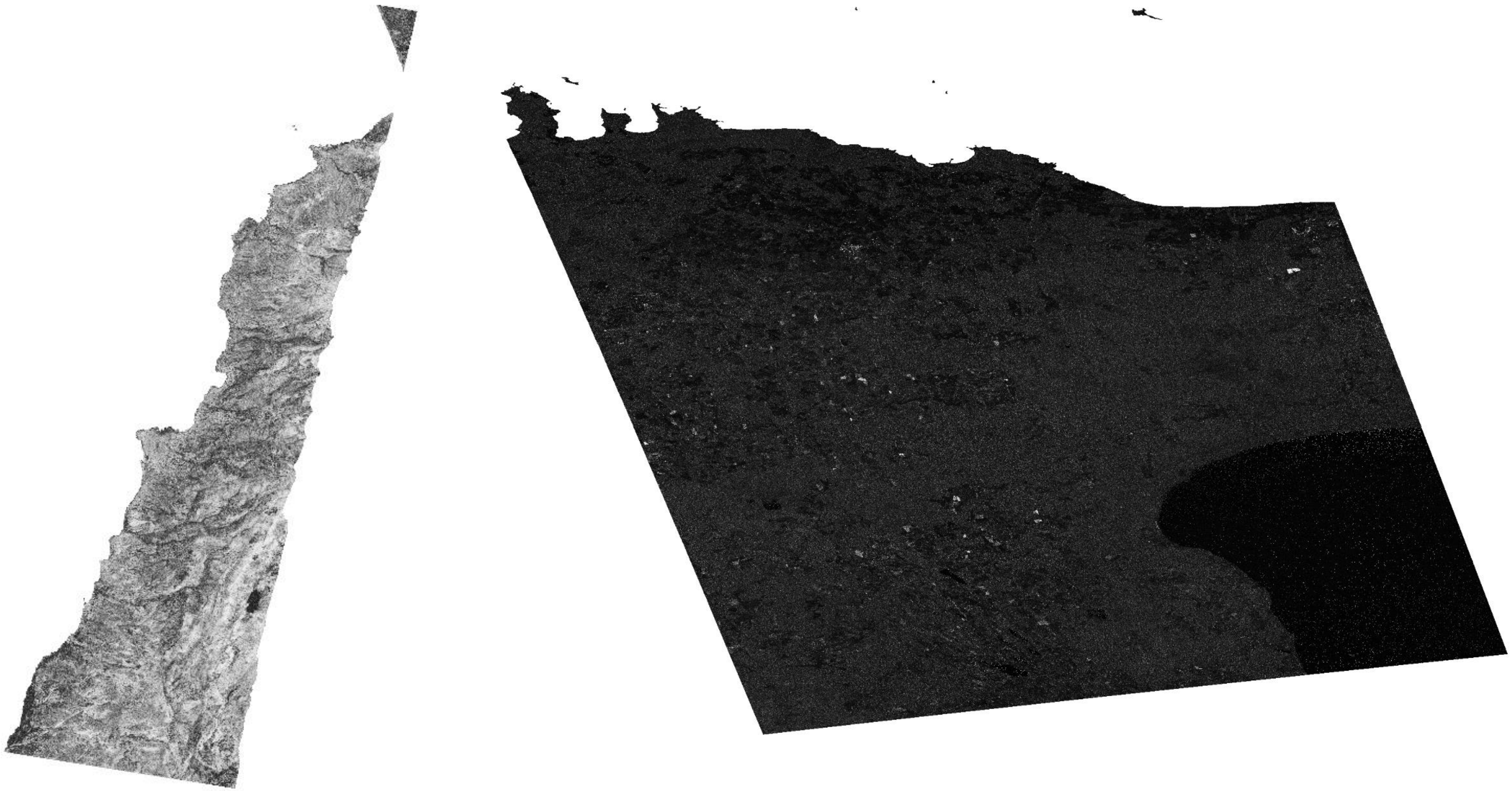
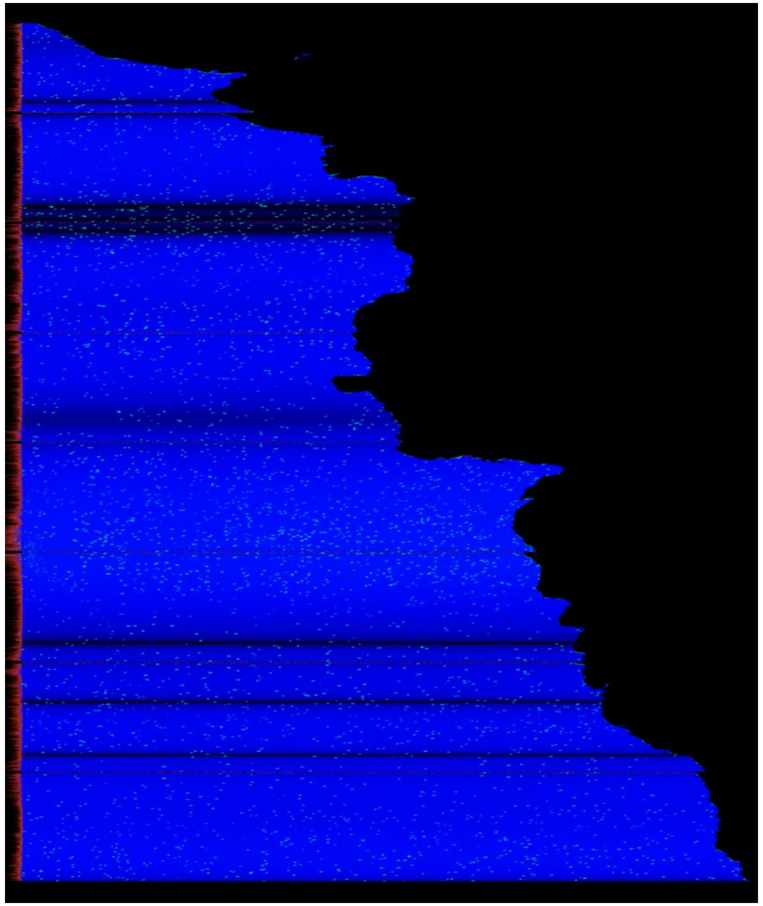
# Demonstration



