM)

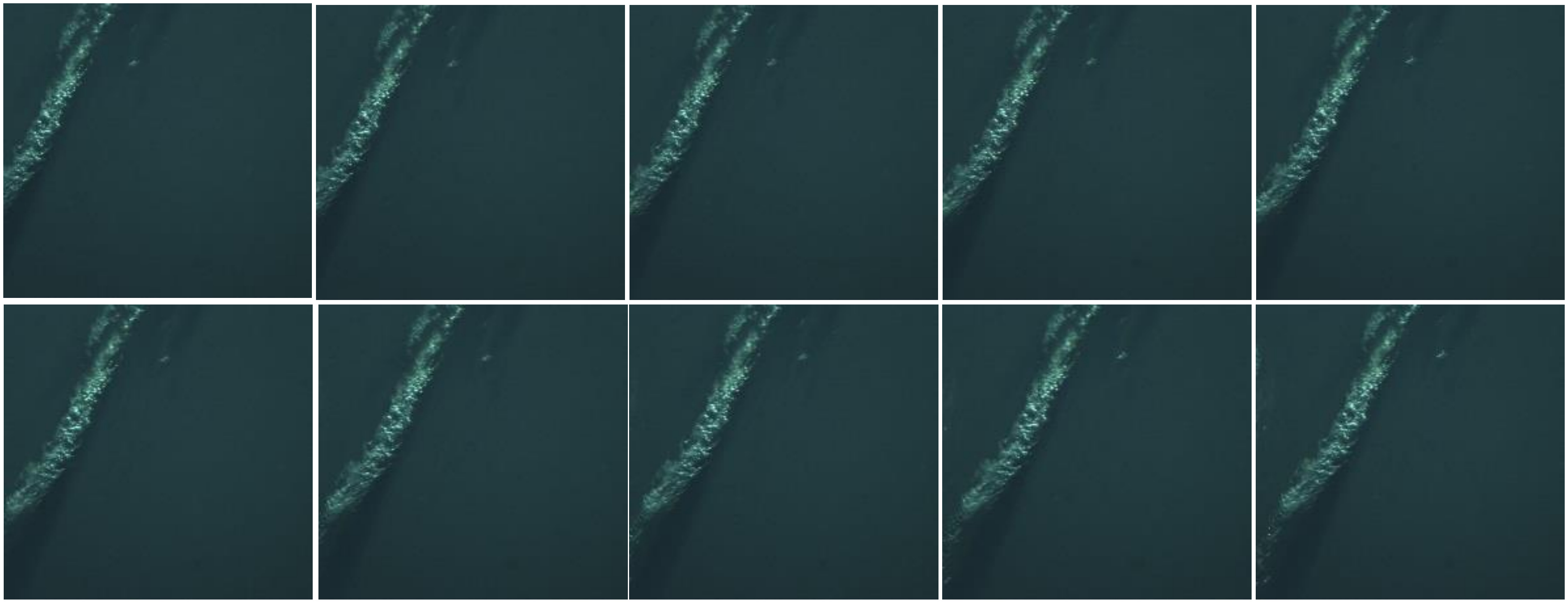
Max res. in blk	Bits/block	Ratio
0	3	**21.3**
±1	19	3.37
±3	27	2.37
±7	35	1.83
±15	43	1.49
±31	51	1.25
±63	59	1.08
±255	67-75	**0.95-0.85**

- Either standalone (delta decorrelator) or as coding core of CILLIC (“fast” option)

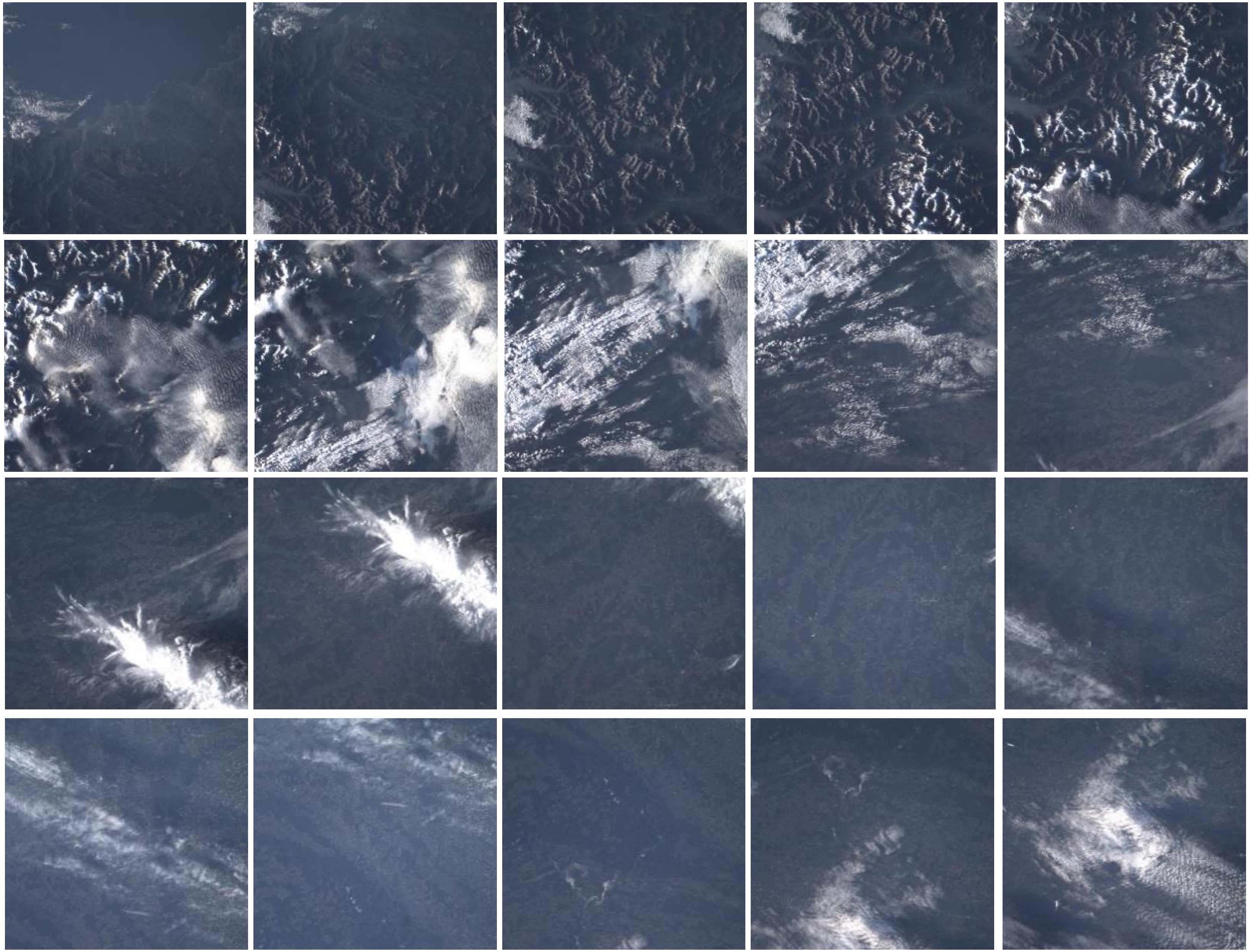
Test dataset 1: miscellaneous (lossy)



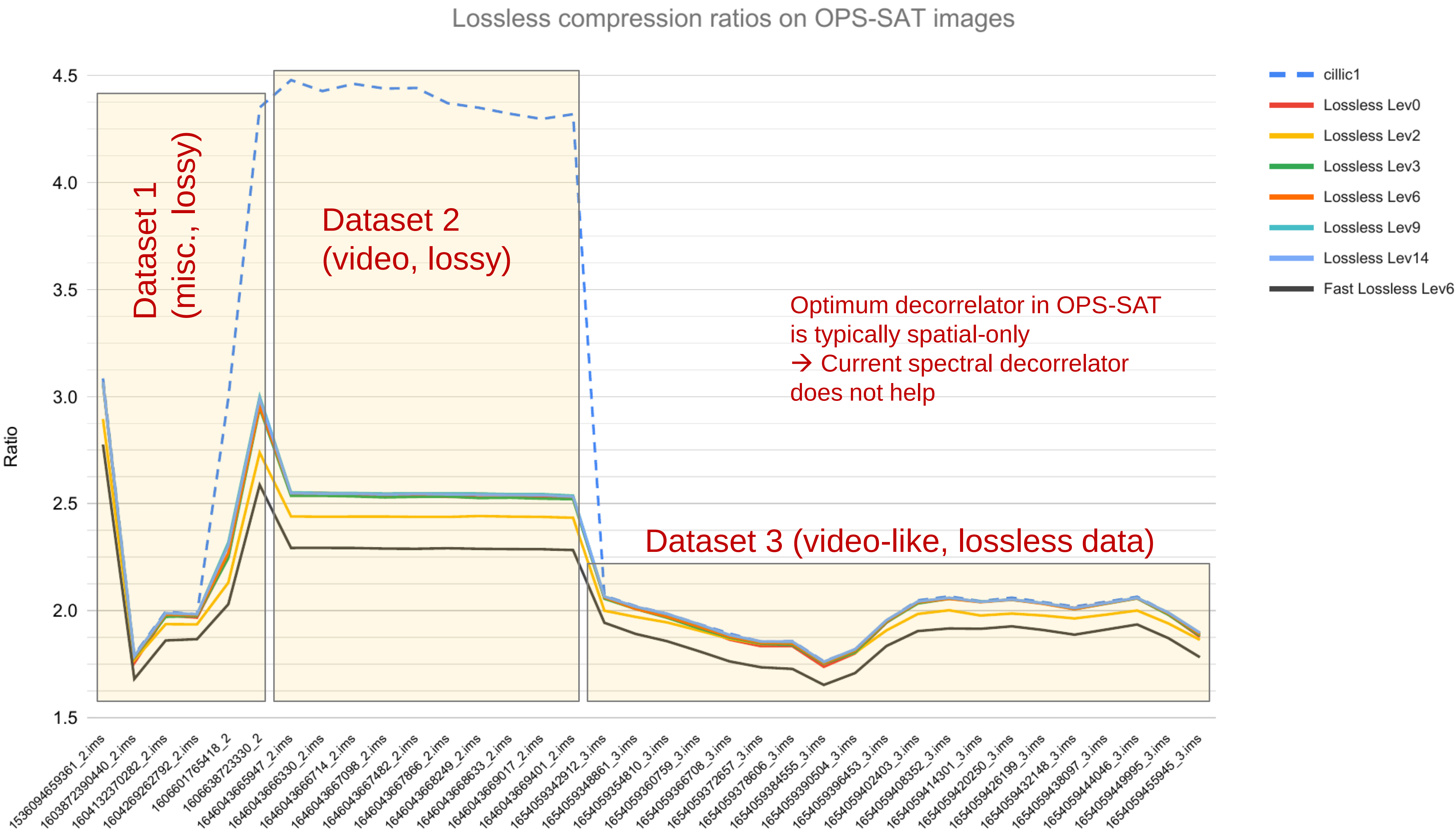
Test dataset 2: video (lossy)



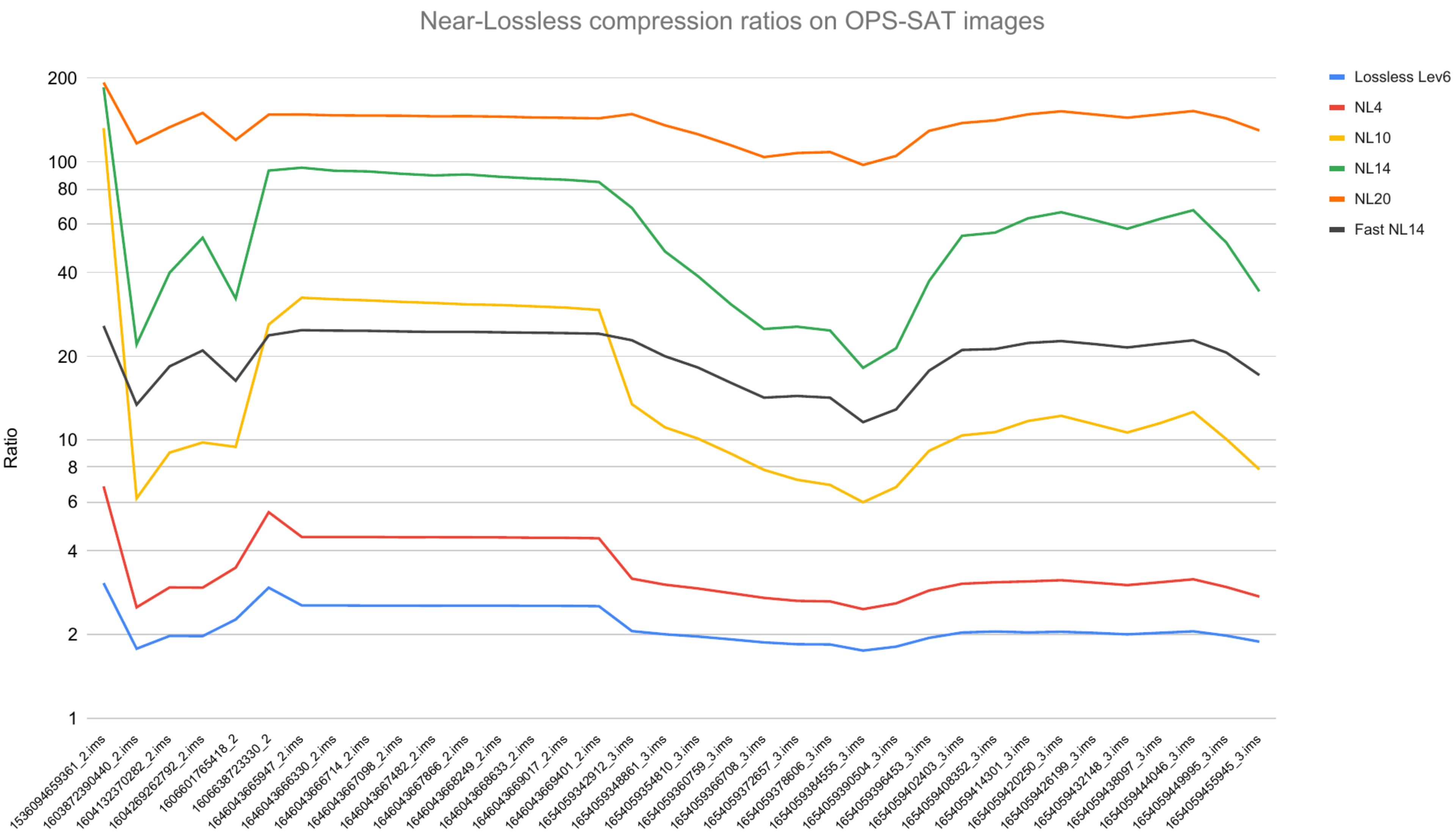
Test dataset 3: video-like (lossless)



