Exascale Real-Time RFI Mitigation

Rob van Nieuwpoort

netherlands eScience center
by SURF & NWO

ASTRON

Radio Frequency Interference

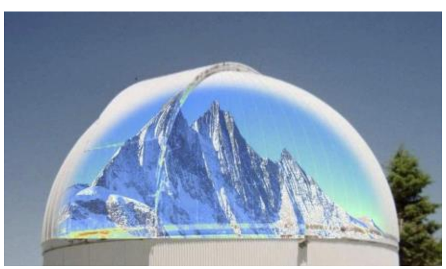
- **RFI is a huge problem for many observations**
- **Caused by**
  - **Lightning, Vehicles, airplanes, satellites, electrical equipment, GSM, FM Radio, fences, reflection of wind turbines, …**
- **Best removed offline**
  - **Complete dataset available**
  - **Good overview / statistics / model**
  - **Can spend compute cycles**

Real-time RFI mitigation

- Some pipelines need to run in real time today
  - Image-based transient detection (LOFAR/AARTFAAC)
  - Pulsar searching (WSRT/Apertif)

- SKA will be entirely real-time
  - Data rates simply too high to store

- Astron & IBM DOME project
- SKA CSP consortium



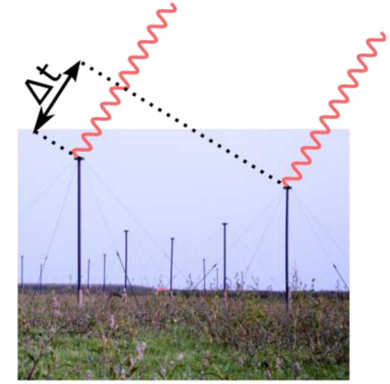
SKA
SQUARE KILOMETRE ARRAY

Real-time RFI mitigation challenges

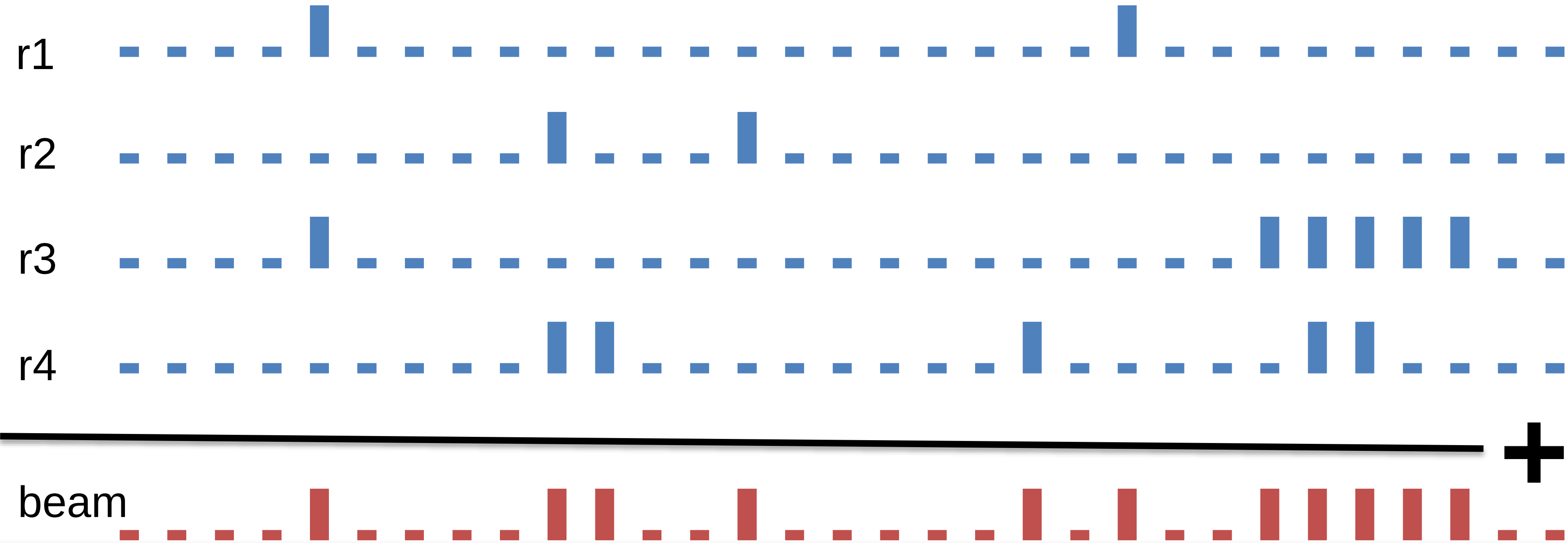
- **Limited amount of samples due to memory and compute constraints**
  - **Only 1 second, no data from the future, only statistics from the past**
  - **Limited statistics due to memory constraints**
  - **Only small number of frequency bands**
- **Distributed system**
- **Real time: We can afford only few operations per byte**

- **Adapt current algorithms, develop new algorithms**

Advantages



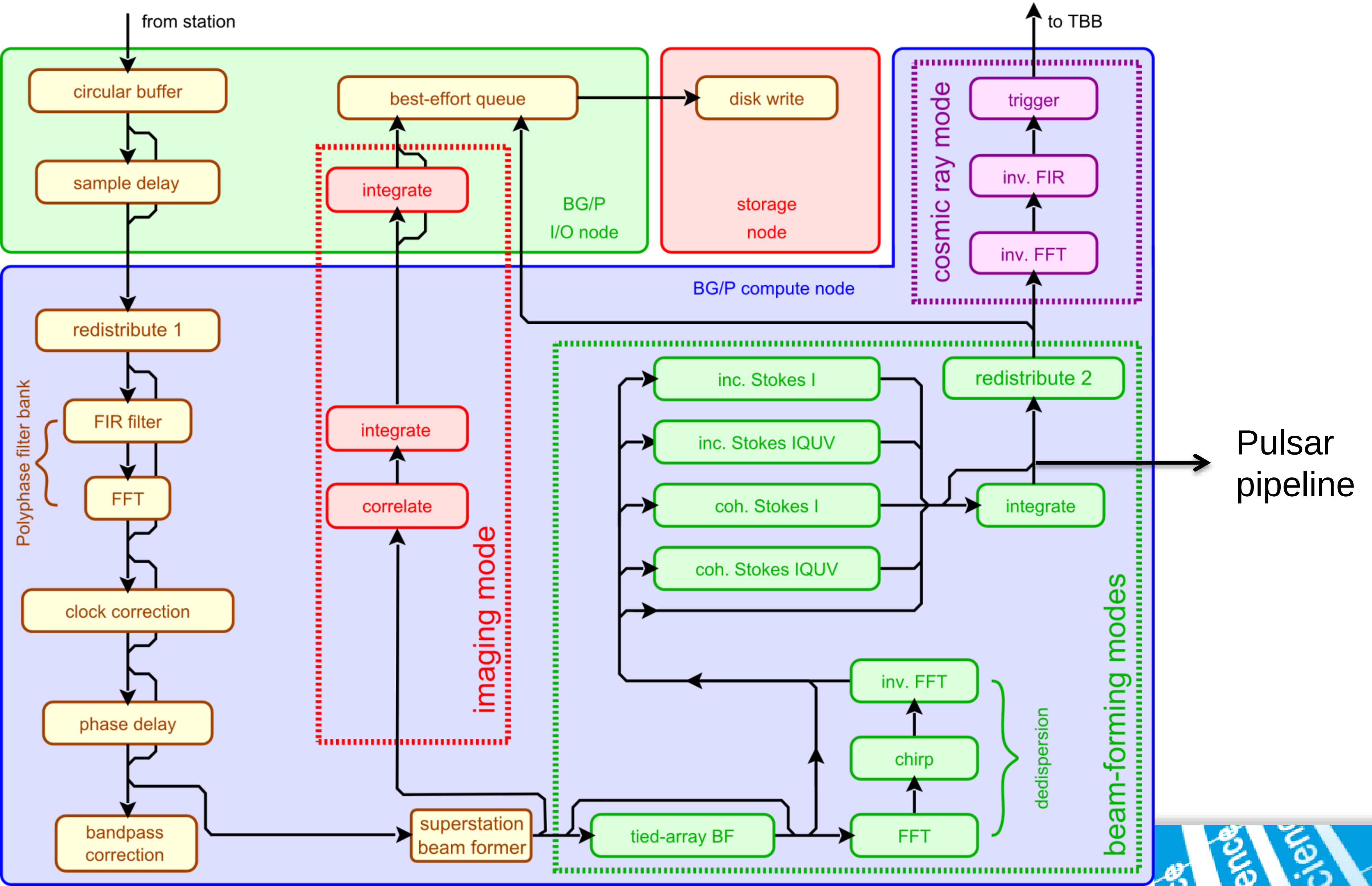
- **Can give better results**
  - **Higher time / frequency resolution before integration**
    Better for bursty and narrowband RFI
  - **Beam forming takes union of RFI of all receivers**