# Accuracy compared to SNAP – Coregistration operator

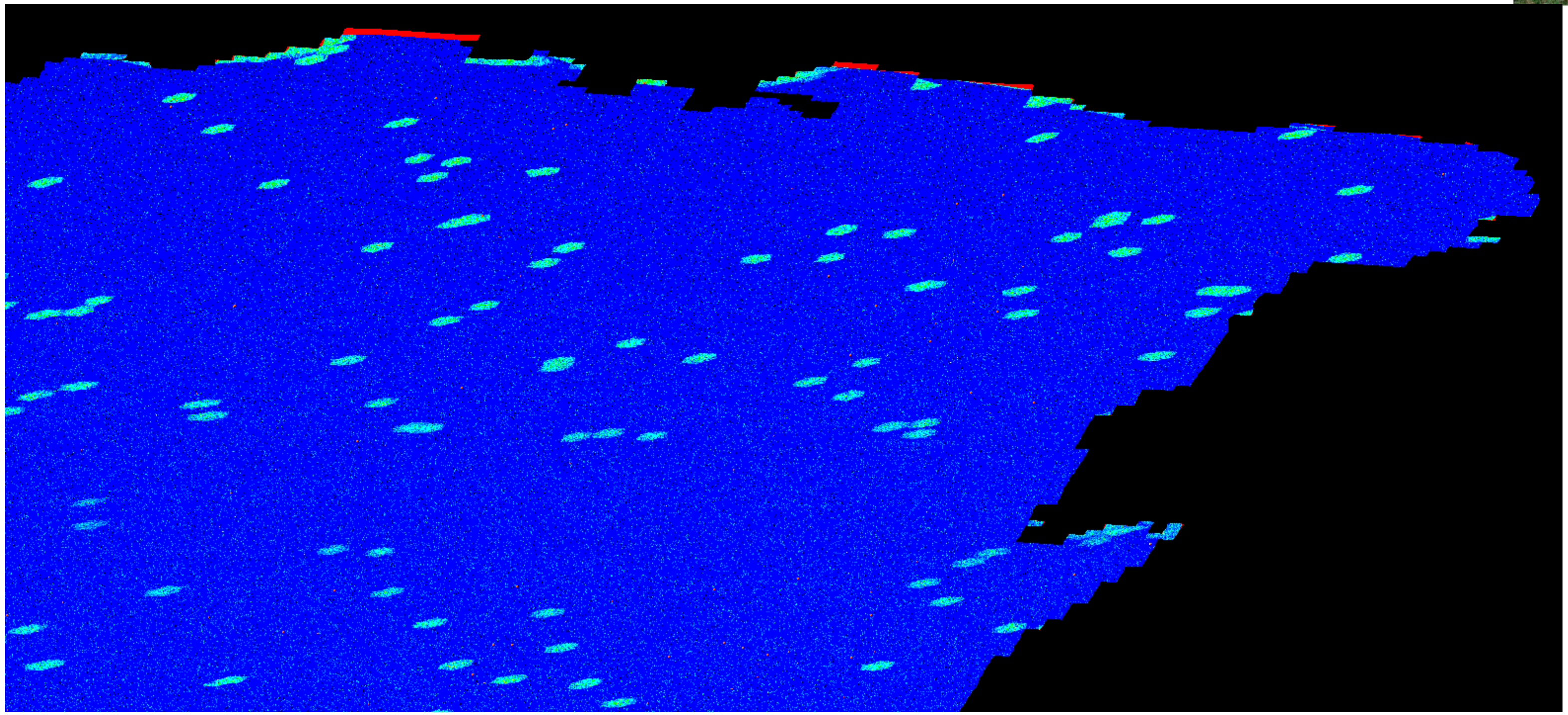
| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |



Dataset: Beirut
Average relative error: 2%, however this is mostly on burst edges

# Accuracy compared to SNAP – Coregistration operator

| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |



# Accuracy compared to SNAP – Coherence operator

| Legend | Relative error |
| --- | --- |
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |

Dataset:S1A_IW_SLC__1SDV_20210424T091623_20210424T091650_037590_046F03_6E8D, S1B_IW_SLC__1SDV_20210430T091550_20210430T091617_026694_03303D_FC2B

Location: Argentina

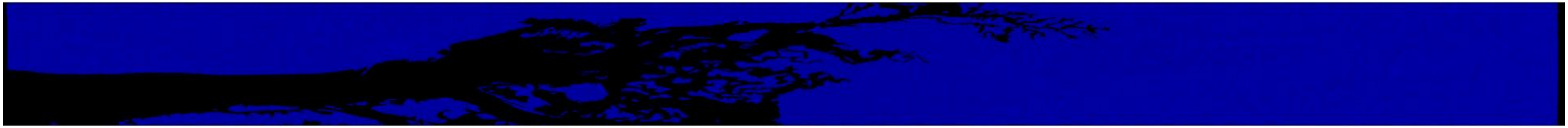
Without flat earth subtraction:

Average relative error of pixels that differ: 0.1ppm

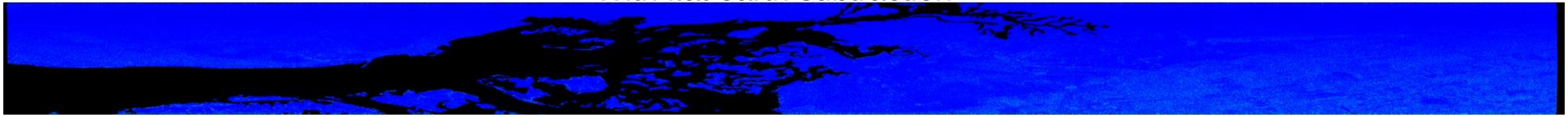
Flat earth subtraction:

Average relative error of pixels that differ: 35ppm

No flat earth subtraction

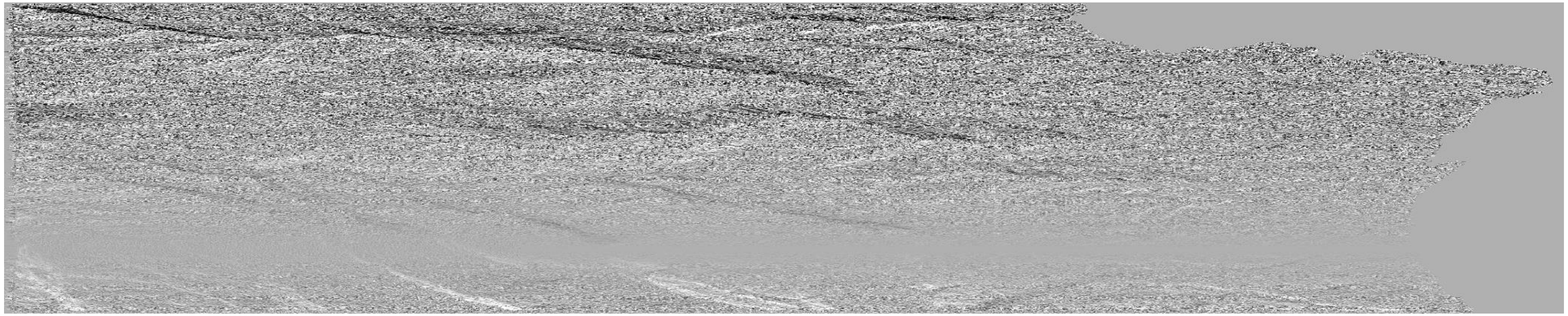


With flat earth subtraction



# Accuracy compared to SNAP – Coherence operator

Flat earth subtraction significantly increases the error and it has a visible pattern.



# Accuracy compared to SNAP – Terrain correction operator

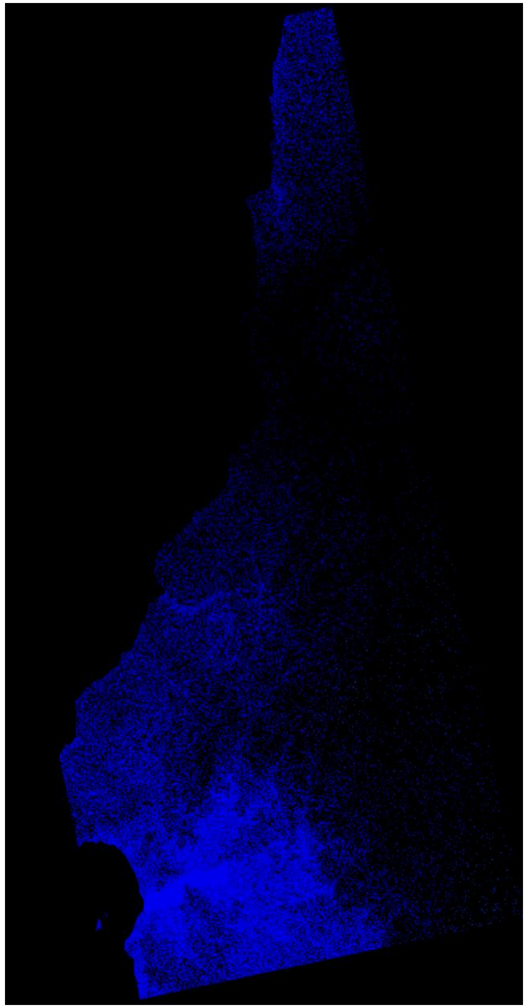
| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |

Dataset:S1A_IW_SLC__1SDV_20210627T173600_20210627T173629_038529_048BF4_EFBE

Location: Equatorial Guinea, Africa

Peak relative error 0.2%

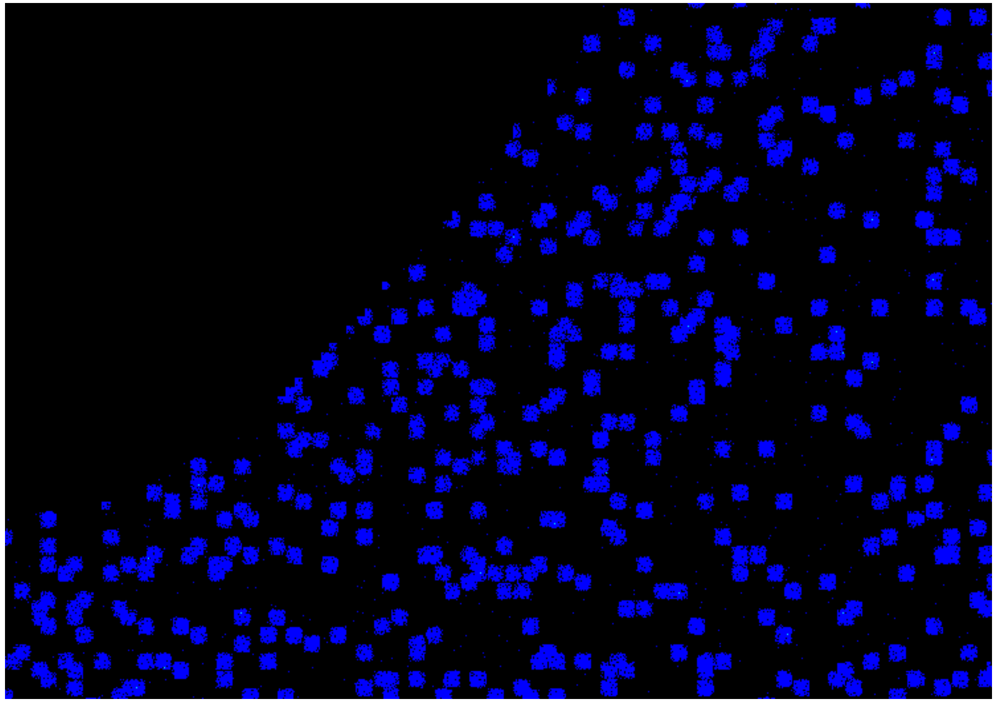
Average error of pixels that differ: less than 1ppm(parts per million)



# Accuracy compared to SNAP – Terrain correction operator

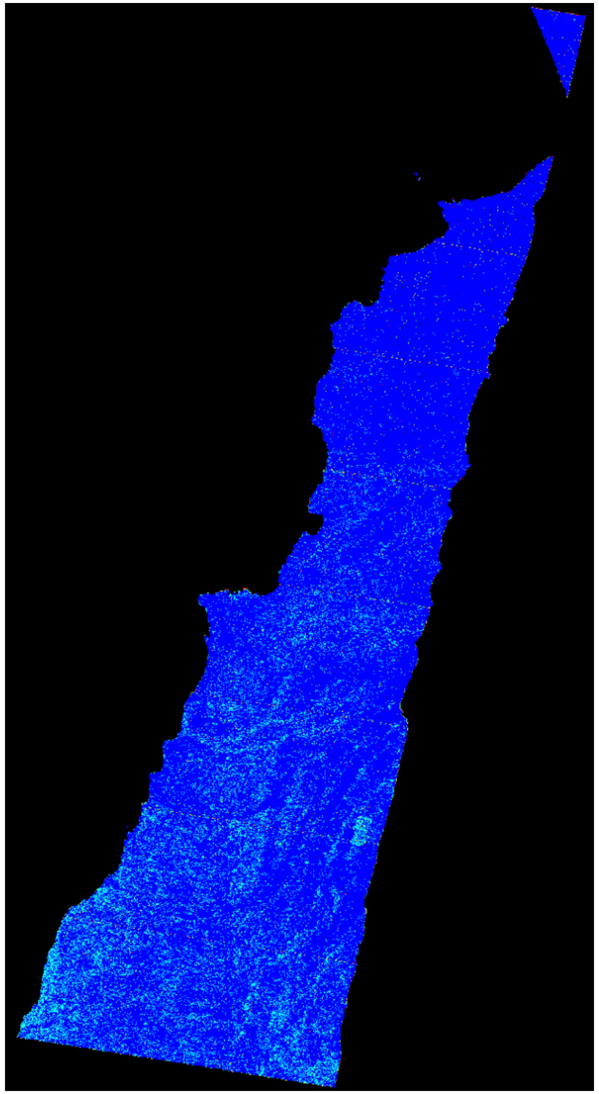
| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |

Relative error image:
A clear pattern occurs, with visible 13x13 pixel squares visible in the image.



# Accuracy compared to SNAP – Coherence chain

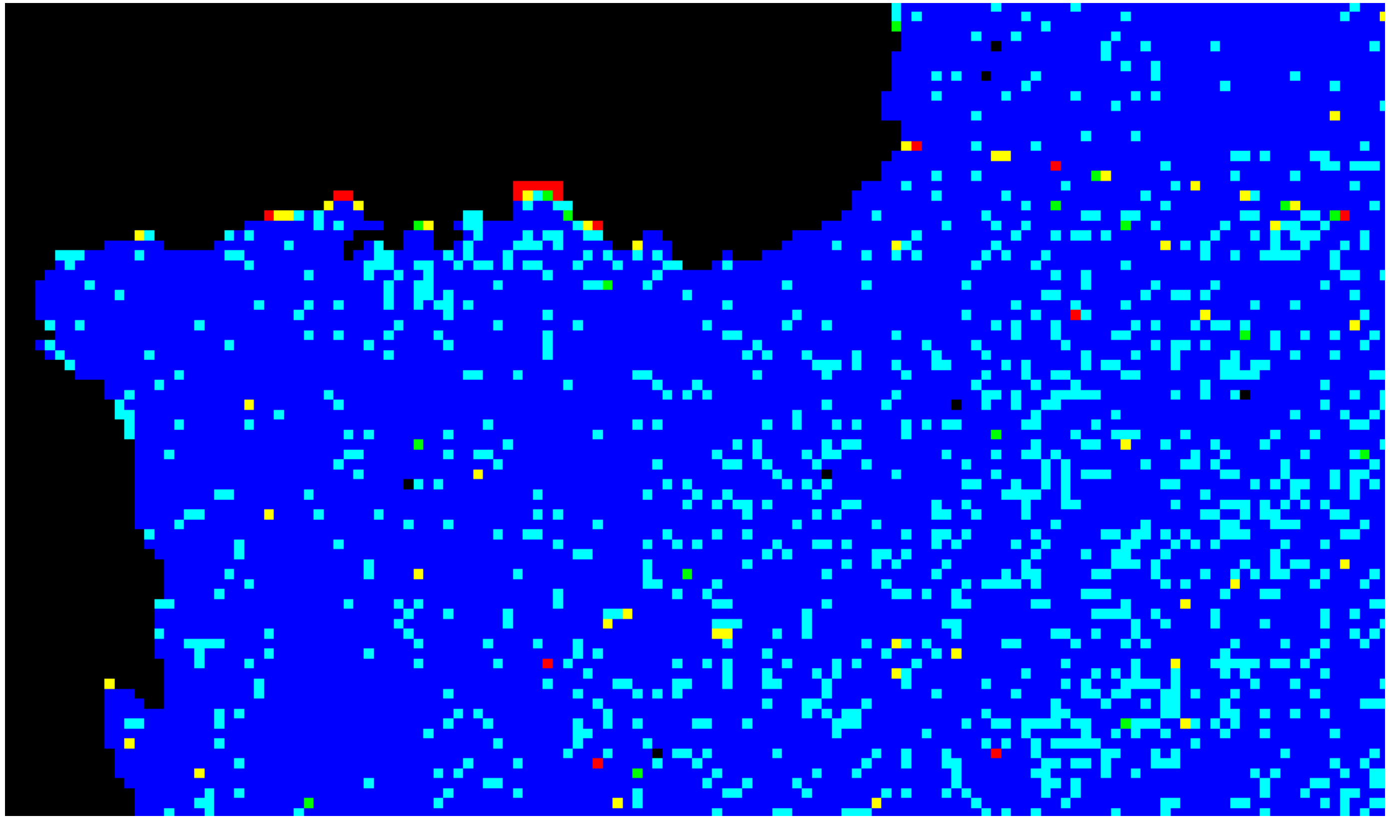
| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |



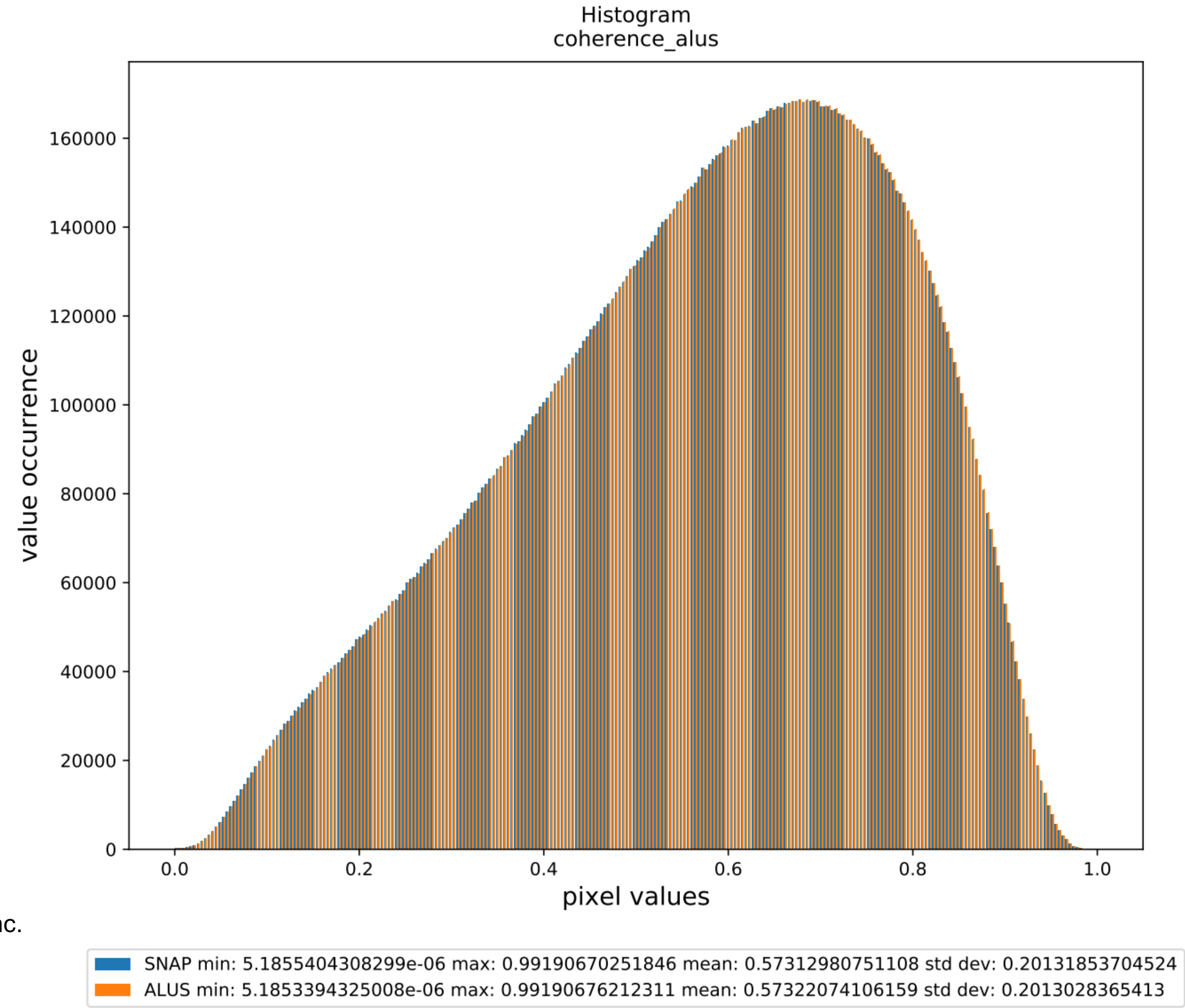
Dataset: Beirut
Average relative error:
0.12%

# Accuracy compared to SNAP – Coherence chain

| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |



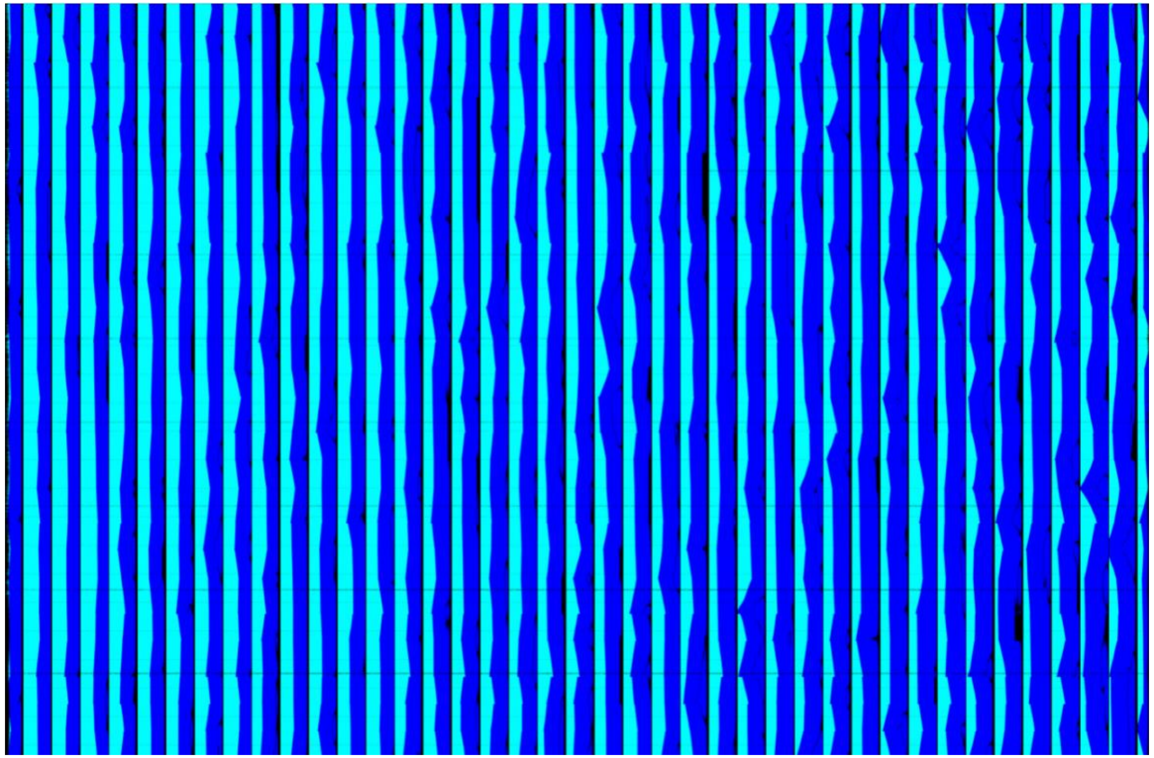
# Accuracy compared to SNAP – Coherence chain