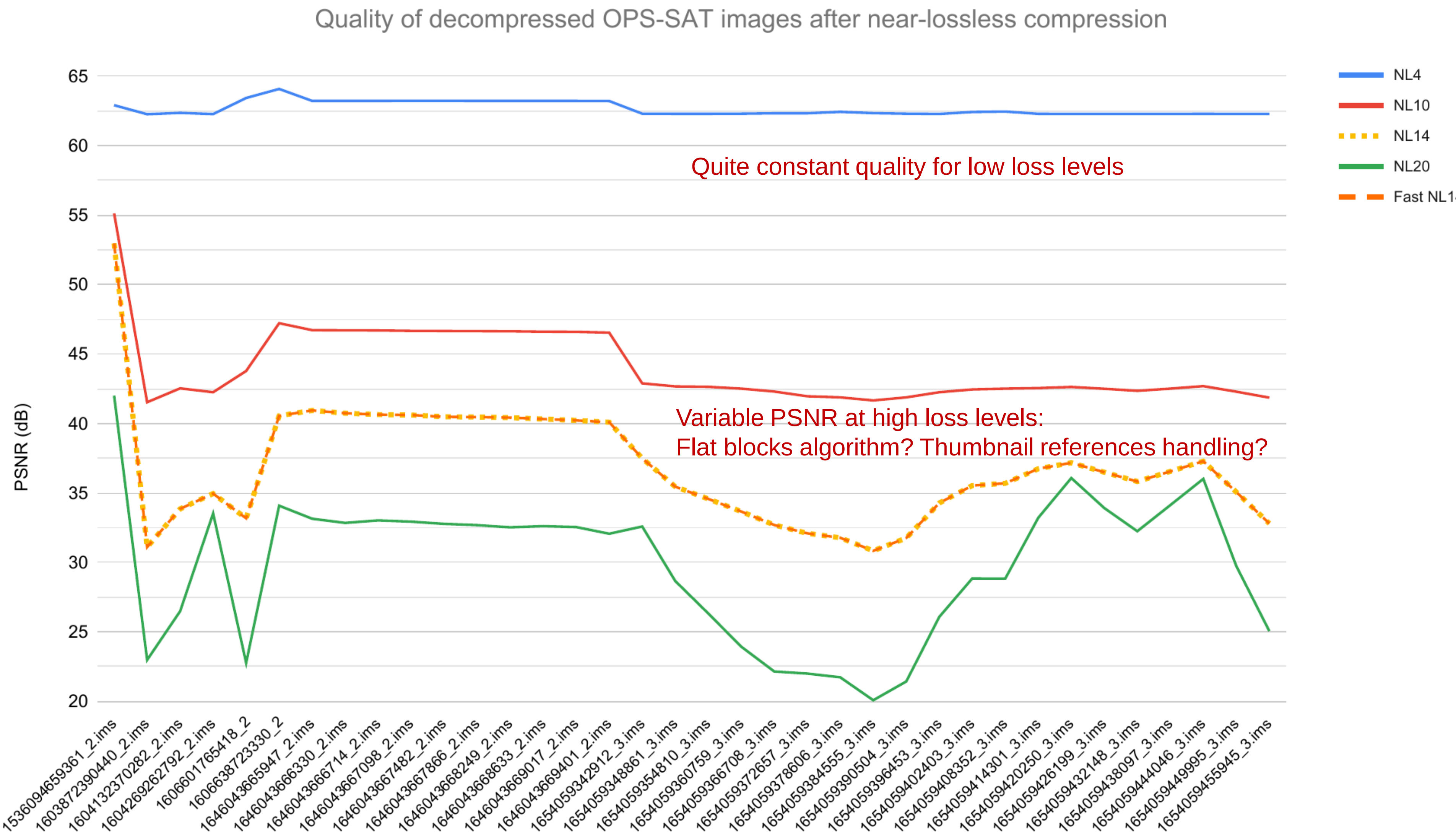
Lossless compression ratios



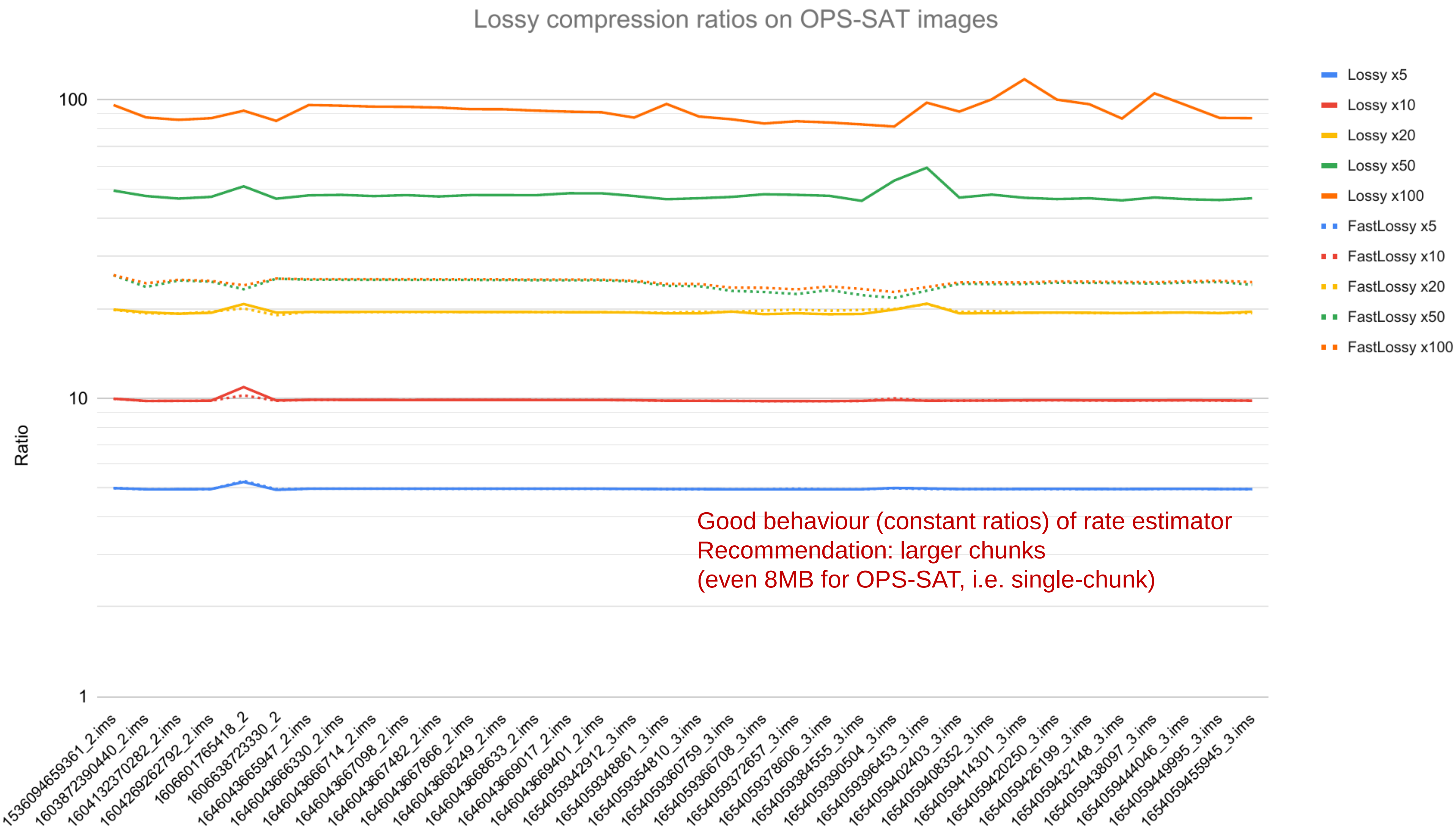
Near-lossless compression ratios



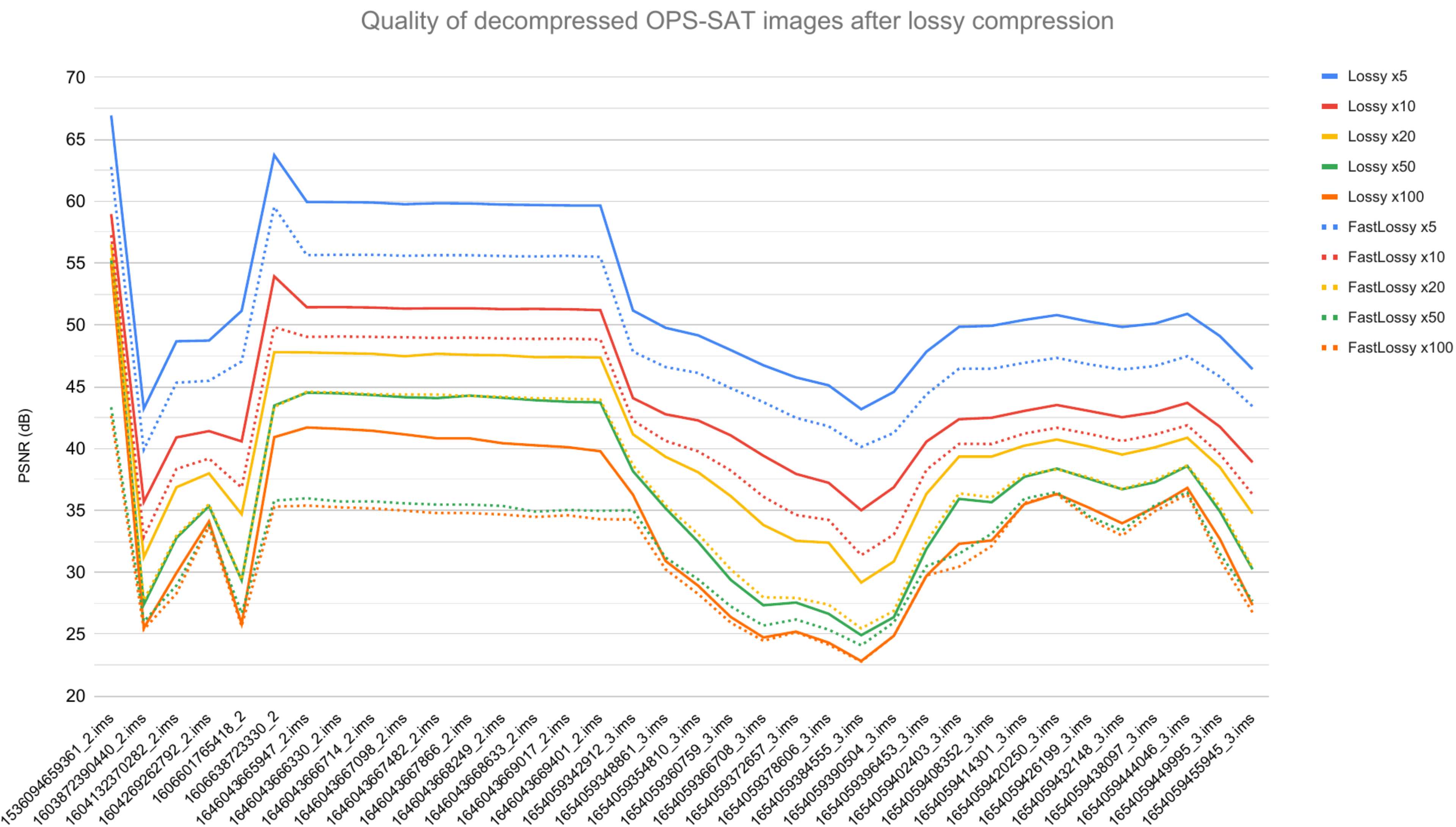
Near-lossless quality (PSNR)