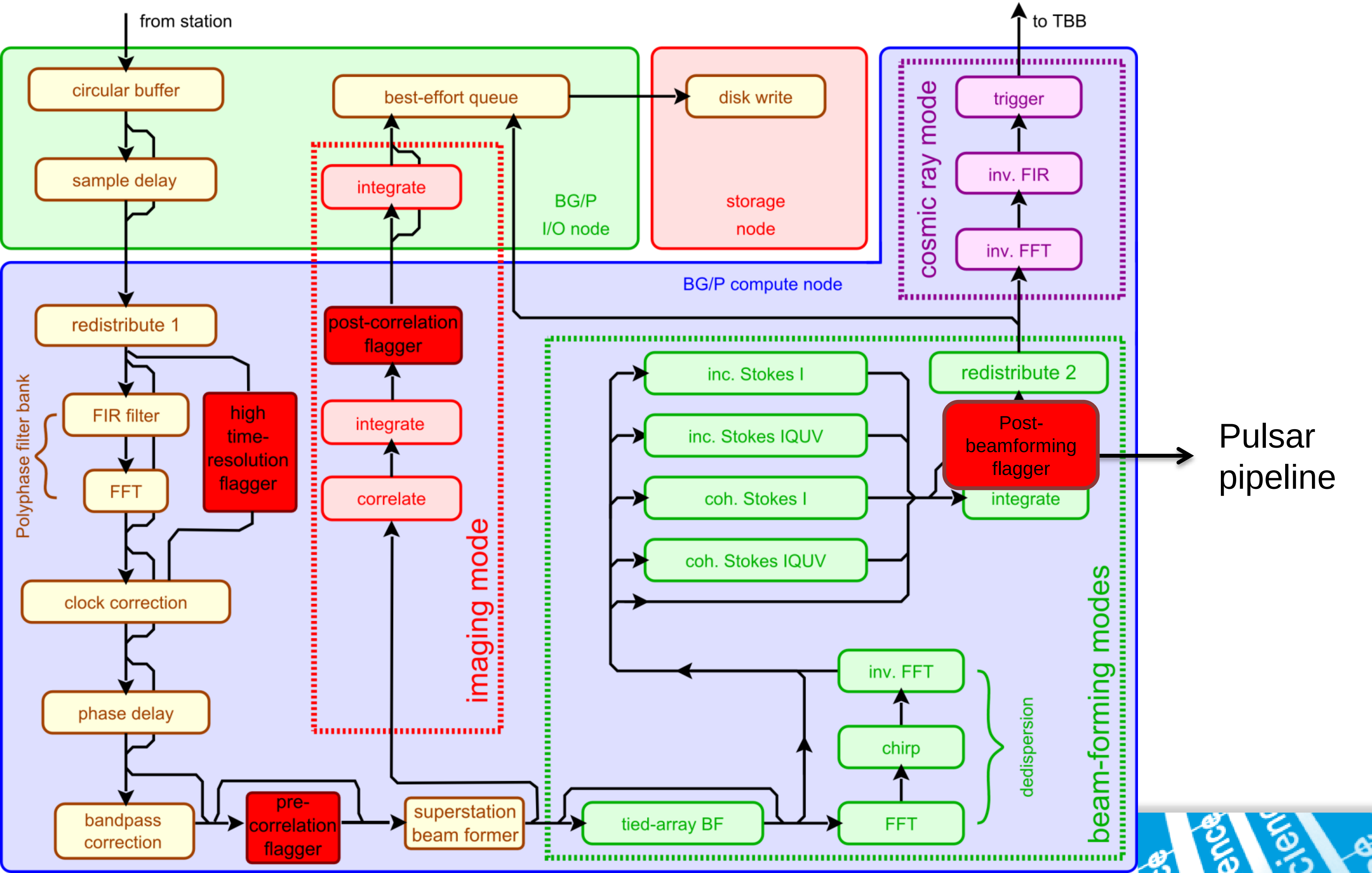
LOFAR

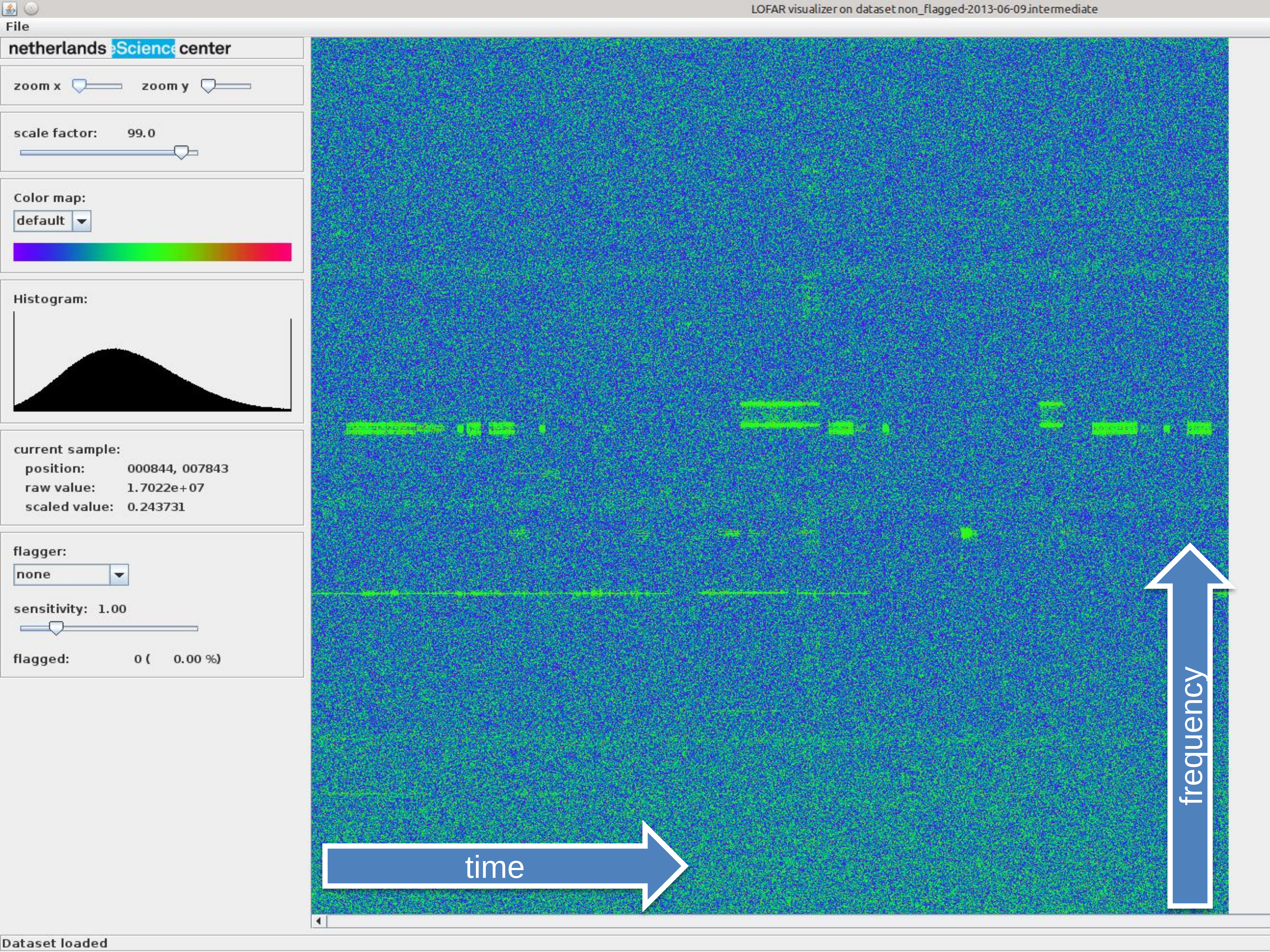

Kristiansand
Trollhättan
Göteborg
Borås
Aberdeen
Dundee
Perth
Kirkcaldy
Edinburgh
Aalborg
Halmstad
Randers
Holstebro
Silkeborg
Herning
Aarhus
Helsingborg
København
Malmö
Danmark
(Denmark)
Esbjerg
United
Kingdom
Flensburg
Kiel
Rostock
Lübeck
York
Hull
Leeds
Manchester
Sheffield
Liverpool
Stoke-on-Trent
Nottingham
Leicester
Birmingham
Coventry
Cambridge
Stade
Hamburg
Bremen
Hannover
Spandau
Berlin
Nederland
(Netherlands)
Den Haag
Bielefeld
Braunschweig
Magdeburg
Halle
(Saale)
Leipzig
Breda
Essen
Antwerpen
Düsseldorf
Köln
Gento
Belgique
België
(Belgium)
Deutschland
(Germany)
Chemnitz
Zwickau
Frankfurt
am Main
Cardiff
London
Bristol
Feltham
Southampton
Portsmouth
Brighton
Exeter
English
Channel
Lëtzebuerg
(Luxembourg)
Nürnberg
Plzeň
Saarbrücken
Reims
Octeville
Le Havre
Guernsey
Jersey
Caen
Paris
Karlsruhe
Nancy
Strasbourg
Stuttgart
Augsburg
Freiburg im
Breisgau
München
Rennes
Le Mans
Alençon
Troyes
Angers
Tours
Nantes
Dijon
Basel
Schweiz
Suisse
Svizzera
Liechtenstein
France
Poitiers
Lausanne
Linz
Wels
Passau
Landau

LOFAR Real-Time Processing