# Accuracy compared to SNAP – Calibration operator

| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |

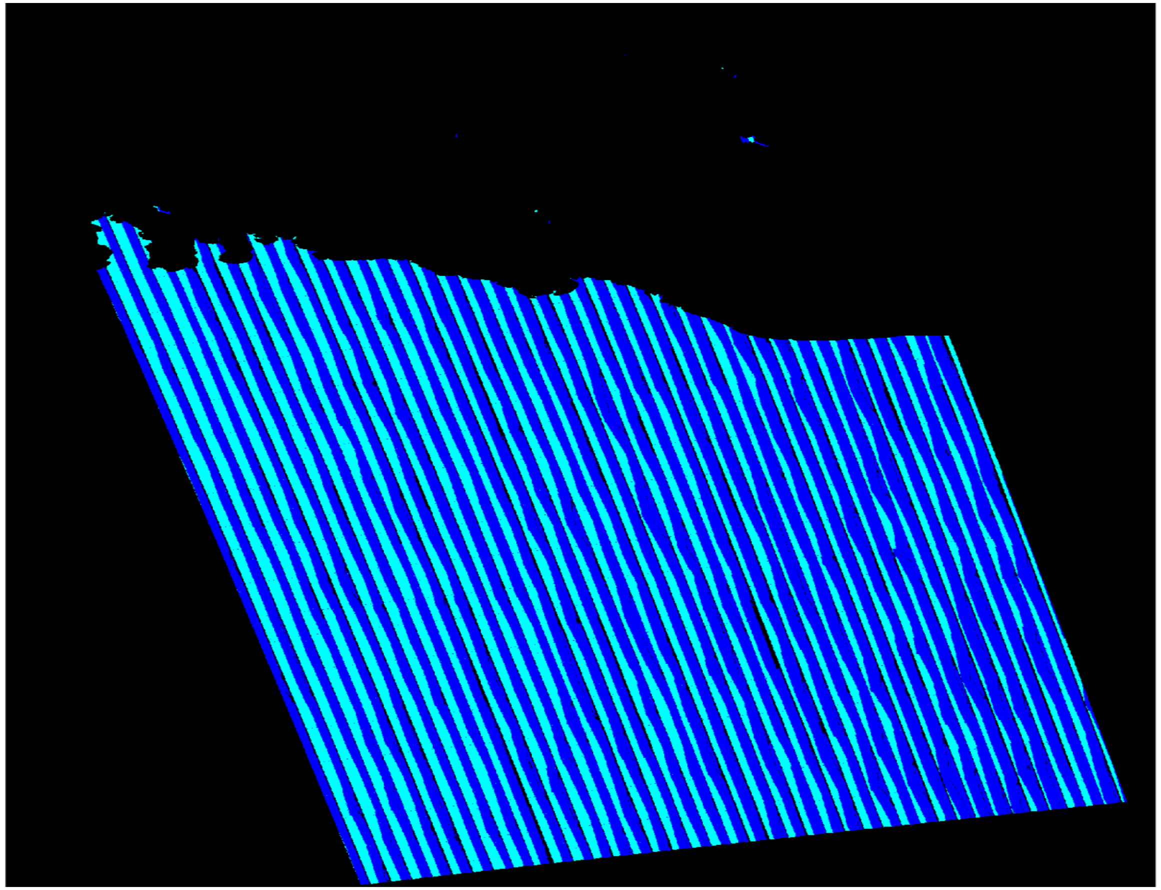
Clear error pattern due to bug in SNAP code (for sigma and gamma, for beta not present)
90% of pixels differ, average error 10ppm
This error is not present in ALUS code therefore this visible error pattern



Sigma

# Accuracy compared to SNAP – Calibration operator

| Legend | Relative error |
|---|---|
| RED | one pixel zero, second is not |
| ORANGE | >10% |
| YELLOW | > 1% |
| GREEN | > 0.1% |
| CYAN | > 100ppm |
| LIGHT BLUE | > 10ppm |
| BLUE | < 10ppm |
| BLACK | equal |

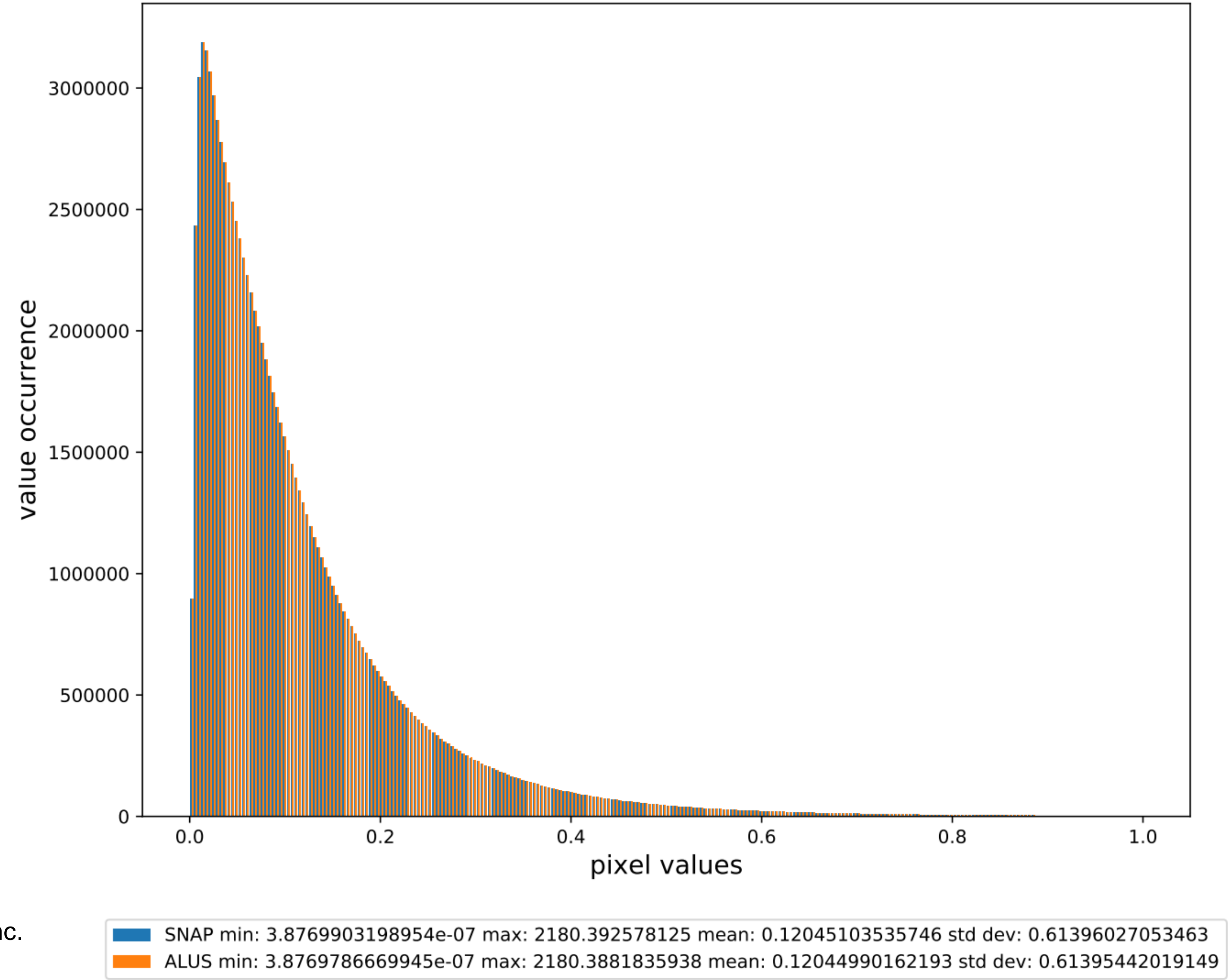


Dataset:

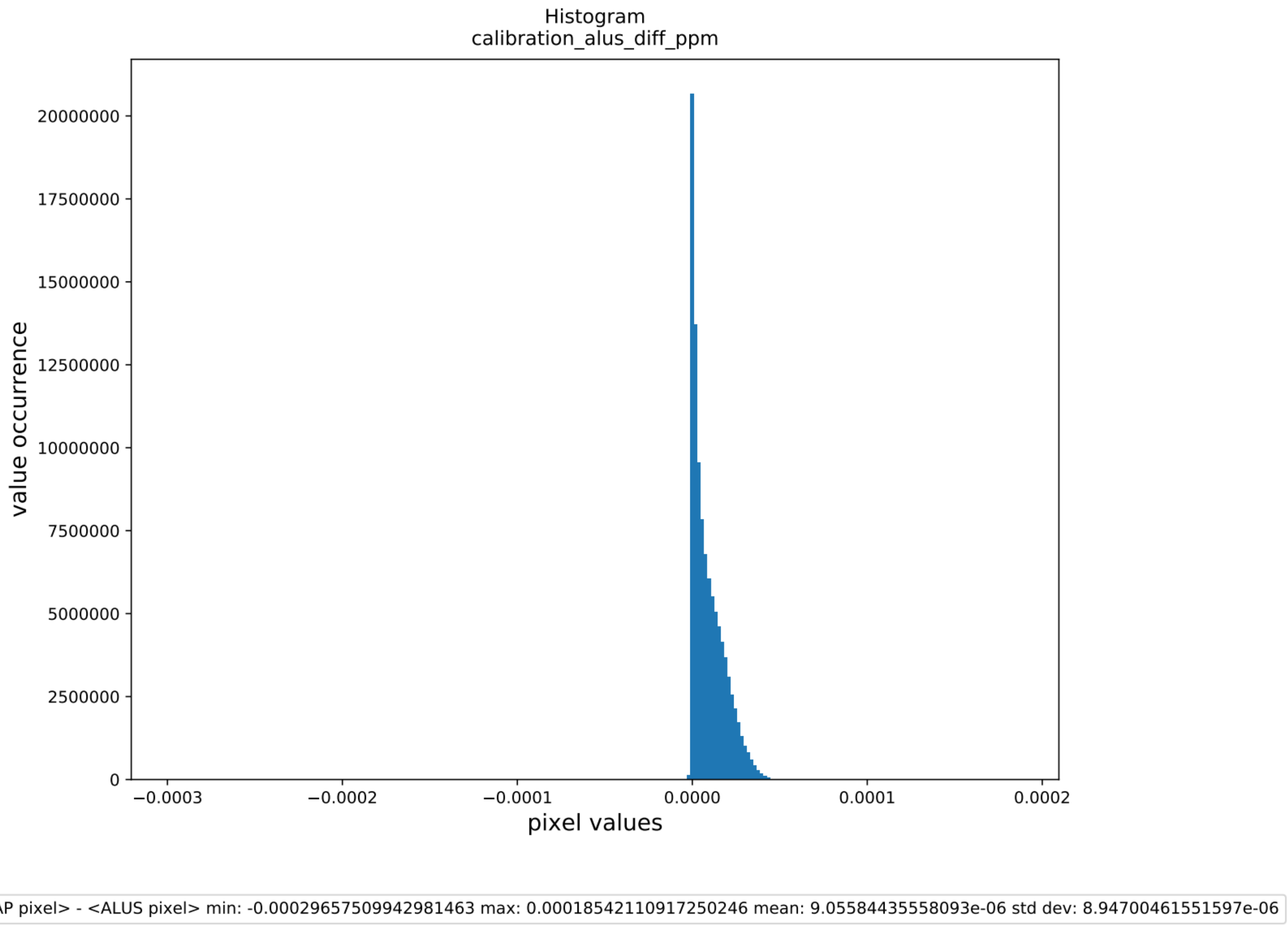
Location: North-East Estonia

Average relative error: 10ppm

# Accuracy compared to SNAP – Calibration chain



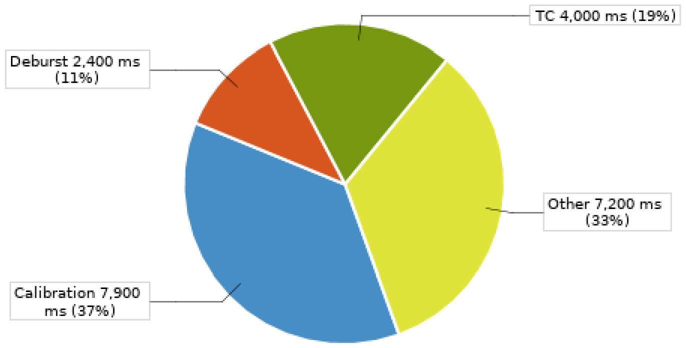
# Accuracy compared to SNAP – Calibration chain



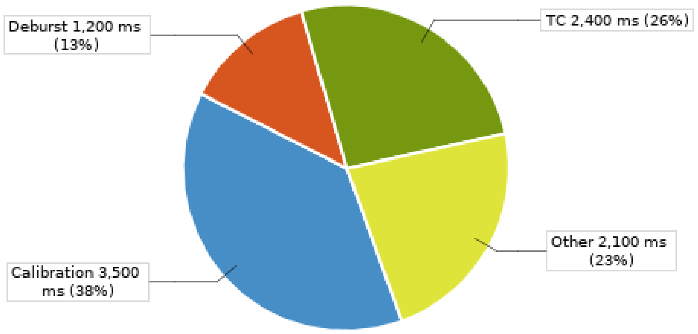
# Performance improvements since June – Calibration chain

Time spent per operator
Tesla V100 AWS P3 instance

June (Total = 21.5s)



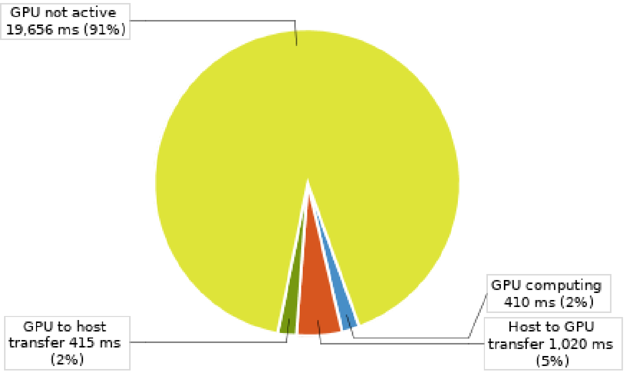

August (Total = 9.2s)