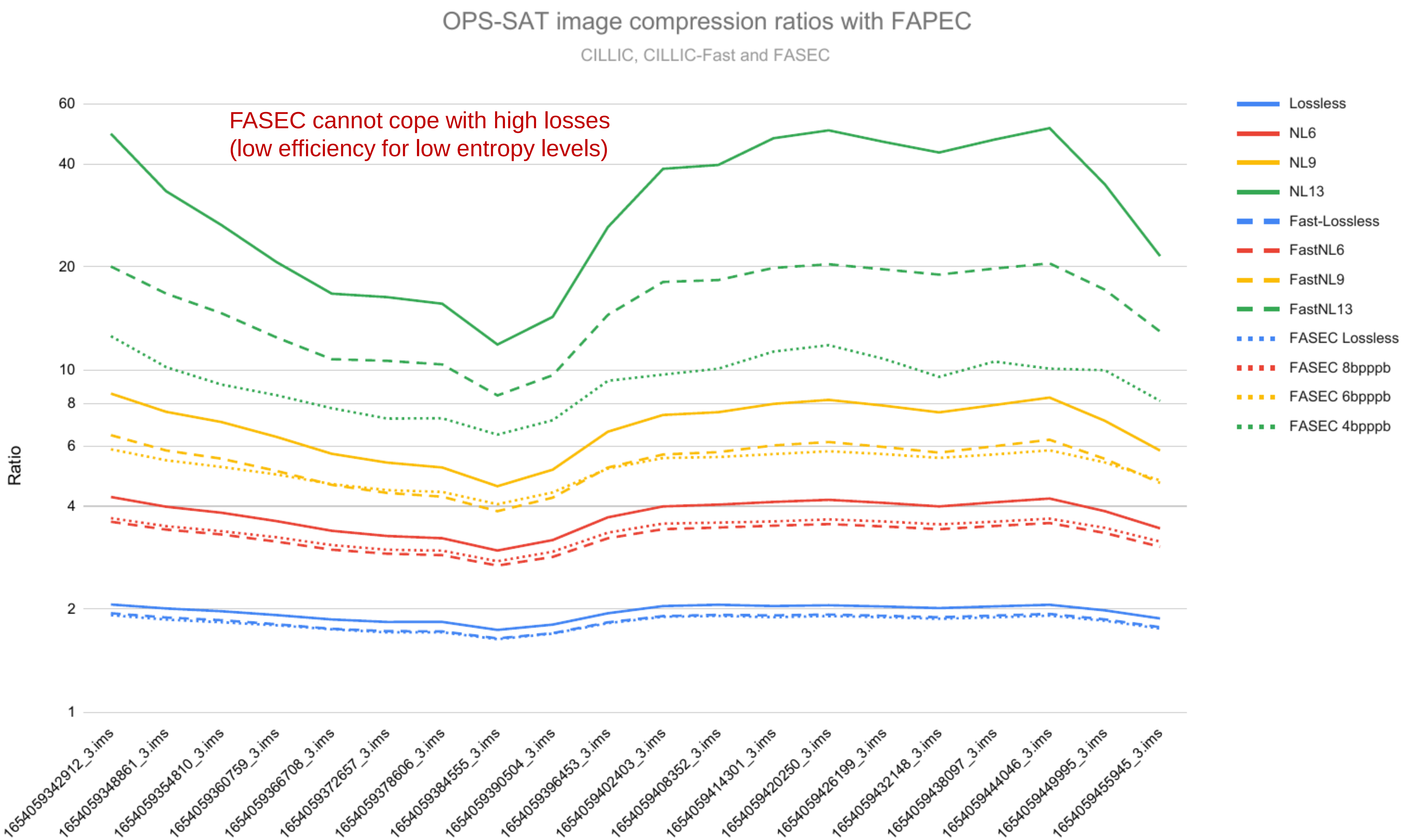
Lossy (fixed-rate) compression ratios



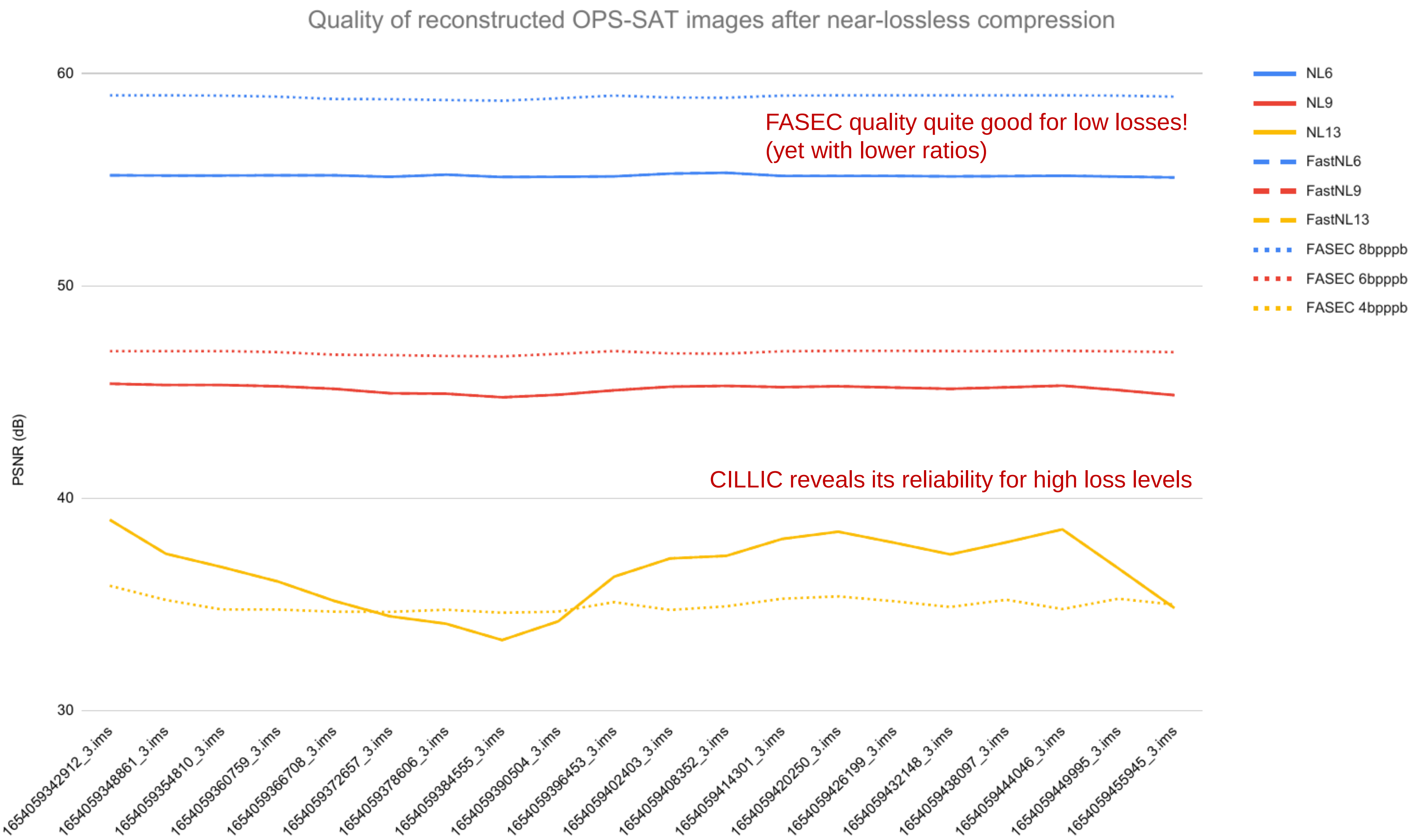
Lossy quality (PSNR)



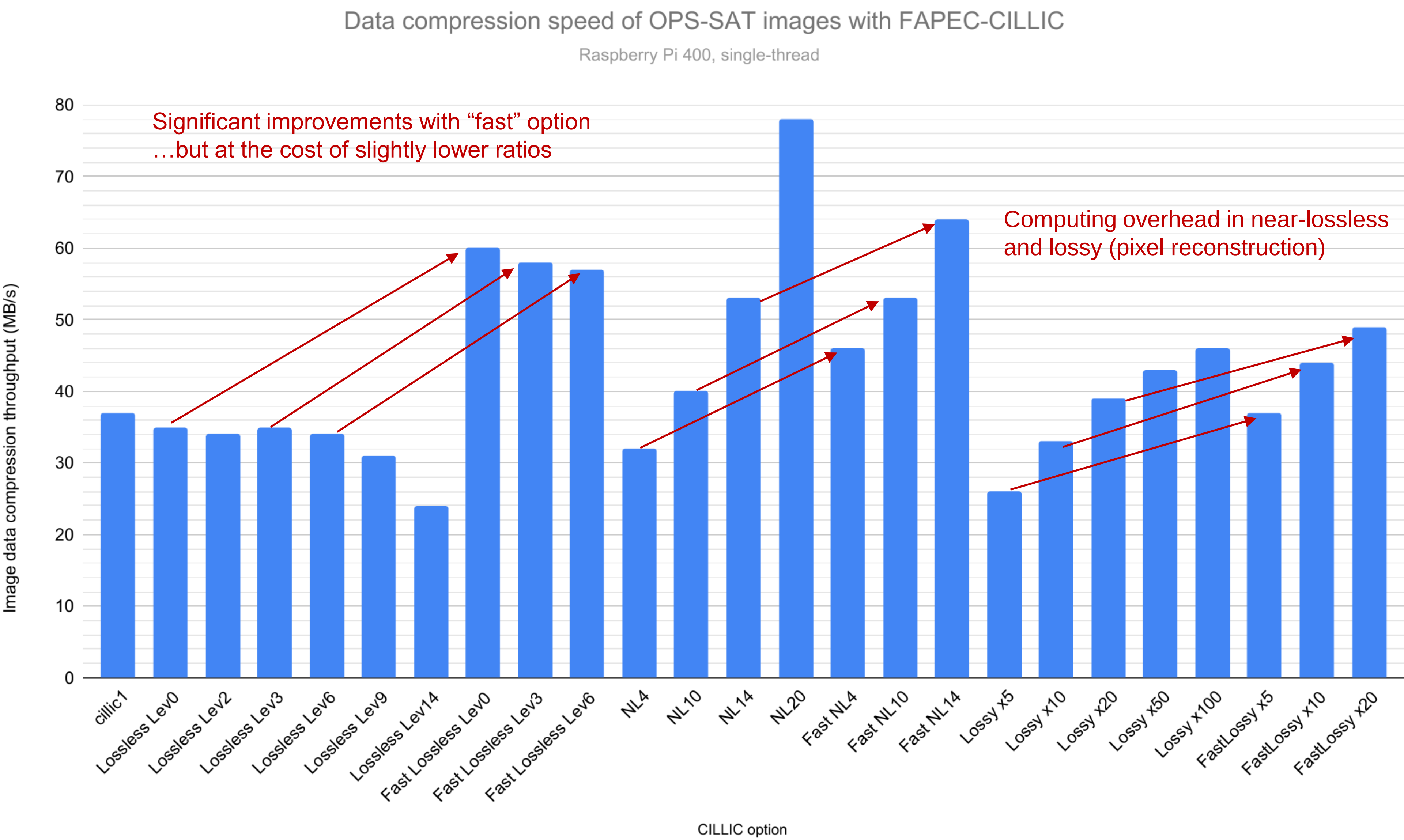
Compression ratios with fast options



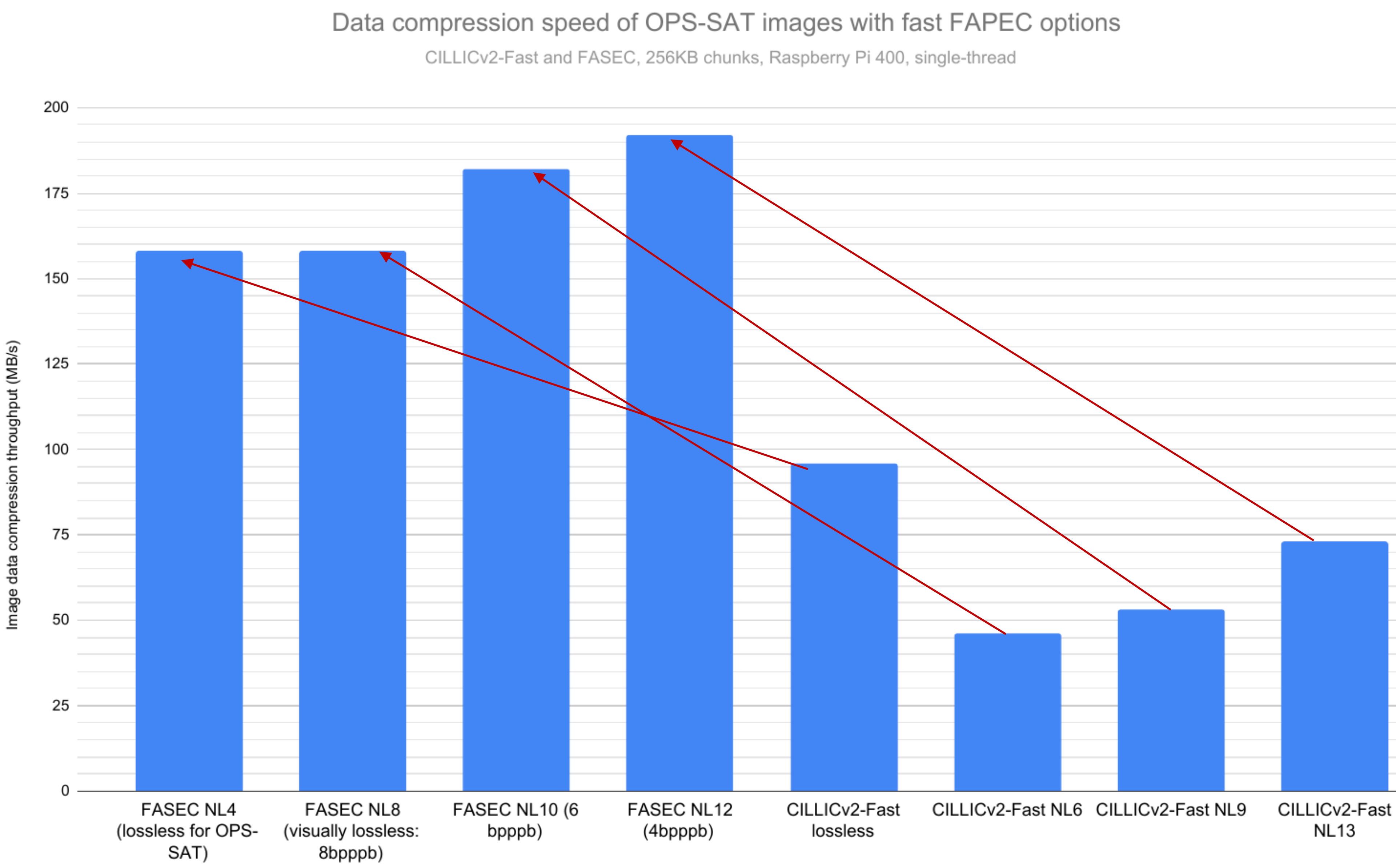
Quality levels with fast options



Compression speed with CILLIC



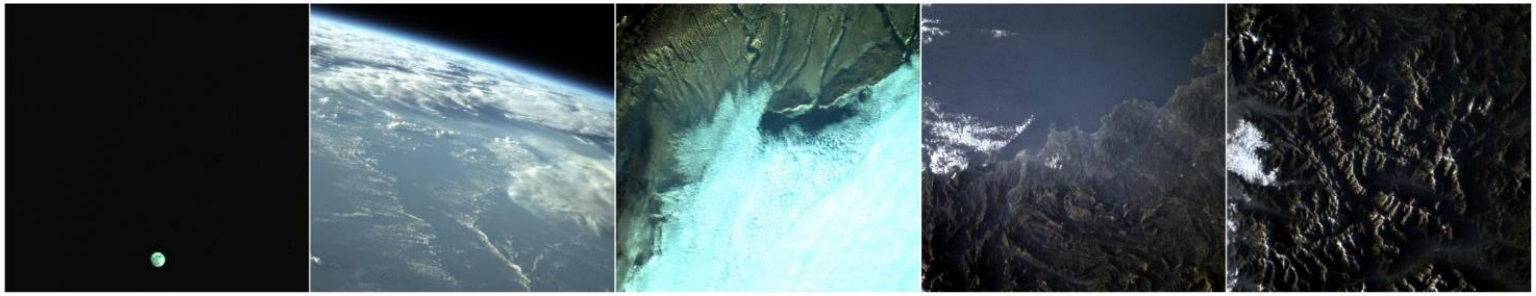
Compression speed with FASEC



Examples

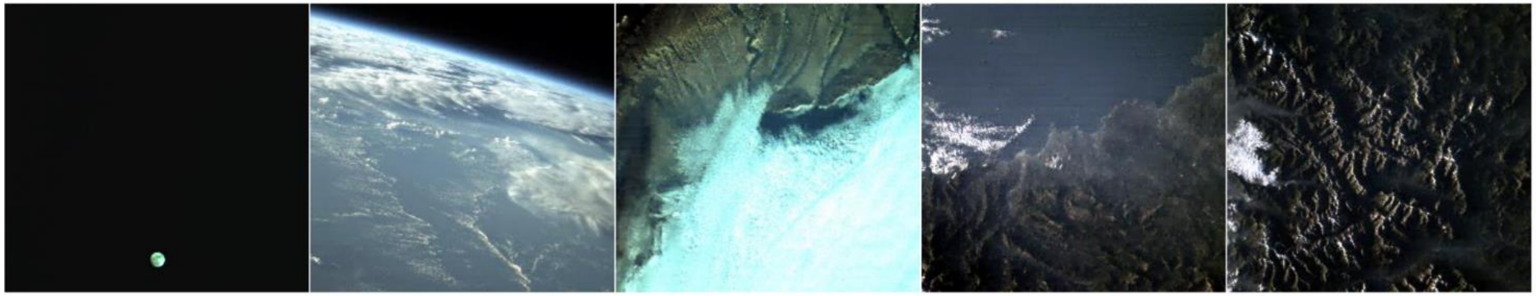
Some qualitative and quantitative results on a small subset of representative images:

CILLIC v2, lossless (ratios 3.07, 1.99, 1.98, 2.07 and 1.98, 45 MB/s single-thread on Raspberry Pi 400):