LOF: LOFAR Online Flagger




LOFAR visualizer on dataset non_flagged-2013-06-09.intermediate
File
netherlands eScience center
zoom x
zoom y
scale factor: 99.0
Color map:
default
Histogram:
current sample:
position: 000844, 007843
raw value: 1.7022e+07
scaled value: 0.243731
flagger:
none
sensitivity: 1.00
flagged: 0 (0.00 %)
Dataset loaded
time
frequency

SumThreshold and AOFlagger

- **André Offringa et al**

  [post-correlation radio frequency interference classification methods, Monthly Notices of the Royal Astronomical Society, Vol 405:1, 2010.]

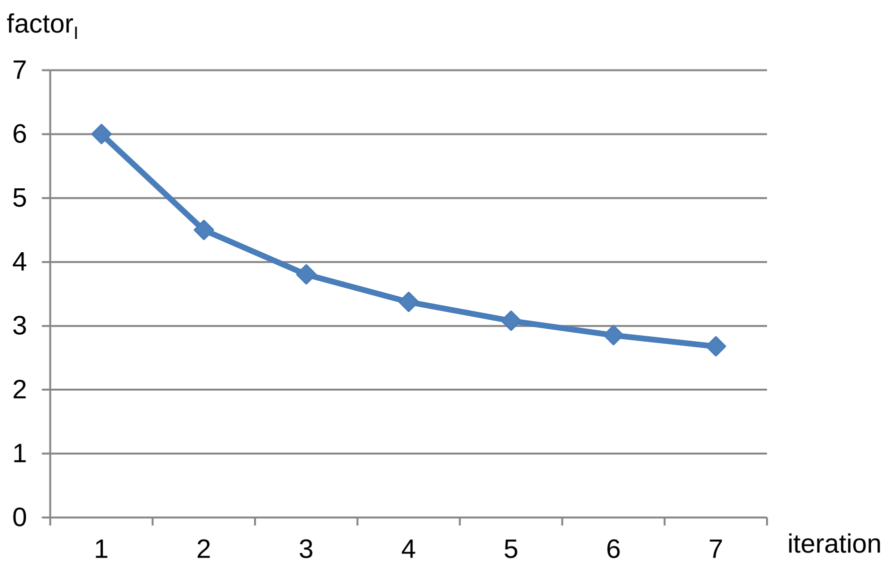
- **Thresholding with exponentially increasing window and sharper threshold**
  - **Detect RFI at different scales**
- **Fast enough to be applied in modern high resolution observatories**
- **Used in the default LOFAR offline pipeline**

Increasingly lower threshold

$$threshold_I = median + stddev \cdot factor_I \cdot sensitivity$$

$$factor_I = \frac{startThreshold \cdot p^{2\log(I)}}{I}$$

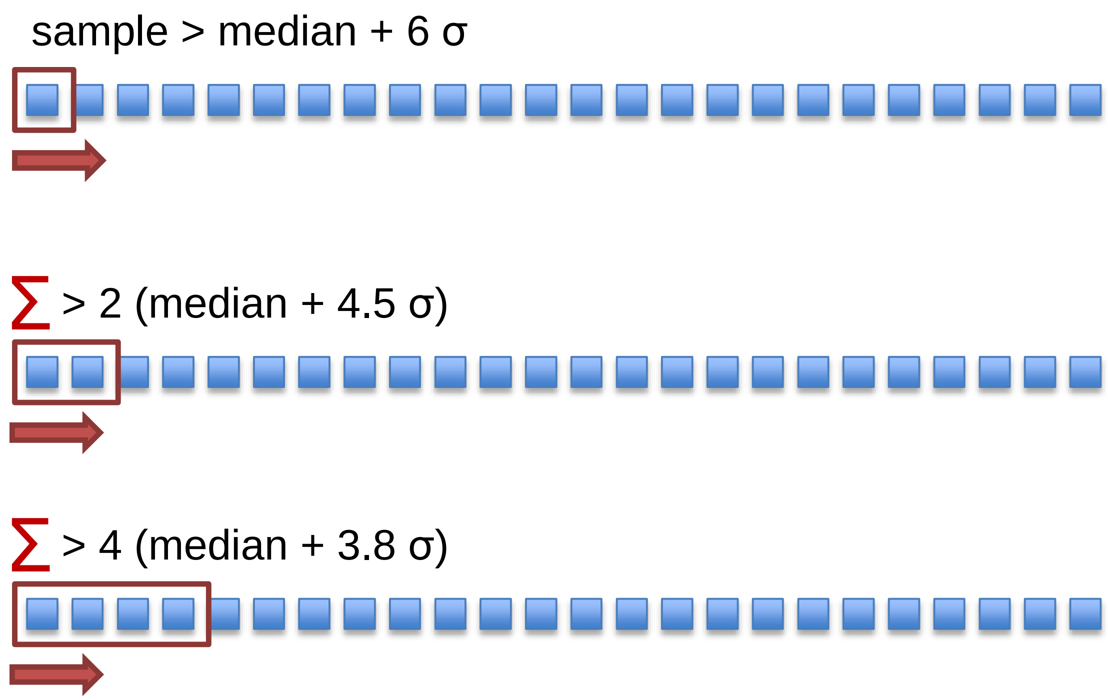
Typical p = 1.5; Typical sensitivity = 1.0



iteration	window	factor
1	1	6.00
2	2	4.50
3	4	3.80
4	8	3.38
5	16	3.08
6	32	2.85
7	64	2.68