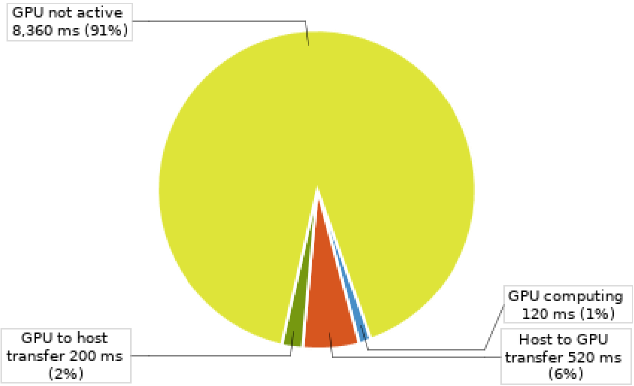
# Performance improvements since June – Calibration chain

## Time spent by GPU activity
## Tesla V100 AWS P3 instance

### June (Total = 21.5s)



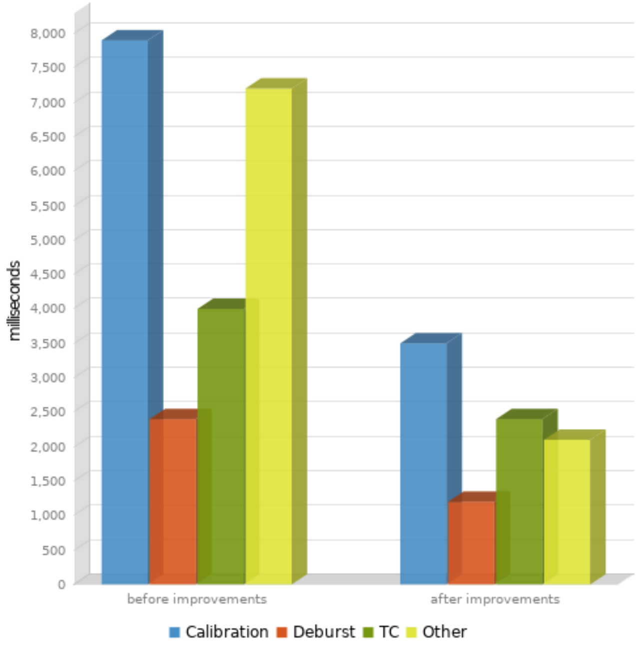

### August (Total = 9.2s)



# Performance improvements since June – Calibration chain

Total chain time spent
Tesla V100 AWS P3 instance

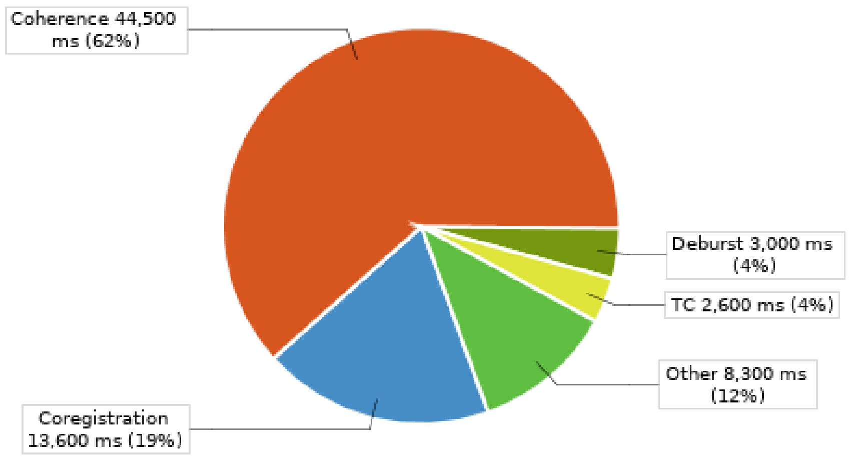
June 21.5 seconds vs August 9.2 seconds



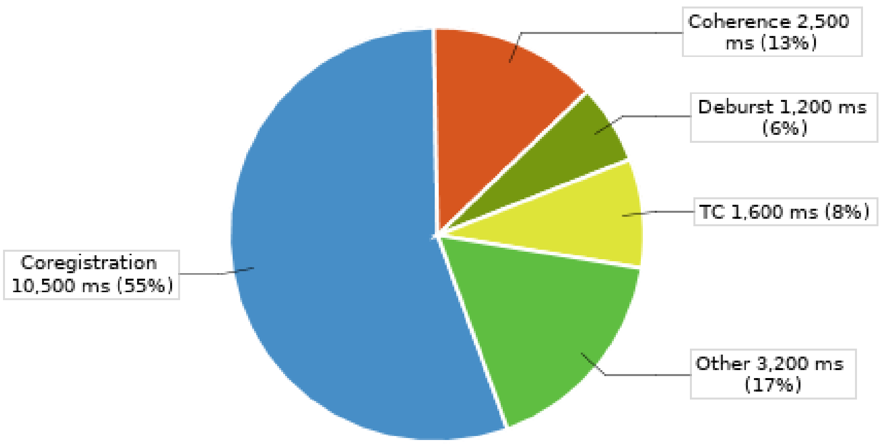
# Performance improvements since June – Coherence chain

Time spent per operator
Tesla V100 AWS P3 instance

June (Total= 72s)



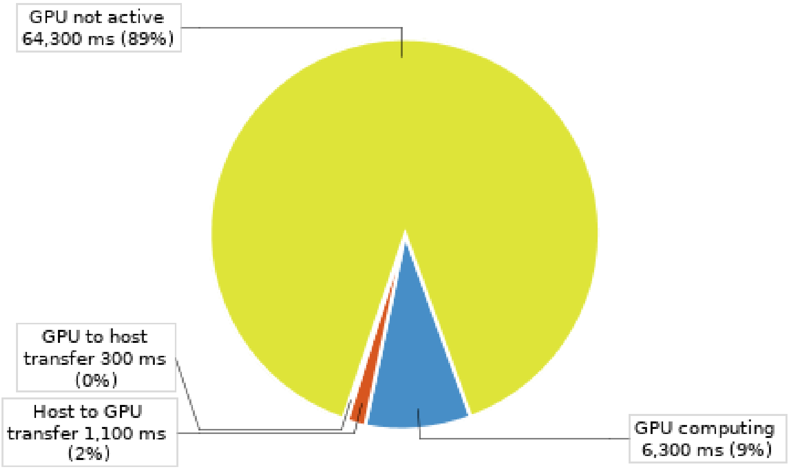

August (Total=19.1s)



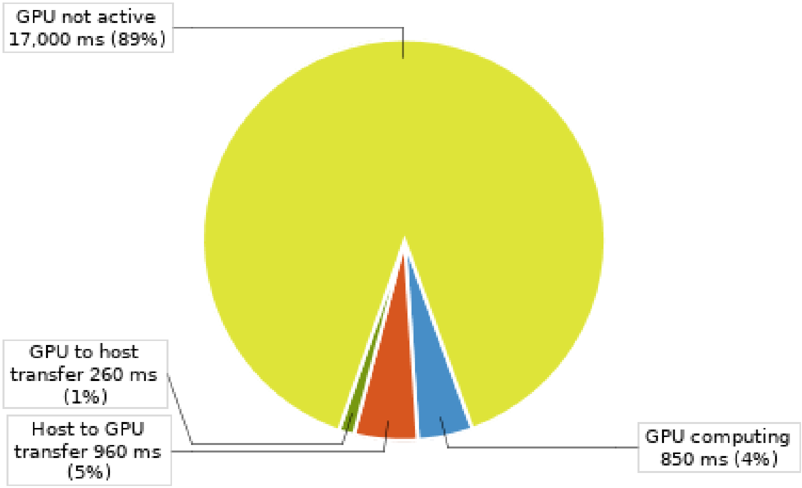
# Performance improvements since June – Coherence chain