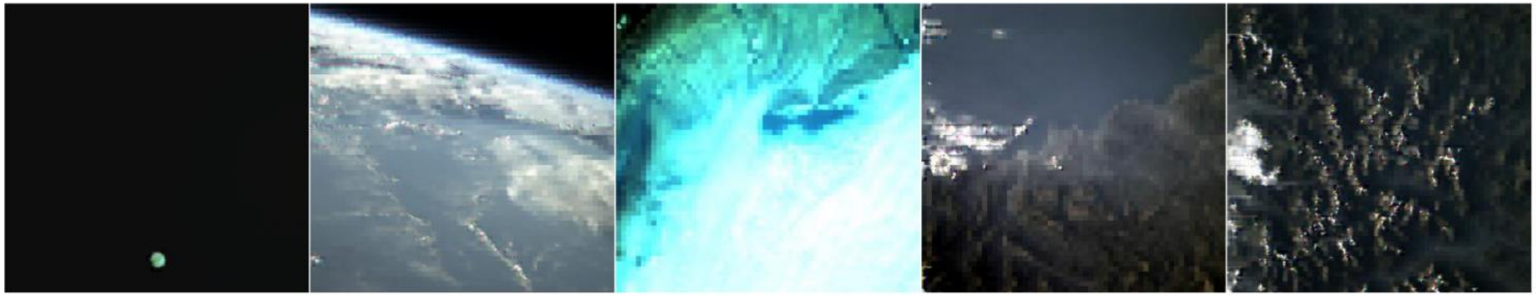
Max recomm. setting →

CILLIC v2, near-lossless level 14 (ratios 185.5, 40.0, 53.5, 68.4 and 38.9, 58 MB/s):



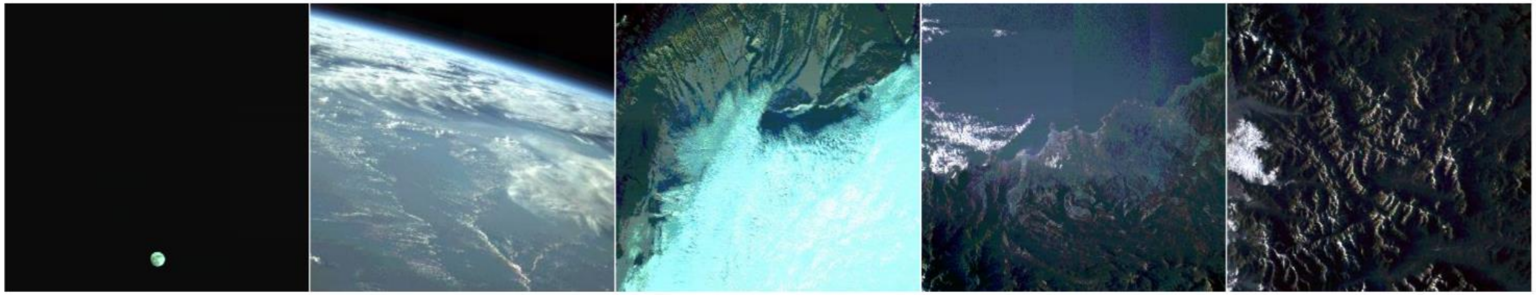
Just for thumbnails →

CILLIC v2, near-lossless level 20 (ratios 193.1, 133.4, 150.2, 148.8 and 125.9, 80 MB/s):

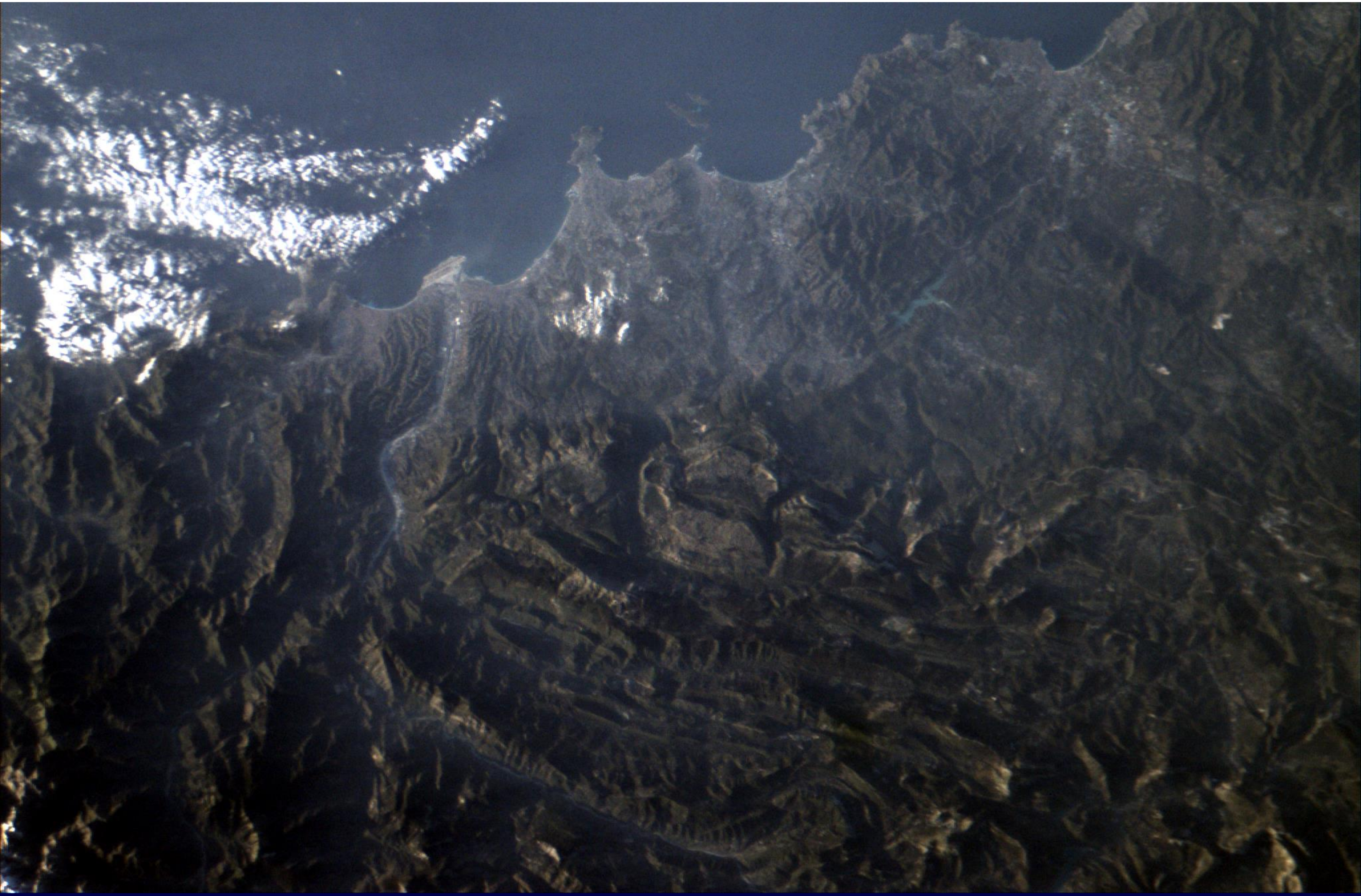


Much faster, modest quality →

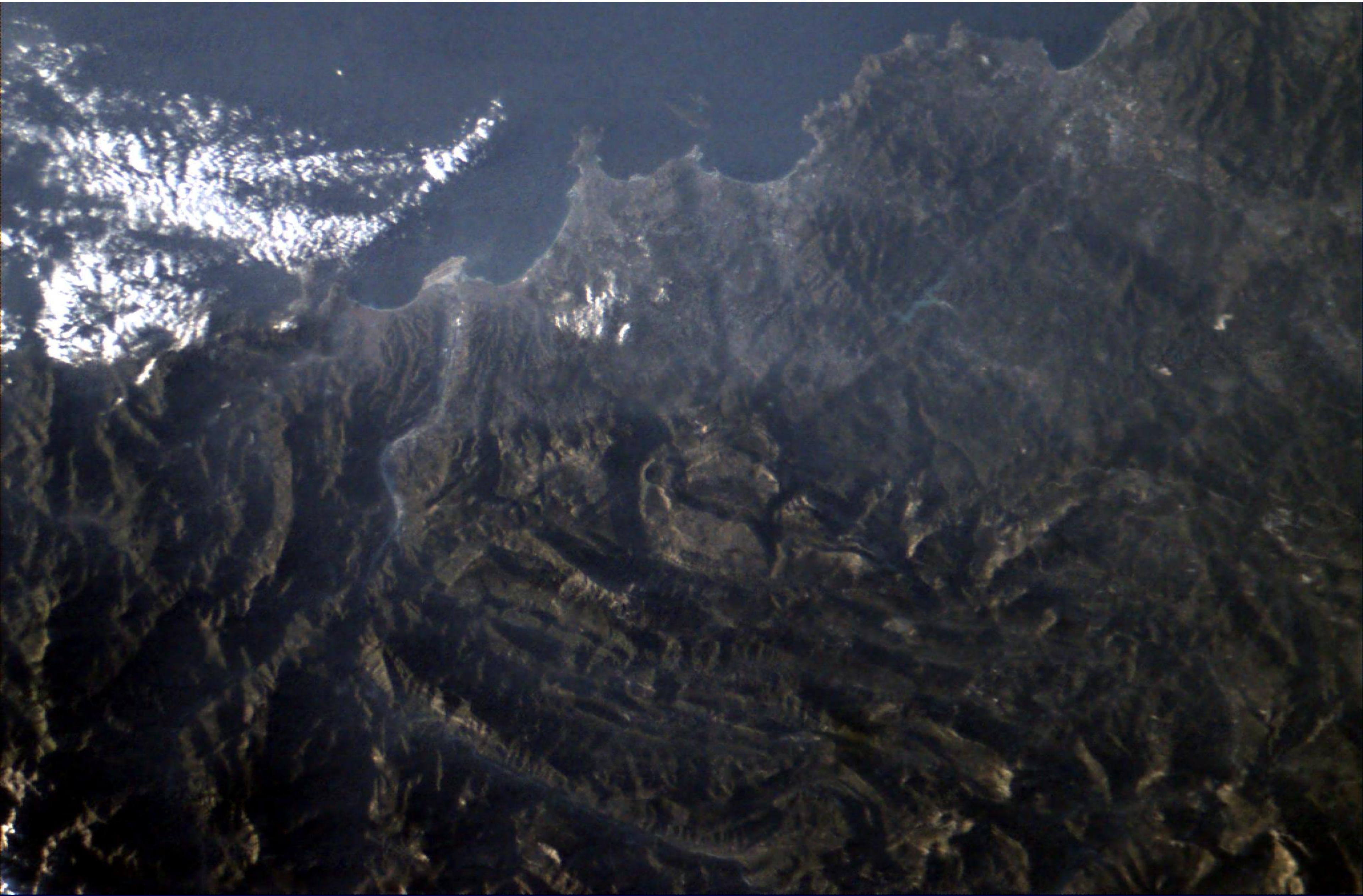
FASEC, near-lossless level 12 (ratios 38.5, 9.8, 9.6, 12.6 and 9.1, 180 MB/s):



Specific example: Lossless (ratio 2.06)



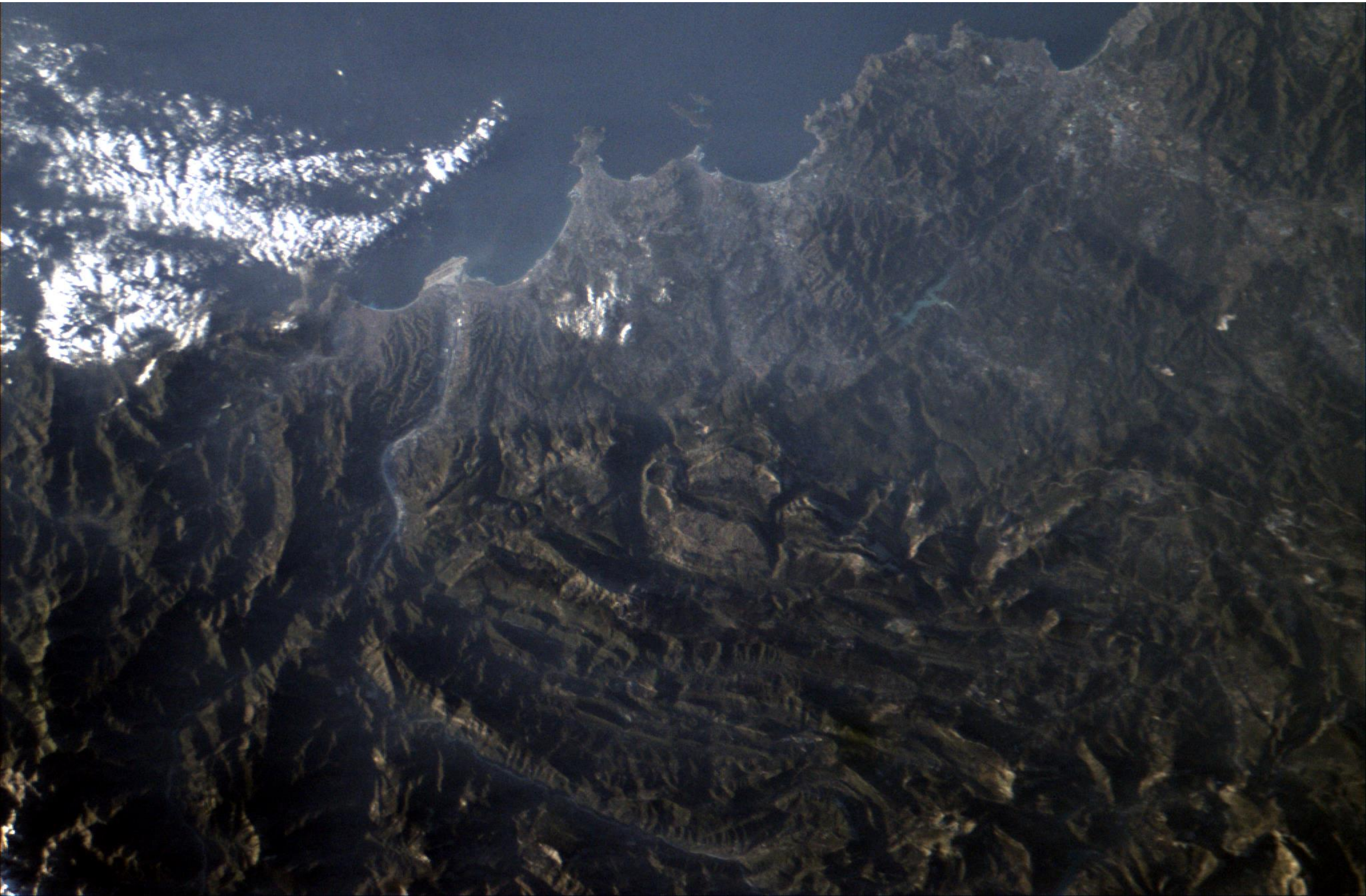
Specific example: NL10 (ratio 13.5)



Specific example: NL14 (ratio 68.4)

Specific example: NL20 (ratio 148)