SumThreshold

iter	window	factor
1	1	6.00
2	2	4.50
3	4	3.80



Scale-Invariant Rank operator (SIR)

- **André Offringa**
  - **A morphological algorithm for improving radio-frequency interference detection [*Astronomy & Astrophysics*, Volume 539, Issue A95, March 2012]**
- **One-dimensional morphological technique**
- **Can be used to find adjacent intervals in the time or frequency domain that are likely to be affected by RFI**
- **Faster, linear, algorithm by Jasper van de Gronde**

- **Only run on data flagged by algorithm: data flagged due to other causes (dropped UDP packets) is precise**

SIR Operator

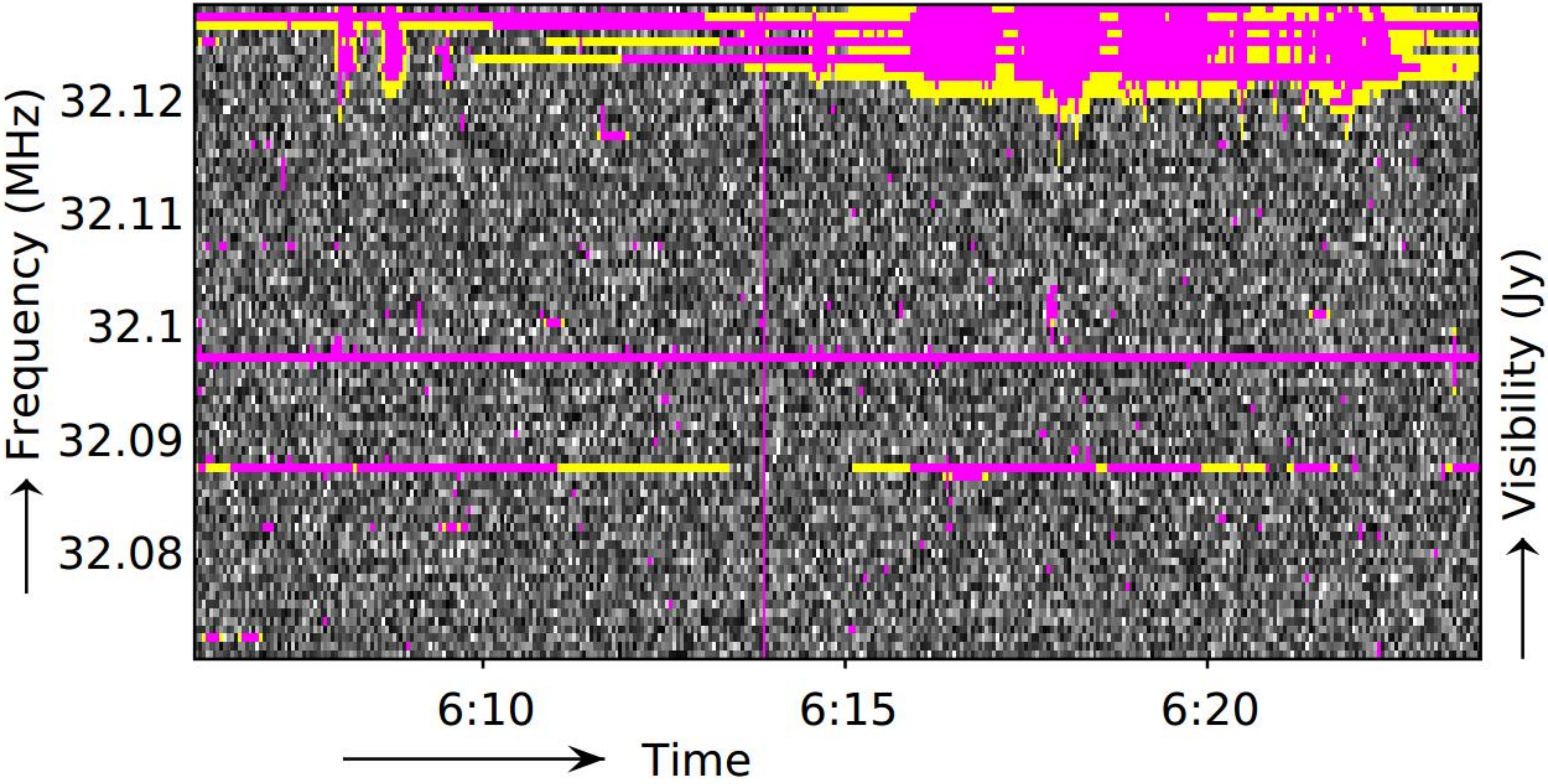


Image courtesy André Offringa

Changes for real-time use

- Compute amplitudes and integrate data
  - Improves sensitivity
  - Reduces computation time
  - Integrate time direction fully for frequency flagging
  - Integrate frequency fully for time domain flagging
  - 2D flagging: partially integrate in both directions

- All algorithms have linear computational complexity in the number of samples
    - PreCorrelation: O(nrStations * nrPols * nrChannels * nrTimes)
    - PostCorrelation: O(nrBaselines * nrPols * nrChannels)
    - PostBeamforming: O(nrBeams * nrChannels * nrTimes)

- Optional smoothing spectral detector
  - Low-pass filter in frequency direction (convolution with gaussian); SumThreshold on difference
  - Better sensitivity for narrowband RFI
  - Very limited improvement in practice
  - Expensive

Statistics

- Trivial in theory, much harder in practice
- Important, since sample set is small
- Medians are expensive
  - Use fast O(n) median (more robust than mean)
  - May require extra data copy
- Winsorized: ignore top and bottom 10% for means and standard deviations
- In the wrong place
  - Complex communication patterns due to scheduling
  - Asynchronous communication & synchronization issues
  - Cannot compute running statistics (second of data takes more than a second to compute)

Algorithm example: post-correlation 1D

```
// Do the following in frequency and time directions

for p in polarizations {
    calculateAndIntegratePowers(p)

    calculateStatistics(p)

    flagger(p)
    if(samplesFlagged) {          // Make more robust:
        calculateStatistics(p) // recalculate statistics
        flagger(p)             // omitting flagged data.
    }
}
takeUnionOfFlags()              // Unify polarizations.
ScaleInvariantRankOperator() // Expand flagged regions.
HistoryFlagger()               // Flag based on statistics.
```

Start with frequency direction: remove strong narrowband RFI. It pollutes integrated data and statistics.