Time spent by GPU activity
Tesla V100 AWS P3 instance

June (Total = 72s)



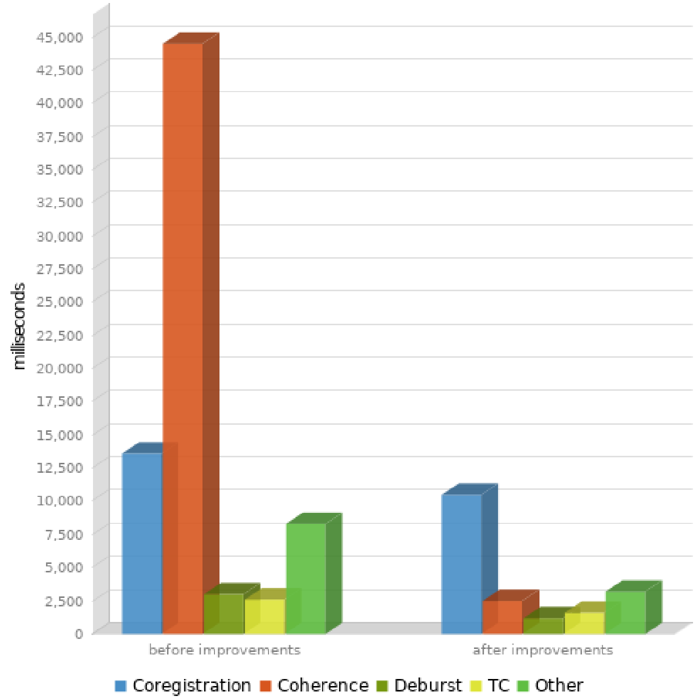

August (Total = 19.1s)



# Performance improvements since June – Coherence chain

Total chain time spent
Tesla V100 AWS P3 instance

June 72 seconds vs August 19.1 seconds



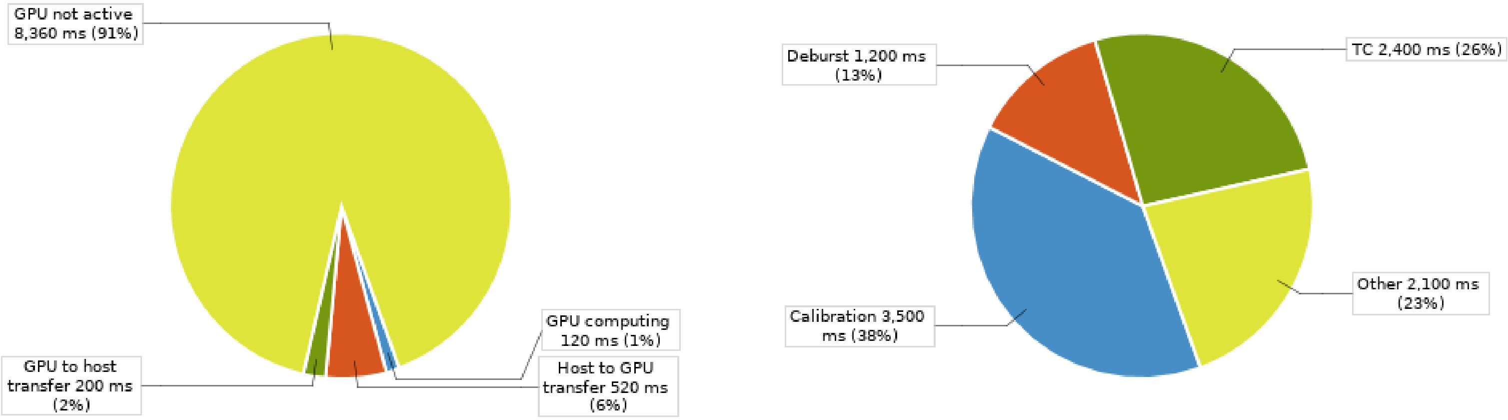
# Problem 1 GPU Double precision FLOPS

| | Compute Capability | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 3.5, 3.7 | 5.0, 5.2 | 5.3 | 6.0 | 6.1 | 6.2 | 7.x | 8.0 | 8.6 |
| 32-bit floating-point add, multiply, multiply-add | 192 | 128 | | 64 | 128 | | 64 | | 128 |
| 64-bit floating-point add, multiply, multiply-add | 64[4] | 4 | | 32 | 4 | | 32[5] | 32 | 2 |

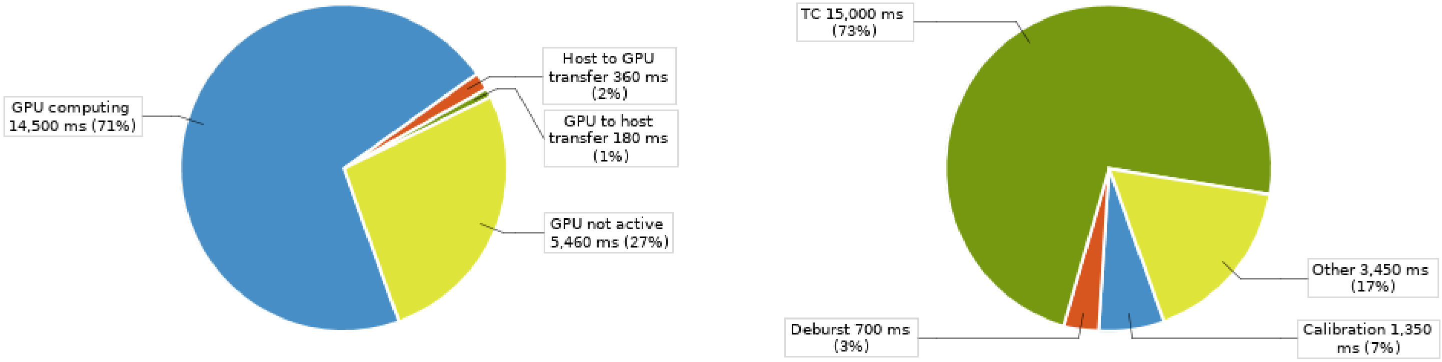
- Most consumer GPUs have float to double performance ration of 1 : 32
- Due to this algorithms heavy on double math do not see significant improvement compared to CPU

# Problem 1 GPU Double precision FLOPS

## Cloud mainframe instance with Nvidia Tesla V100 total 9.2 seconds




## Consumer PC with Nvidia MX250 total 20.5 seconds



# Problem 2 GPU memory requirements

- Sentinel1 SLC full swath size – ~1.25GB, final output similar size
- Small laptop GPUs cannot hold input+output+metadata in GPU memory with only 1-2GB of GPU RAM
- On datacenter GPUs most time is actually spent on reading GeoTIFF files and writing results using GDAL driver
- For example, debursting intuitively does not make sense on the GPU. However, given enough GPU memory, this could be used to avoid relatively expensive intermediate transfers.

# Ideas for further improvements/adjustments

- Systematic errors detection and fixes
- SNAP performance results could be better if machines/instances used are on level cost wise to GPU instances
- First and the last algorithm step are bottlenecked by GeoTIFF I/O - this could be enhanced by using faster disk instances and better file reading logic. For example, reading raster data during GPU initialization.
- Every other algorithm step is bottlenecked by GDAL I/O driver on V100
- On laptop GPUs double performance is a significant bottleneck
- Overall same performance results could be obtained with older GPUs or GPUs that are cheaper and with less capabilities (that is concerning cloud platform mainframe devices)