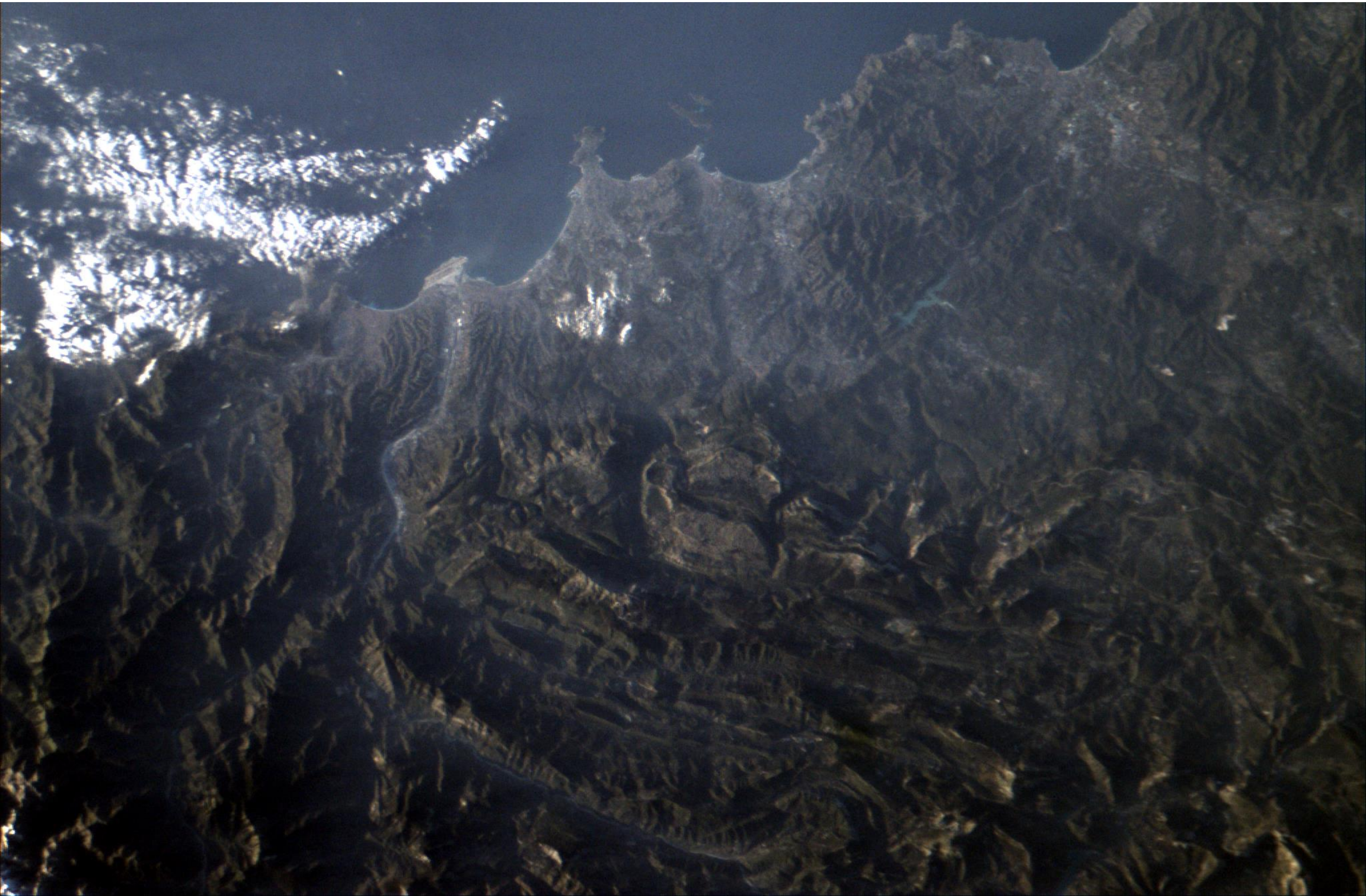
Specific example: Lossless (ratio 2.06)



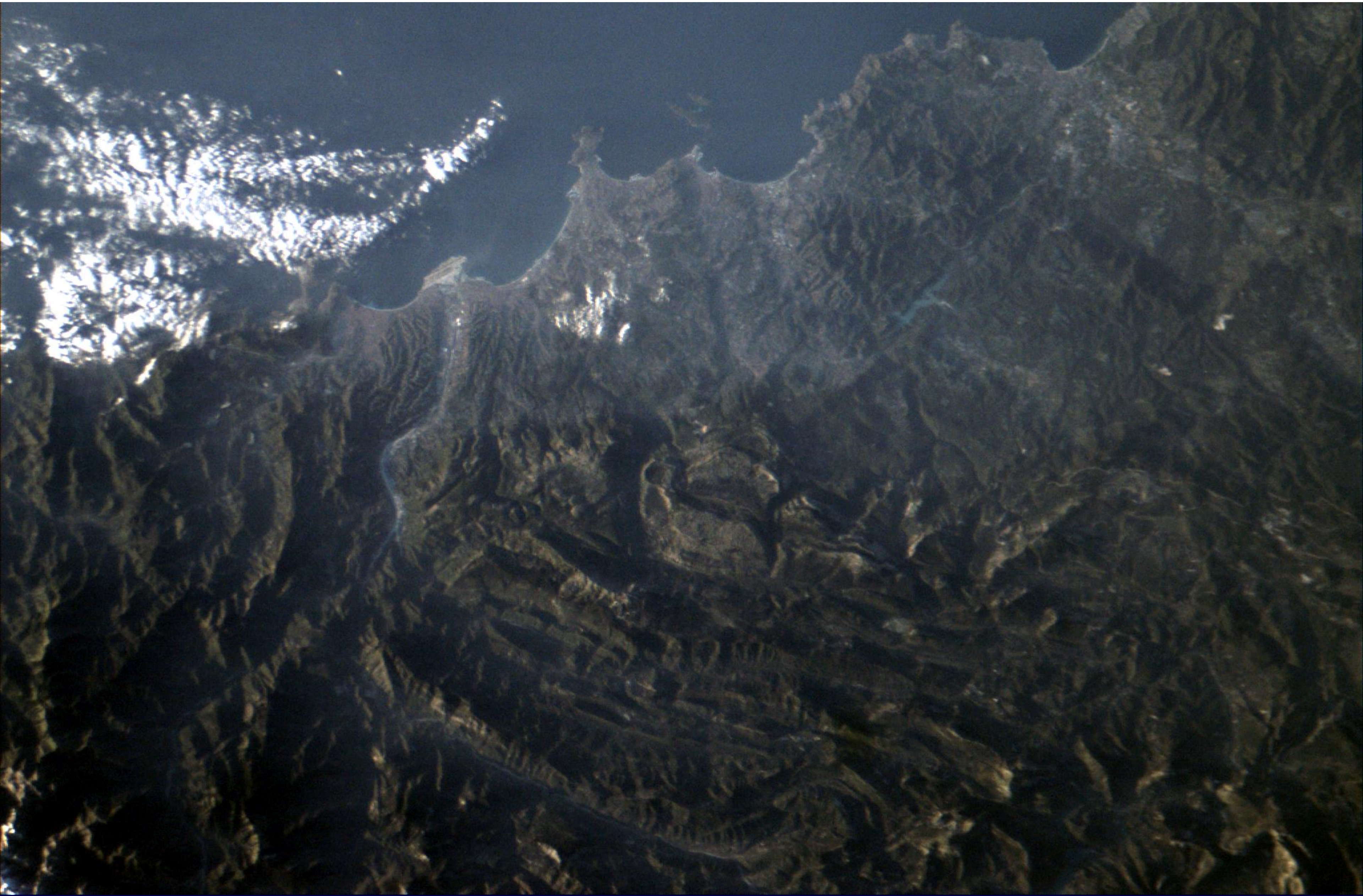
Specific example: Lossy x20 (41 dB)

Specific example: Lossy x50 (38 dB)

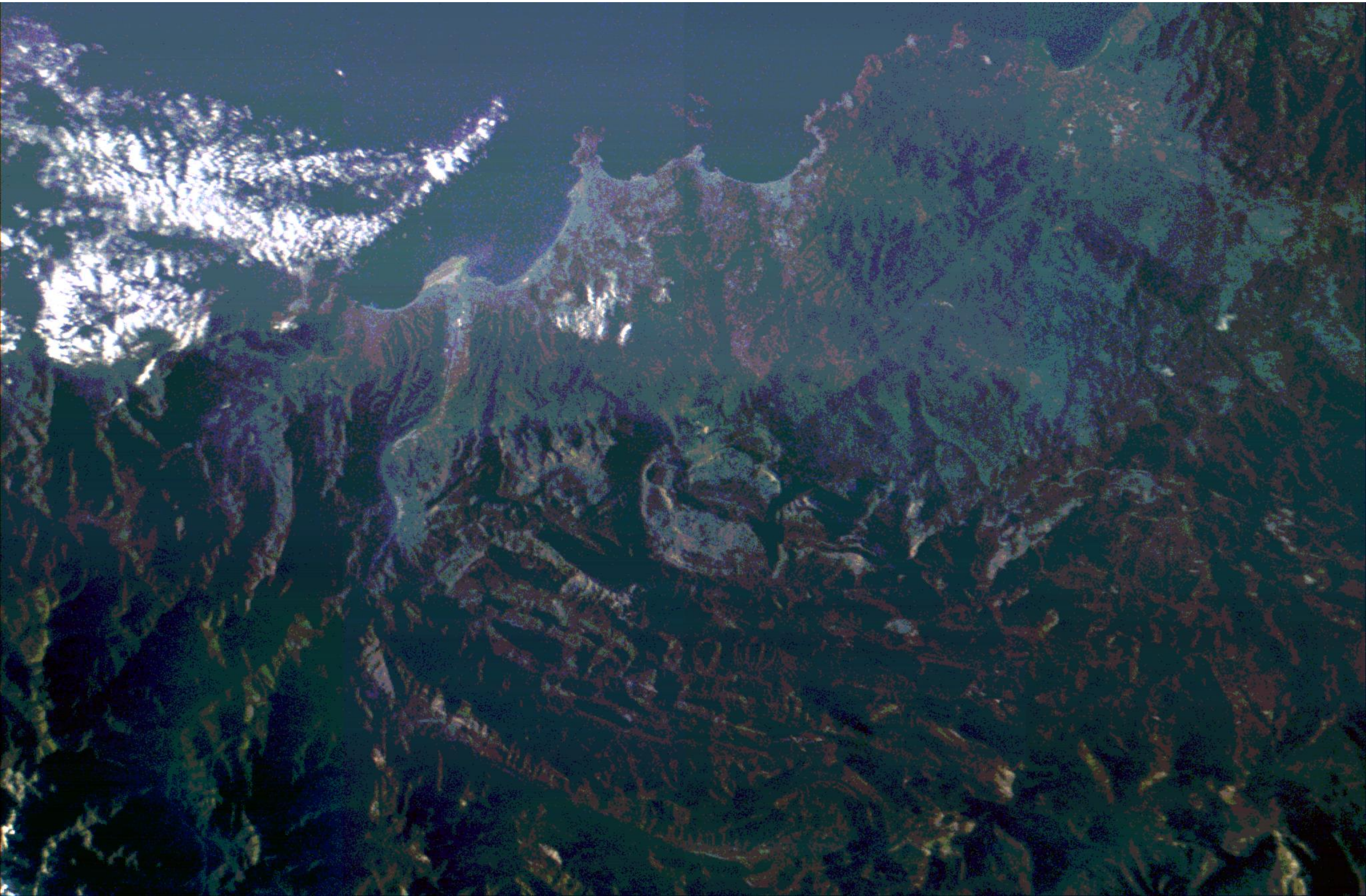
Specific example: Lossless (ratio 2.06)



Specific example: FASEC NL10 (ratio 5.8)