Real-time extension: History Flagger

```
// For all channels we do the following:

// We keep an array (sliding window) of means of
// the past seconds, for each frequency channel

currentValue = winsorizedMeanOfUnflaggedSamples()

threshold = historyMean + sensitivity * historyStddev

if(currentValue < threshold) {
    addToHistory(station, subband, currentValue)
} else {
    addToHistory(station, subband, threshold)
    flagThisIntegrationTime()
}
```

Space requirements of history statistics

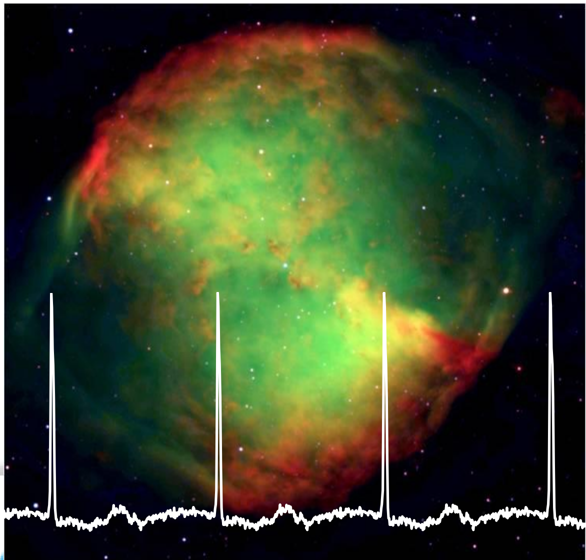
- Pre correlation
  - stations x subbands x channels x 32 bit float
  - 64 x 248 x 256 x 4 = 15.5 MByte per second
  - 5 Minutes = 300 samples = 4.6 GByte
- Post correlation
  - baselines x subbands x channels x 32 bit float
  - 2080 x 248 x 256 x 4 = 504 MByte per second
  - 5 minutes = 300 samples = 148 Gbyte
- Downsample as required

Observation

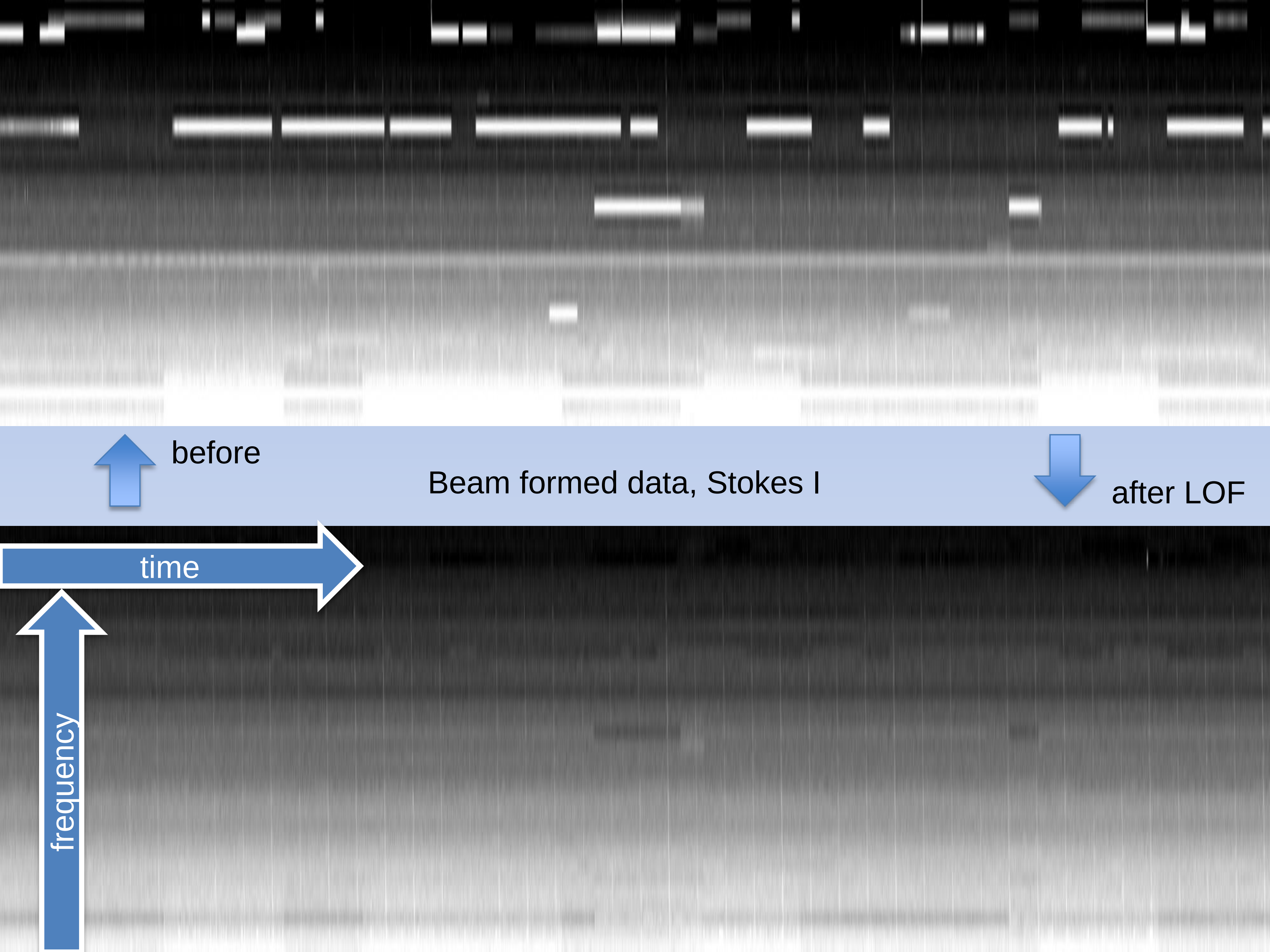
- pulsar B1919+21; period 1.3373 s, pulse width 0.04 s, DM 12.455.
- Observed at 138.0 – 145.2 MHz (32 subbands)
- 5 stations: CS005, CS006, RS205, RS406, UK608
  - Core vs remote: correlated vs uncorrelated RFI, uk station

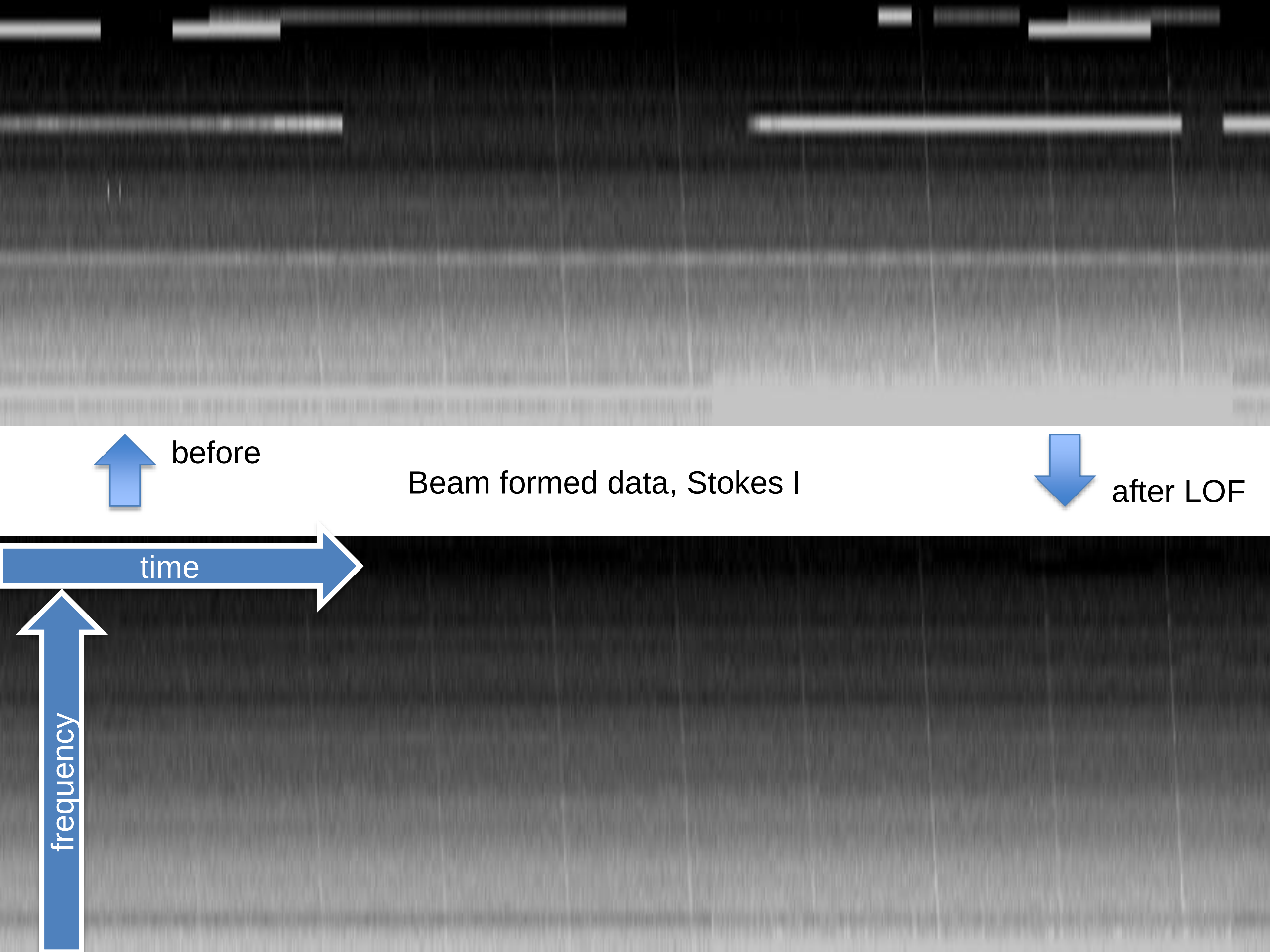
- Stored raw UDP data: can replay real-time pipeline offline
- 16 channels (12 KHz / 82 μs) or 256 channels (0.8 KHz / 1.3 ms)

- Pulsar pipeline allows for quantitative comparison: SNR of folded pulse profile

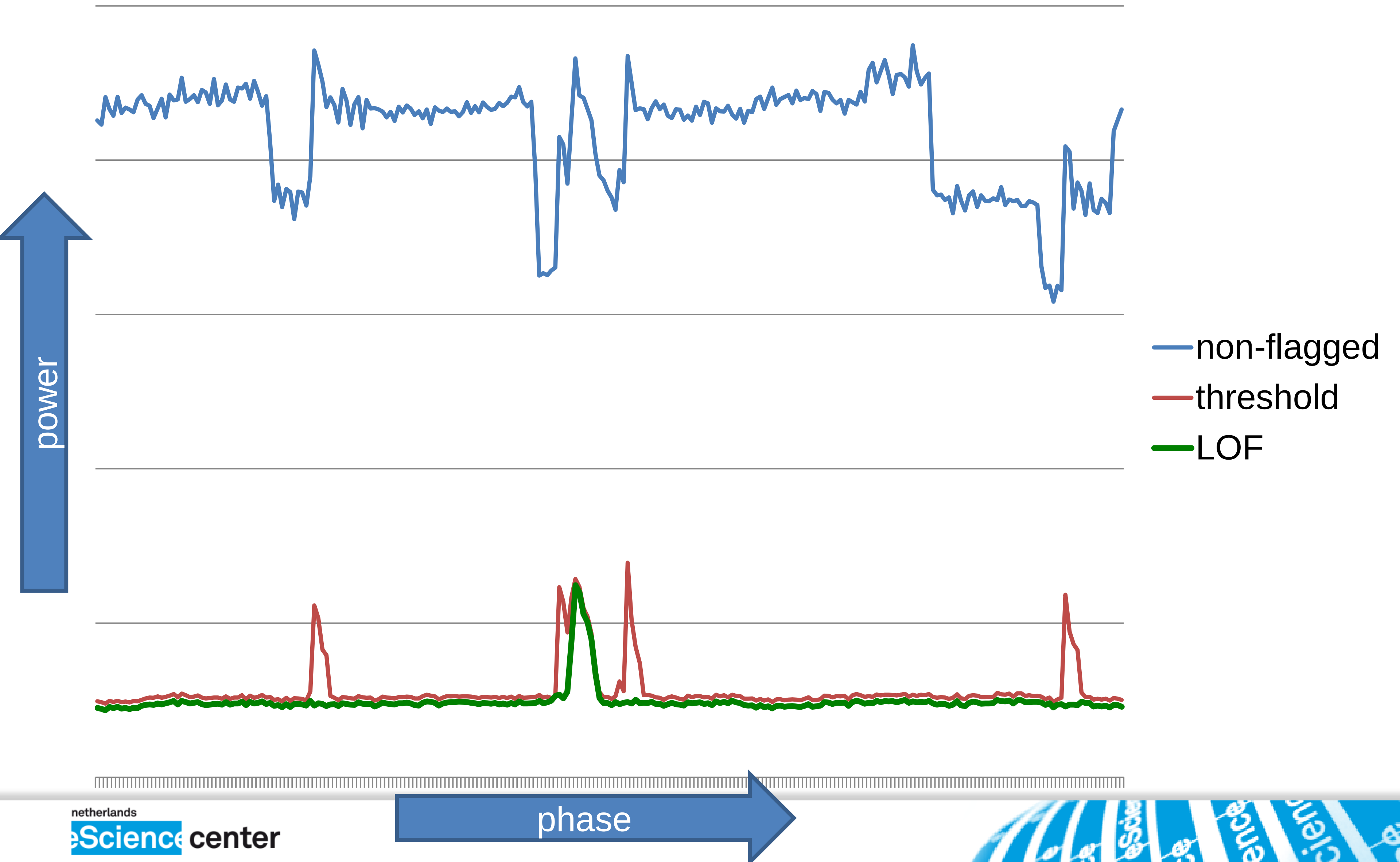


Pulsar B1919+21 in the Vulpecula nebula.
Pulse profile created with the LOFAR software telescope.
Background picture courtesy European Southern Observatory.