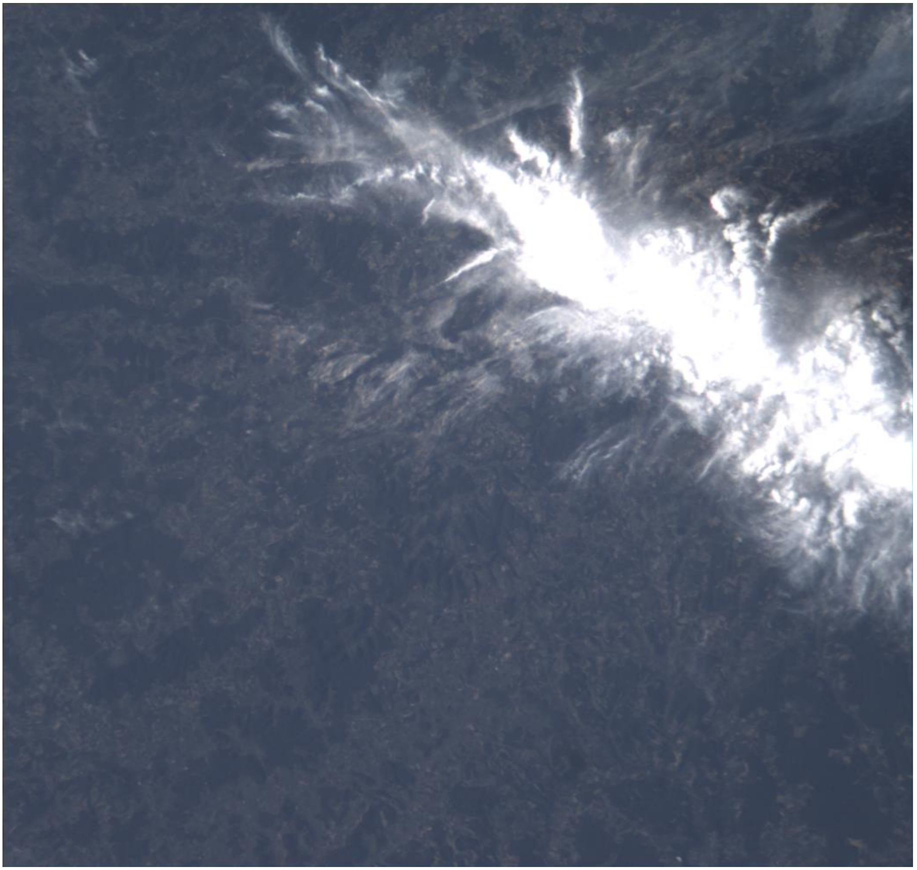


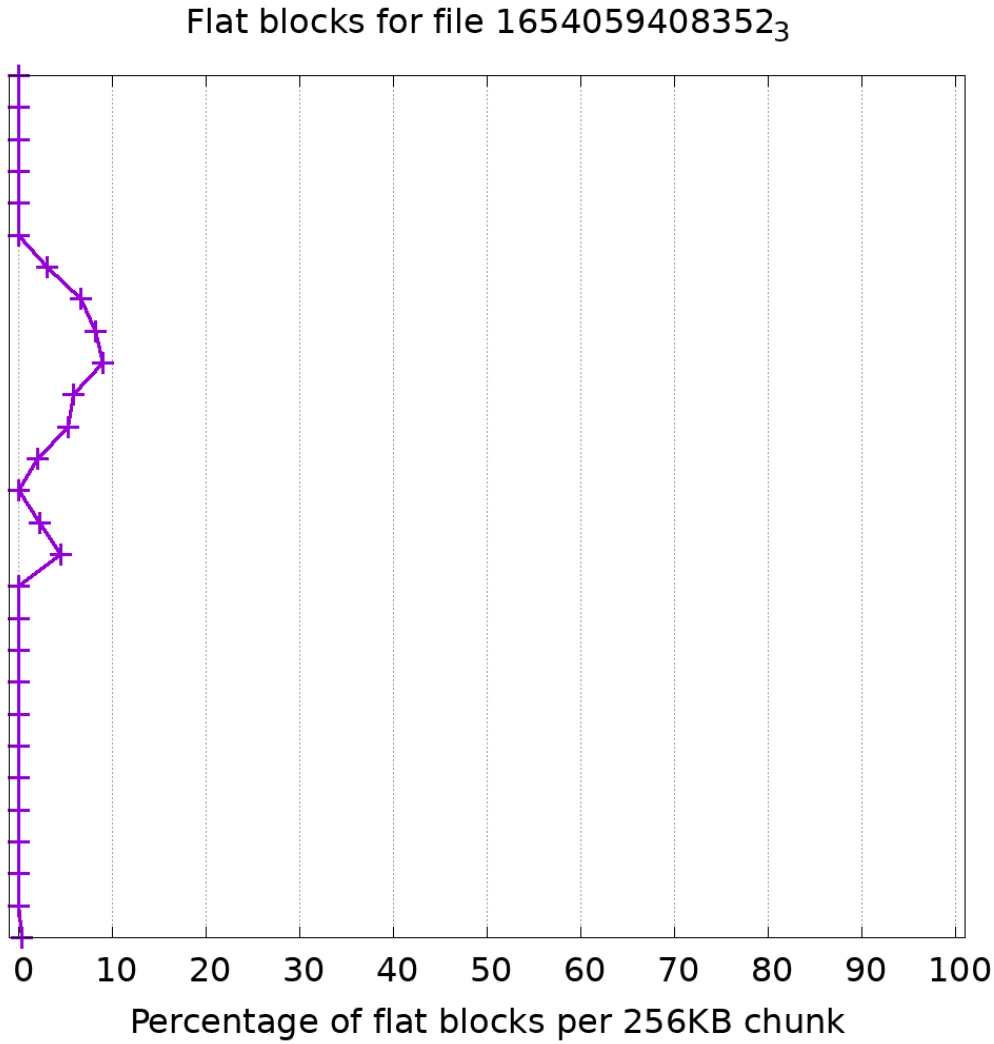
Specific example: FASEC NL12 (ratio 12.5)



Basic data analysis capabilities: flat blocks

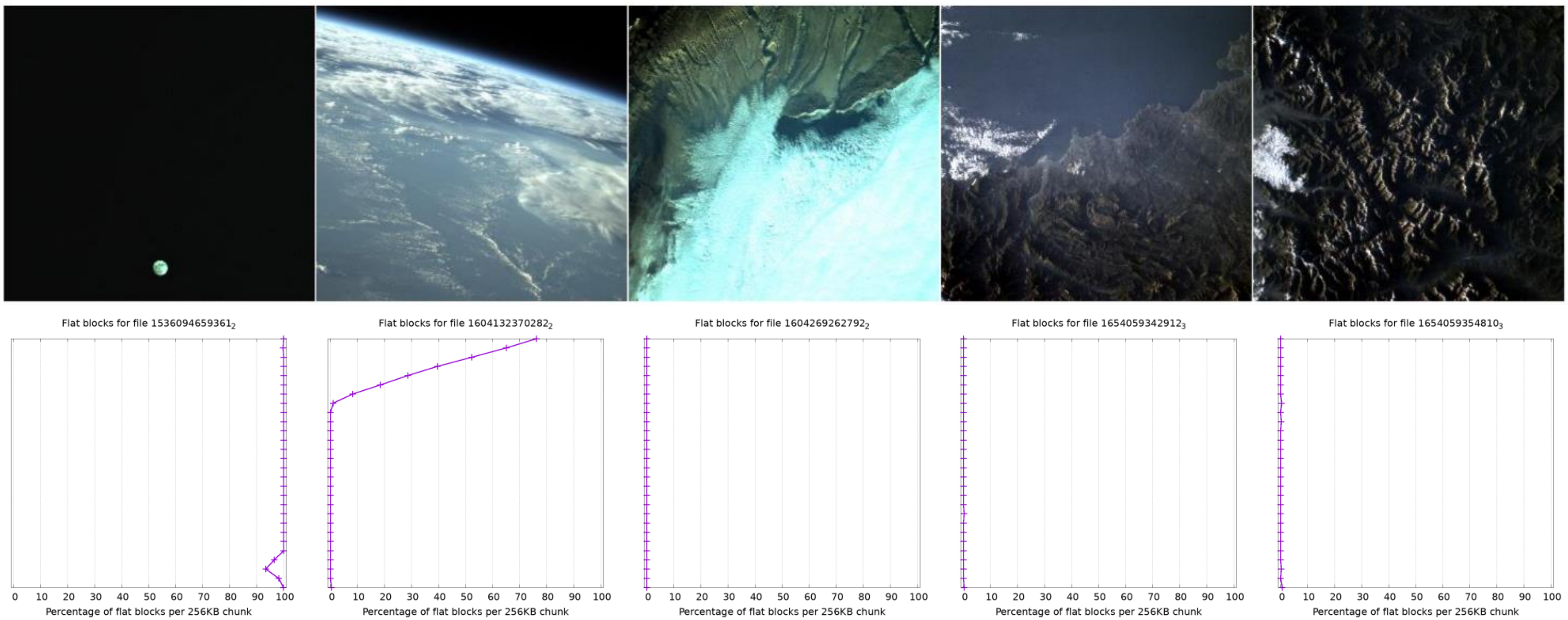
- Small text file with the percentage of "flat blocks" found per chunk
  - FAPEC is fast → we can do an extra run with small chunks to generate this info (3.3 KB per file)





Basic data analysis capabilities: flat blocks

- Flat blocks info can enable “smart downlink” capabilities
  - 100% flat blocks for all chunks? → not worth downloading it (space, ocean?)
  - 100% except for a small section? → potentially very interesting: Moon, another satellite?, space debris??, etc.
  - Gradient in the flat blocks percentage? → Earth limb
  - Local increase with significant variability? → clouds, lakes



CILLIC: motion estimation

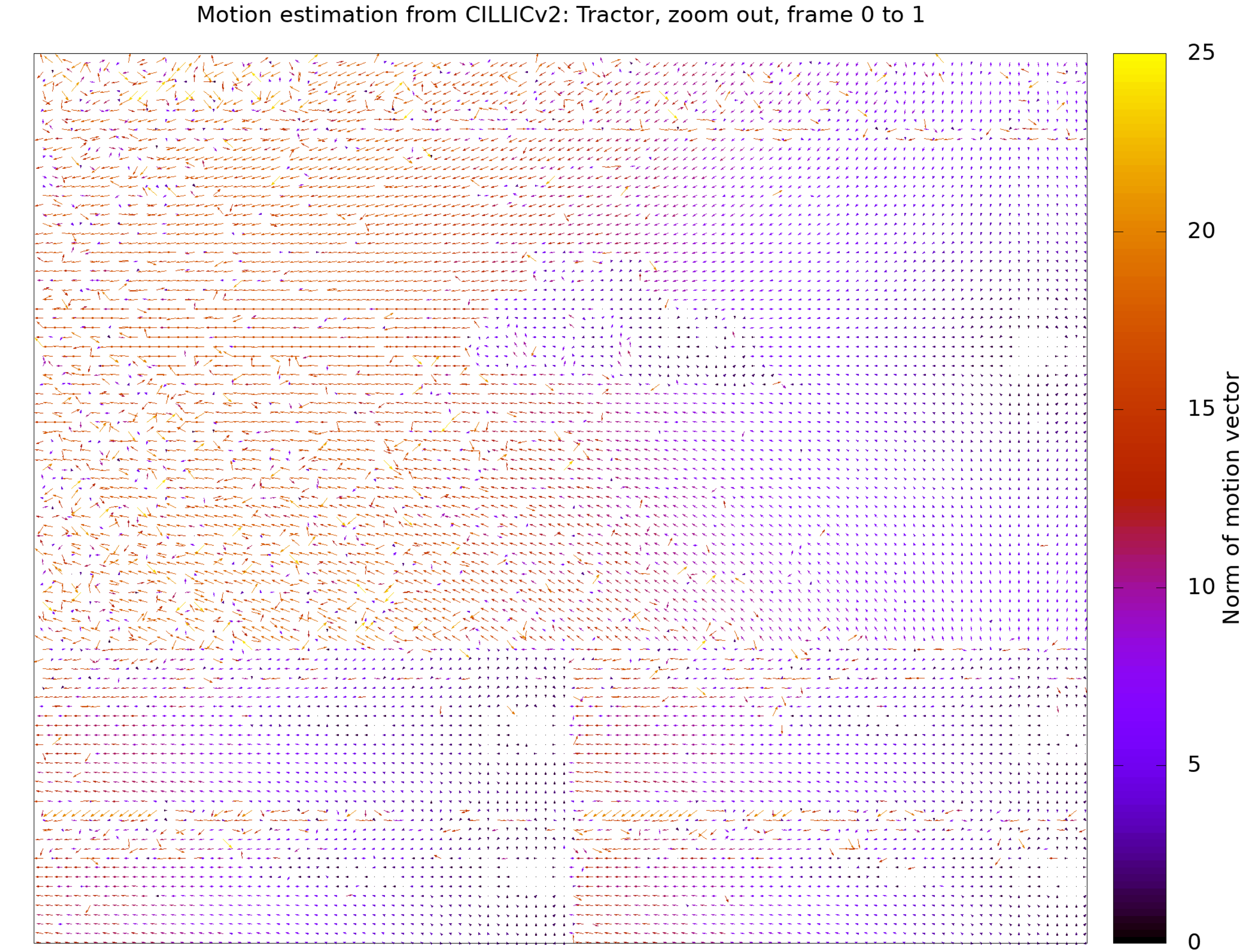
CILLIC: motion estimation

CILLIC: motion estimation

CILLIC: motion estimation

CILLIC: motion estimation

CILLIC: motion estimation



Outline

- Overview of the activity

- Image data compression
- Radio-frequency data compression

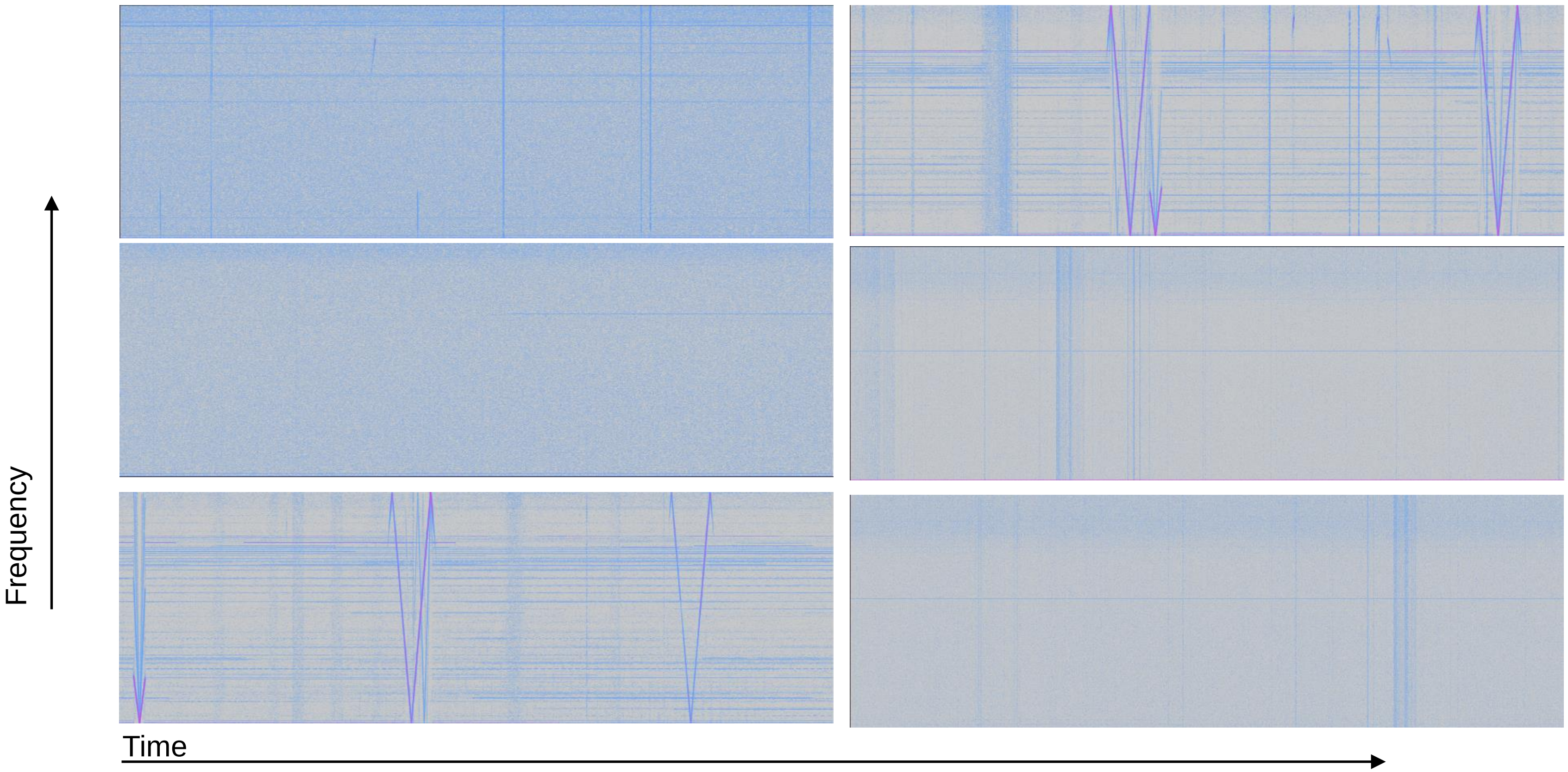
- Conclusions

Improvements in the Wave algorithm

- Same approach as in first version
  - Linear Predictive Coding (LPC) + Levinson-Durbin recursion for coefficients determination
  - Excellent compromise between ratios and speed
  - Up to 32K channels, periods of up to 8M samples, lossless + near-lossless options
- Minor improvements in lossless operation
  - Higher LPC order: Up to 16
  - Adaptive LPC order for each period of samples
  - Larger FAPEC coding blocks
- Remarkably: "**smart lossy**" algorithm
  - Detect presence of signals in the RF data files
  - **Automatically set the loss level for each period**
  - Simplistic (fast) option using information from Levinson-Durbin recursion,
    then adjusting loss level (LPC residuals quantization) according to estimated signal/noise levels
  - Rigorous (slower) approach:
    Welch method + Akaike Information Criterion, to estimate noise power;
    Neyman-Pearson detector with different probability levels of false alarm, for signal detection
    → **Published as the Spectra library in GitHub**

OPS-SAT RF dataset

- UHF (433 MHz) and “GNSS” (1.575-1.602 GHz) files
  - 3 Msamples/s, 0.75 MHz bandwidth, 60-66 dB gain
  - Very short files: 0.5 to 2 seconds (plus a few of just 0.7 ms)
- Visualization through spectrograms (using Audacity software)
  - Some very interesting features (very high Doppler): parasitic signals from satellites in other orbits?