before
Beam formed data, Stokes I
after LOF
time
frequency

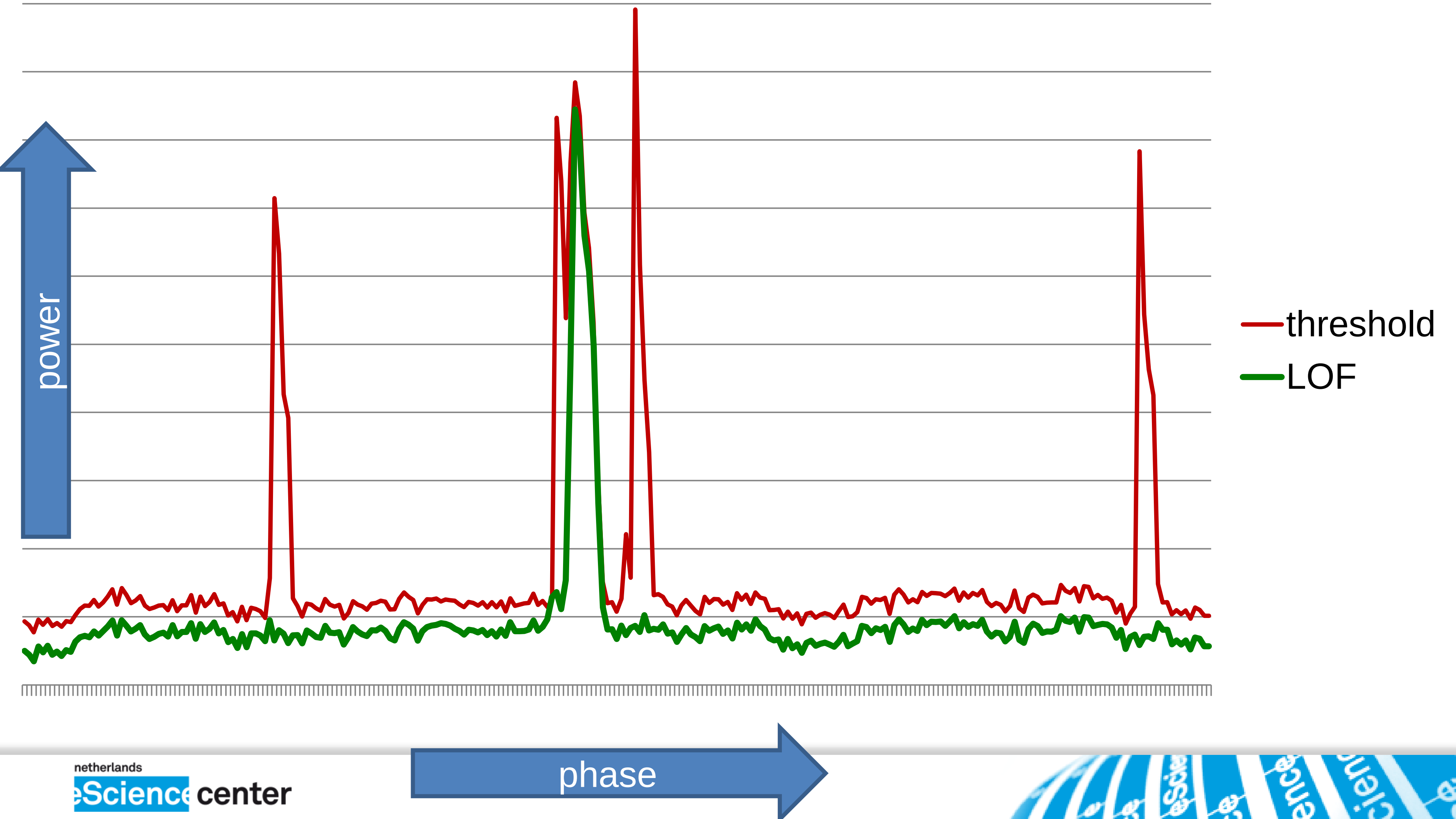

before
Beam formed data, Stokes I
after LOF
time
frequency

Folded pulse profile

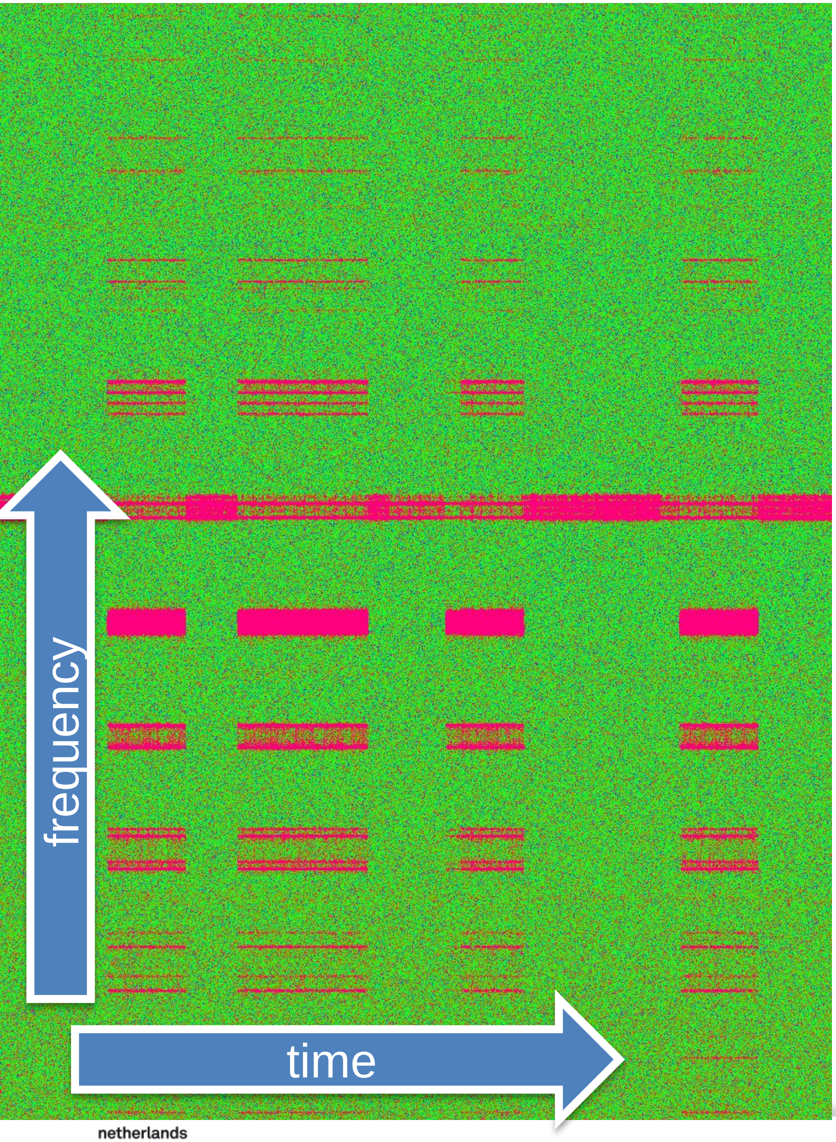