Smart lossy algorithm

- Estimate “signal” and “noise” (unmodelled part)
  - Signal = energy successfully modelled by means of the LPCs
  - Noise = error from Levinson-Durbin recursion (i.e, unmodelled energy)
  - Excellent correlation with actual features seen in the spectrogram (see hereafter)

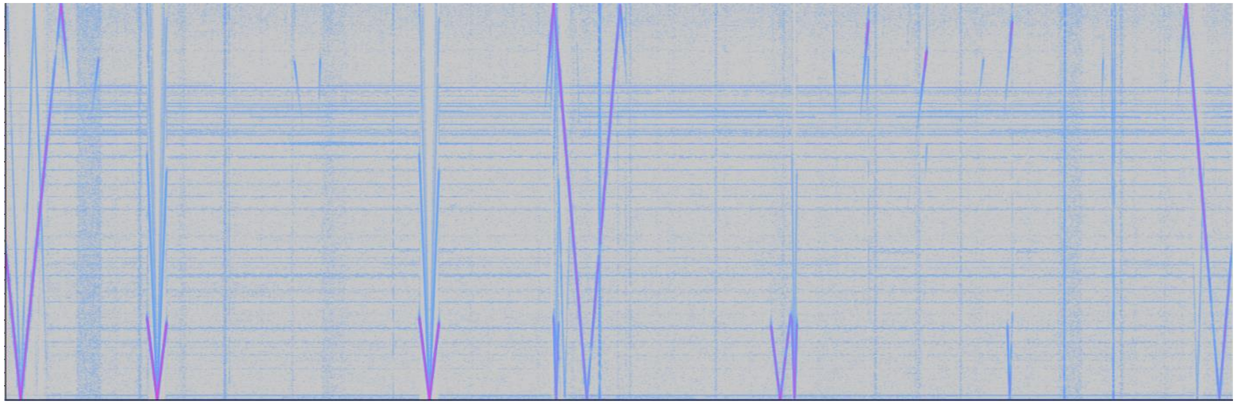
- Algorithm adjustments
  - Maximum loss
  - SNR threshold (using the “signal” and “noise” meaning above)
  - Target bits for periods with signal (above SNR threshold)
  - Target bits for periods with only noise (below SNR threshold)

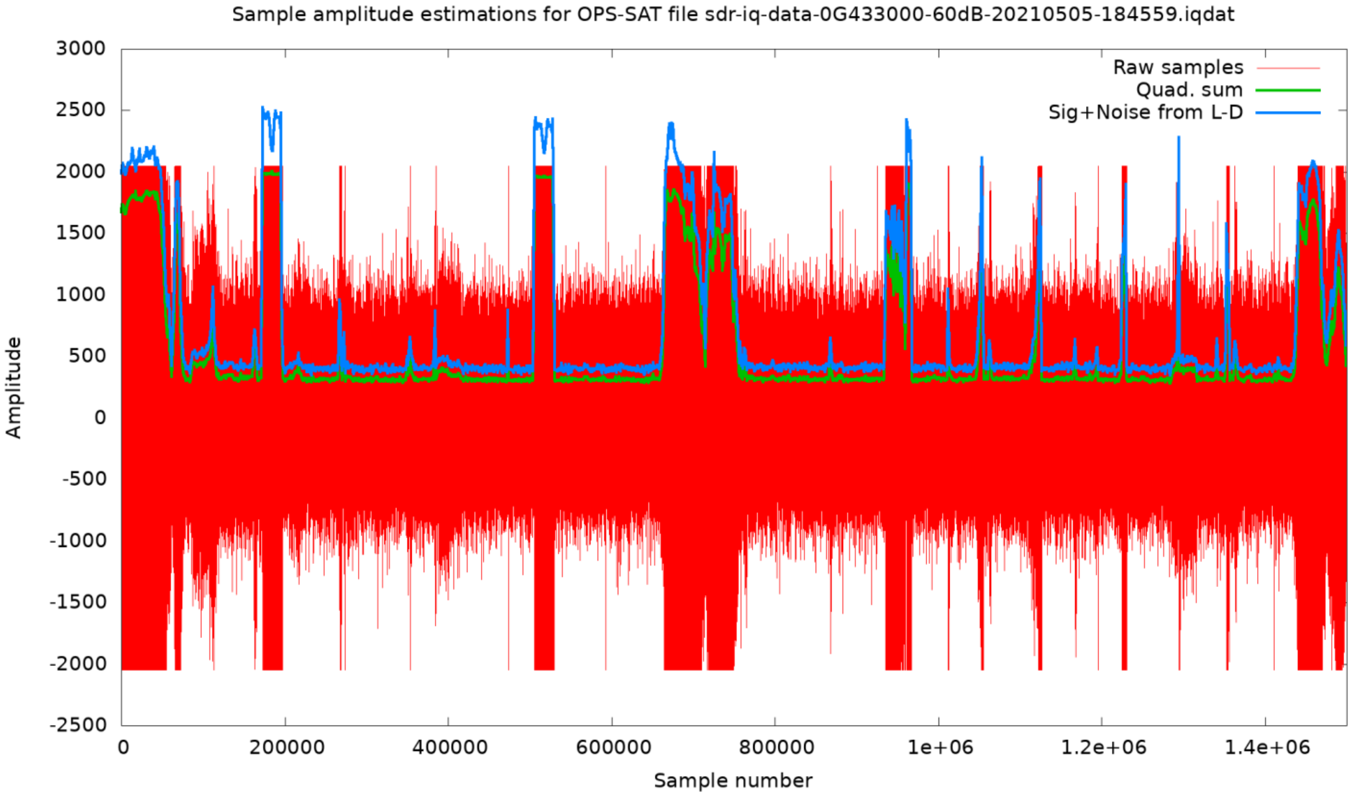
→ Versatility, allowing to focus on, e.g.:

- clear high-SNR signals, applying losses when not detected
- apply more losses when loud parasitic signals appear on top of our target faint signal

Smart lossy algorithm

UHF data file
(433 MHz, 0.5s)



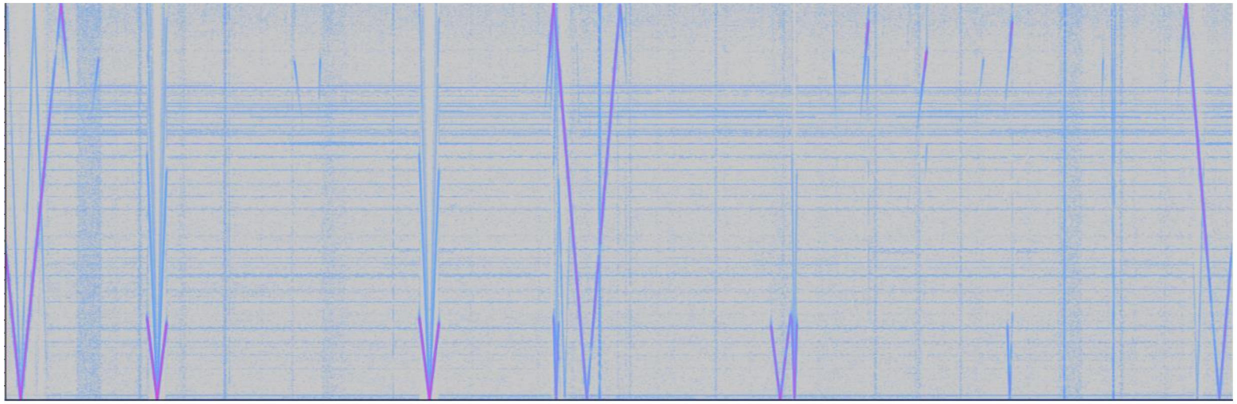


Quadratic sum of all samples in a period shown as reference

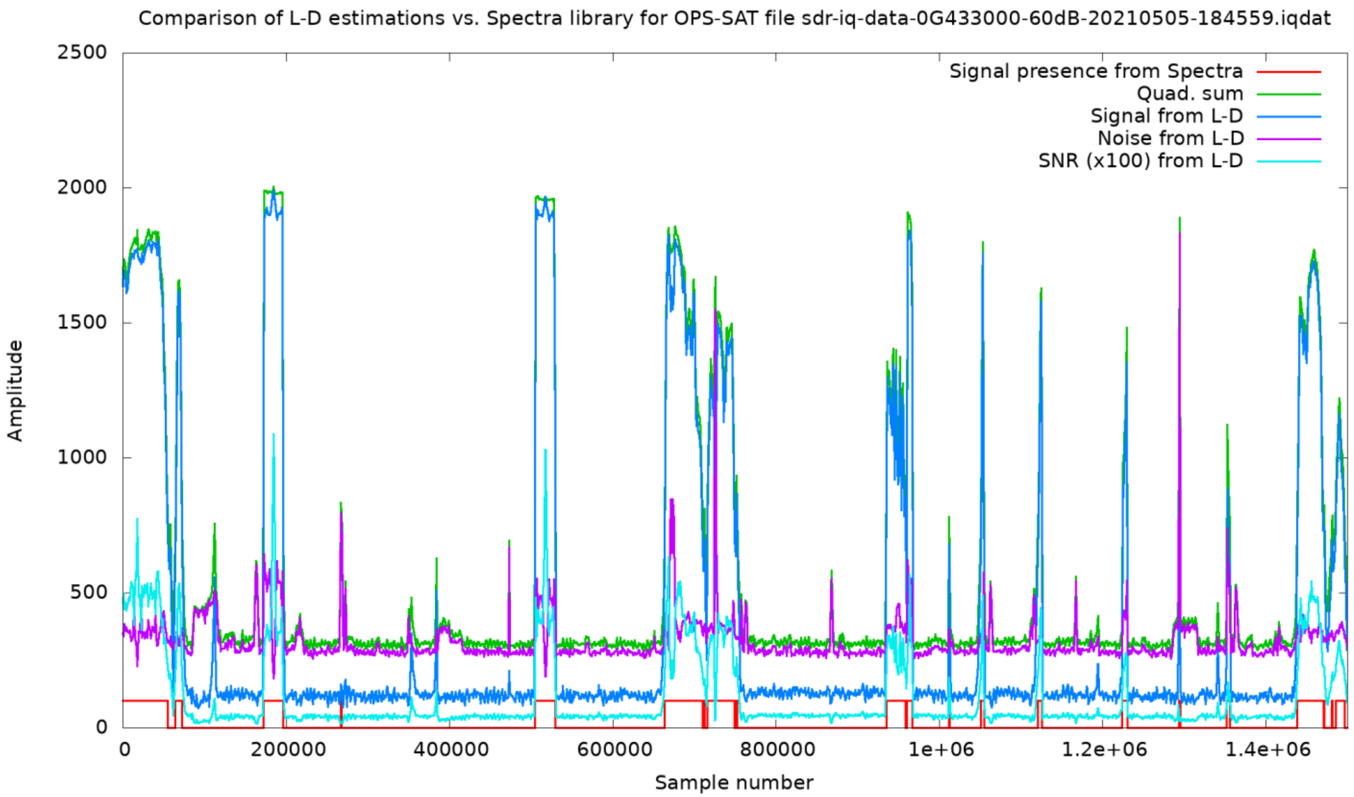
Levinson-Durbin estimations of signal + noise quite consistent with quad. sum (slightly over-estimated)

Smart lossy algorithm

UHF data file
(433 MHz, 0.5s)



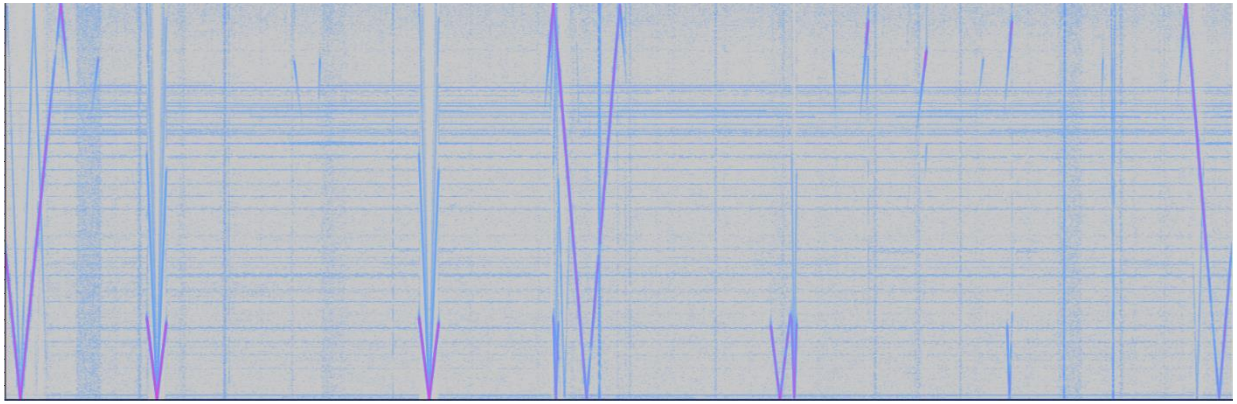
Output from our "Spectra" library taken as reference: The "SNR" from the faster Levinson-Durbin approach raise consistently with the signal presence detection



Smart lossy algorithm

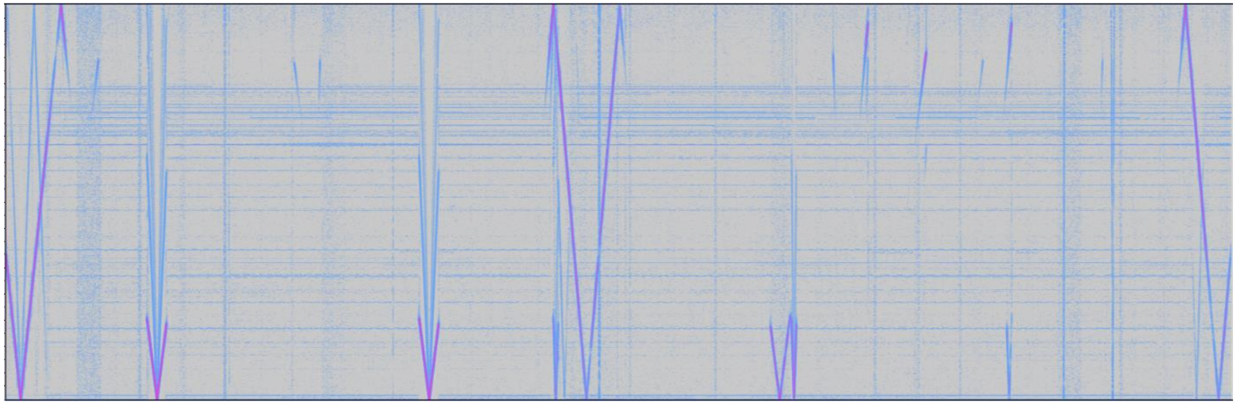
UHF data file
(433 MHz, 0.5s)

Lossless ratio: **1.5**



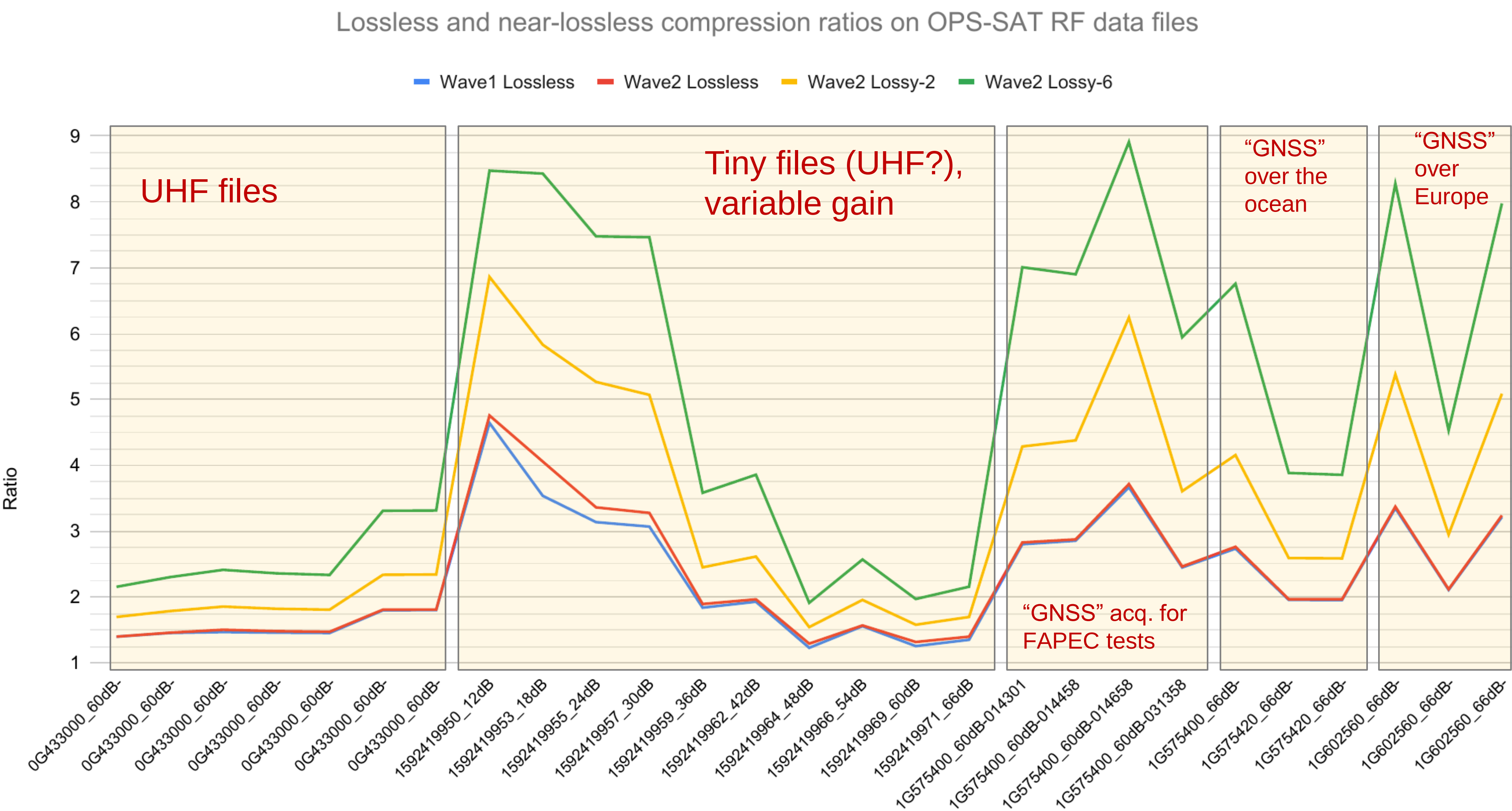
Spectrogram after
smart lossy ("Opt.5")

Ratio: **3.3**



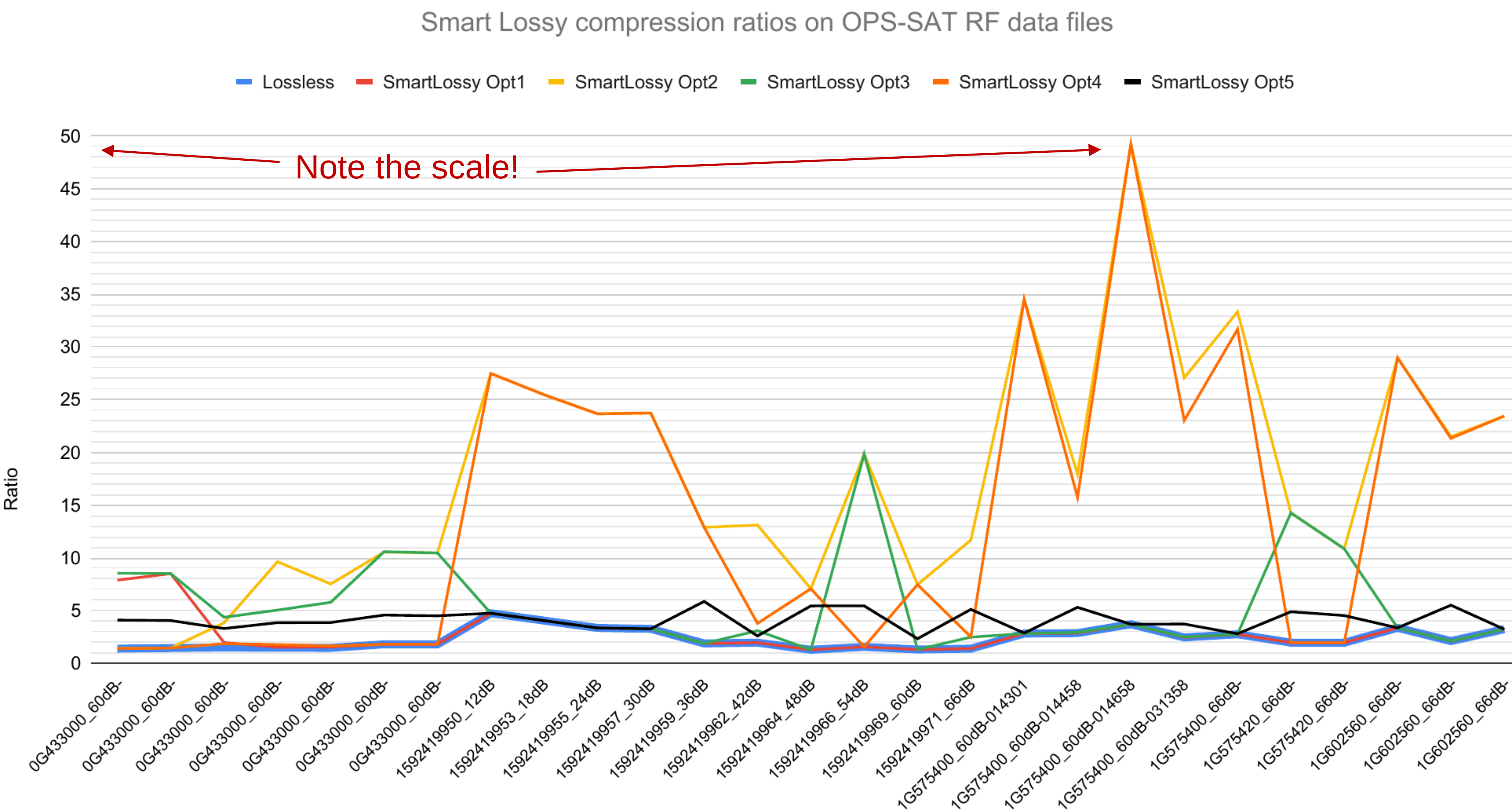
- Some smart lossy options tested:
  - Options 1 and 3: high quality on the signal, aggressive loss on noise, low+medium SNR threshold (0.4+1.1)
  - Options 2 and 4: high quality on the noise, aggressive loss on signal, low+medium SNR threshold (0.4+1.1)
  - Option 5: good quality on the signal, not so aggressive on noise, high SNR threshold (2.0)
  - Option 5 taken as a reasonable reference

Lossless & near-lossless compression ratios



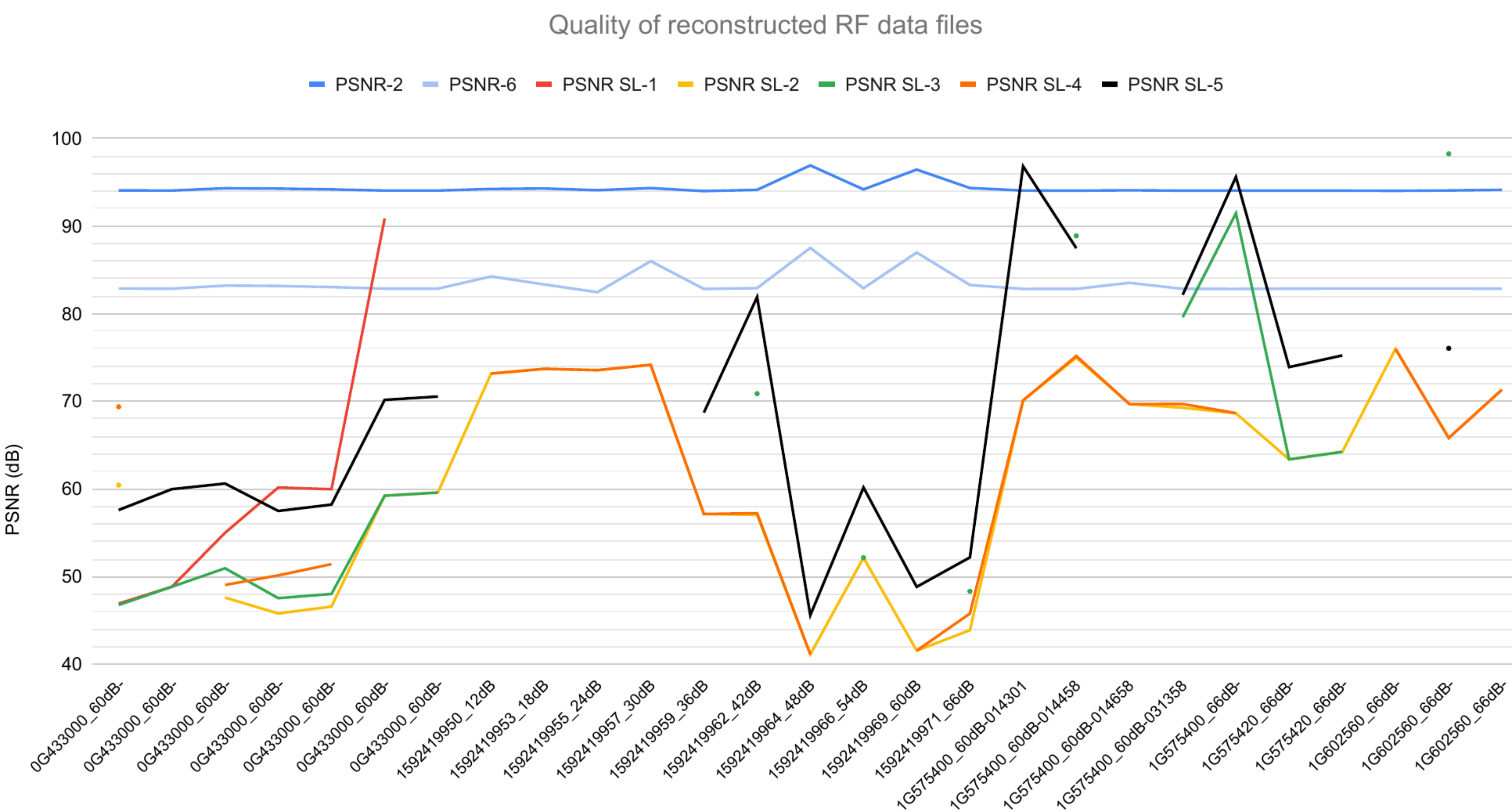
- GNSS-like files better compressed (no clear signal therein); very high improvement in near-lossless (but still useful afterwards?)
- UHF gets quite low ratios (several evident signals therein)

Smart lossy compression ratios



- Black line (Option 5): significant improvement on all UHF files, also good on some of the GNSS files
- Yellow+orange (focus on noise): very high ratios on most GNSS files… where Lev-Durb approach gives good SNR

Near-lossless & smart lossy PSNR



Missing results correspond to “infinite” PSNR (i.e., lossless selected for the whole file)

Additional tests with Wave: GNSS signals

- One of the challenges is **GNSS data compression**
  - Spread spectrum → signal looks like noise → may not be detected by our "smart lossy" algorithms
  - Can we compress SDR data files with GNSS signals **with losses**?
- Preliminary evaluation of GNSS signal detection with **GNSS-SDR** software
  - Quite difficult (very complete!) software package, specially regarding its configuration
  - Some tests on OPS-SAT SDR data files: GPS signals detected, but not really conclusive (perhaps problematic due to high Doppler?)
  - Took other (ground-based) SDR data files with the same format: **Galileo** signals persistently detected
  - Tested FAPEC-Wave with near-lossless compression:
    Lossless → ratio 1.8
    Near-lossless with quite high loss → ratio 7, Galileo signals still consistently detected!

→ We can configure the smart-lossy approach in a conservative manner to ensure usability of GNSS signals while still achieving high ratios

Outline

- Overview of the activity
- Image data compression
- Radio-frequency data compression
- Conclusions

Conclusions

- Most of the tasks completed successfully
- Several difficulties found
  - Complexity of the problem, esp. on multi-band and video
  - Schedule, limited resources
- CILLIC:
  - Significant improvements in lossy compression (quality, high ratios)
  - Simplistic option (FASEC, non-CILLIC) for very high throughput
  - Embedded data analysis: detection of “flat blocks” (or “Regions Of Non-Interest”)
- Wave:
  - Nearly optimum solution for radio-frequency data compression
  - Lossless + near-lossless + “smart lossy”
  - Estimation of noise and signal levels and/or signal detection
  - “Spectra” library for accurate signal detection, published in GitHub
- “Smart downlink” enabling technology
  - Identification of (portions of) image and RF data files with useful/useless information
  - Optimization of downlink (beyond compression): avoid unnecessary downloads

    → **Continuous optical/RF monitoring, download of just interesting files**

DAPCOM Data services

ops-sat

esa DISCOVERY

Versatile data compression software for sustained high-throughput in-orbit data acquisition

(a.k.a. **RICSDAC**: RF and Image Compression Software for Demanding Applications in Cubesats)

Final Review

Jordi Portell

on behalf of the FAPEC team at DAPCOM Data Services S.L.: Aniol Martí, Eloi Saus, Riccardo Iudica