netherlands eScience center

Folded pulse profile

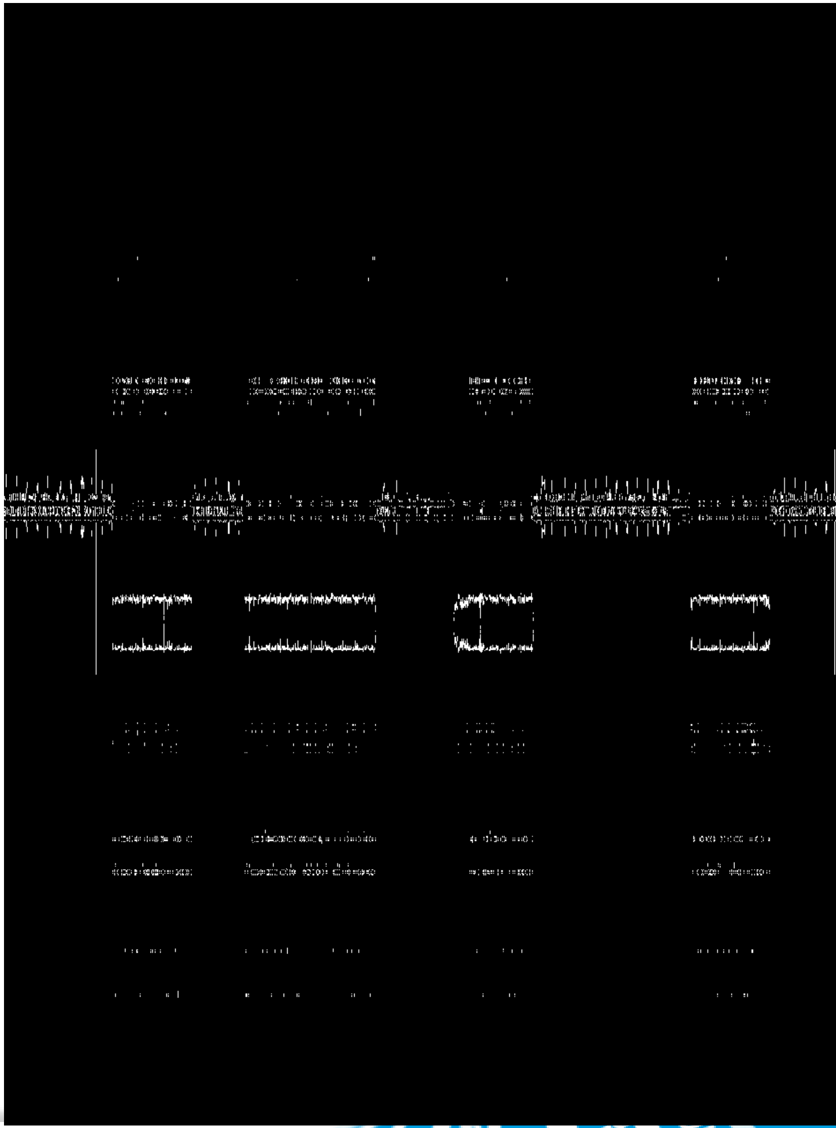


Threshold vs LOF

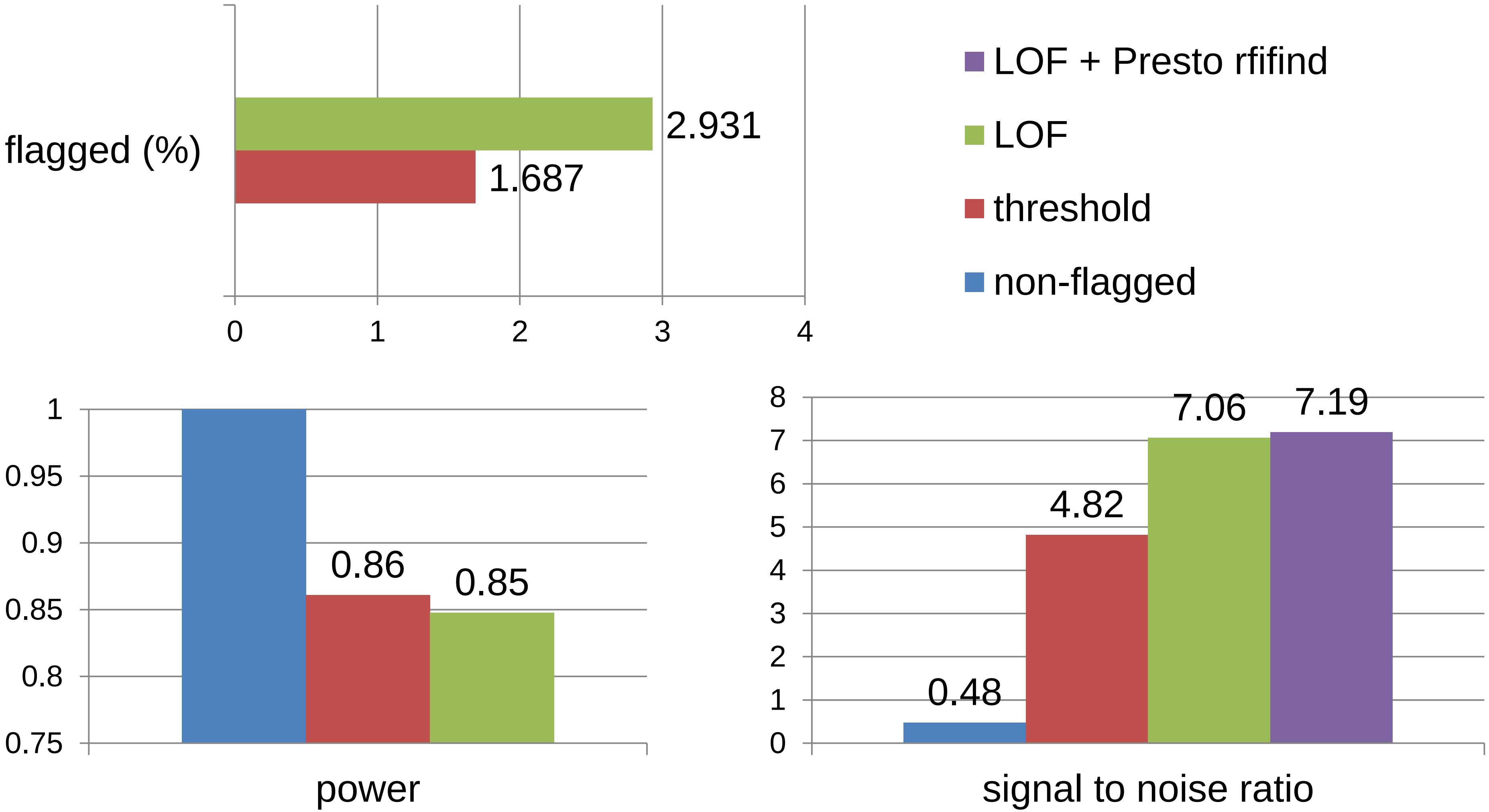


source data

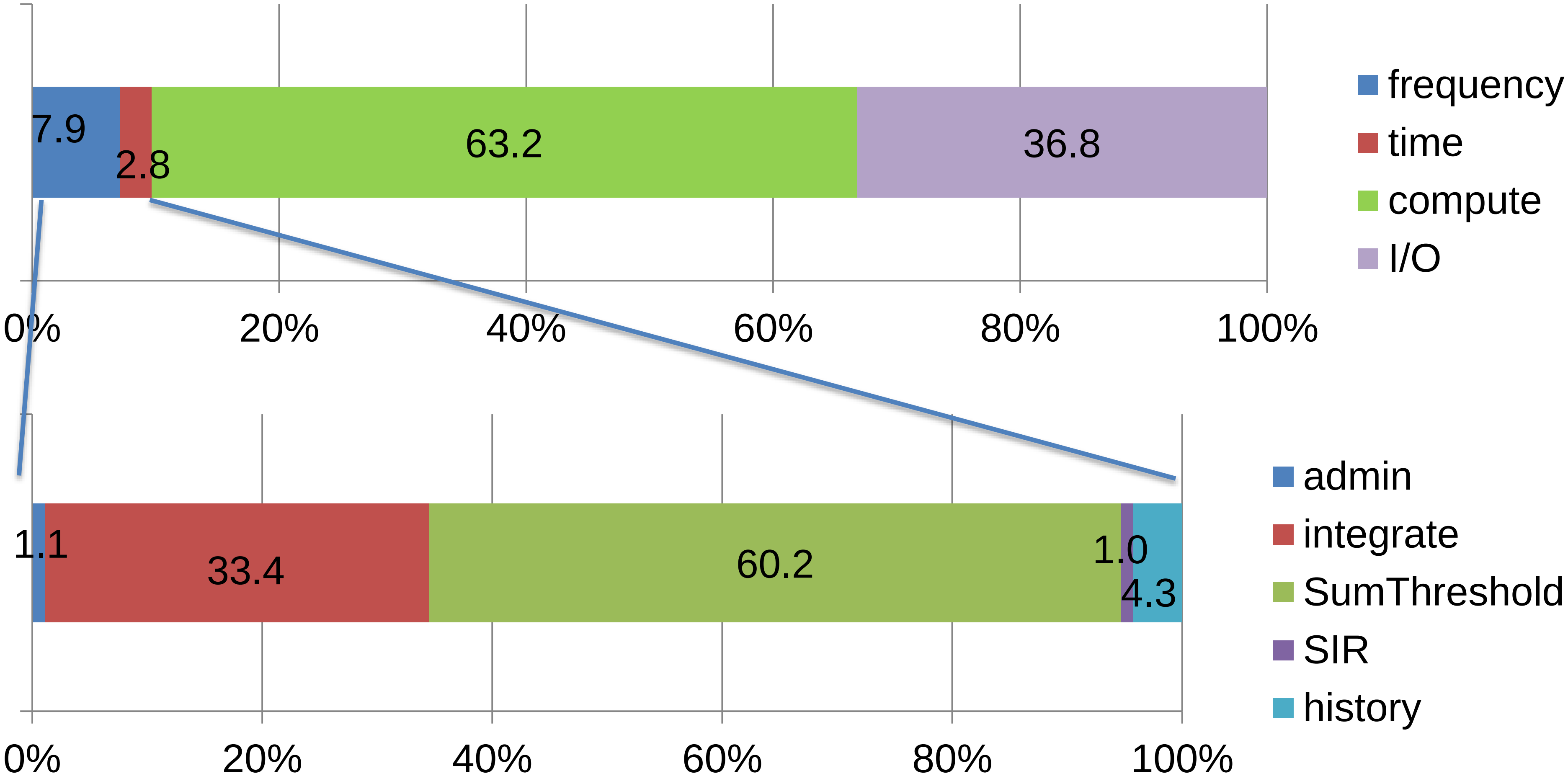
difference



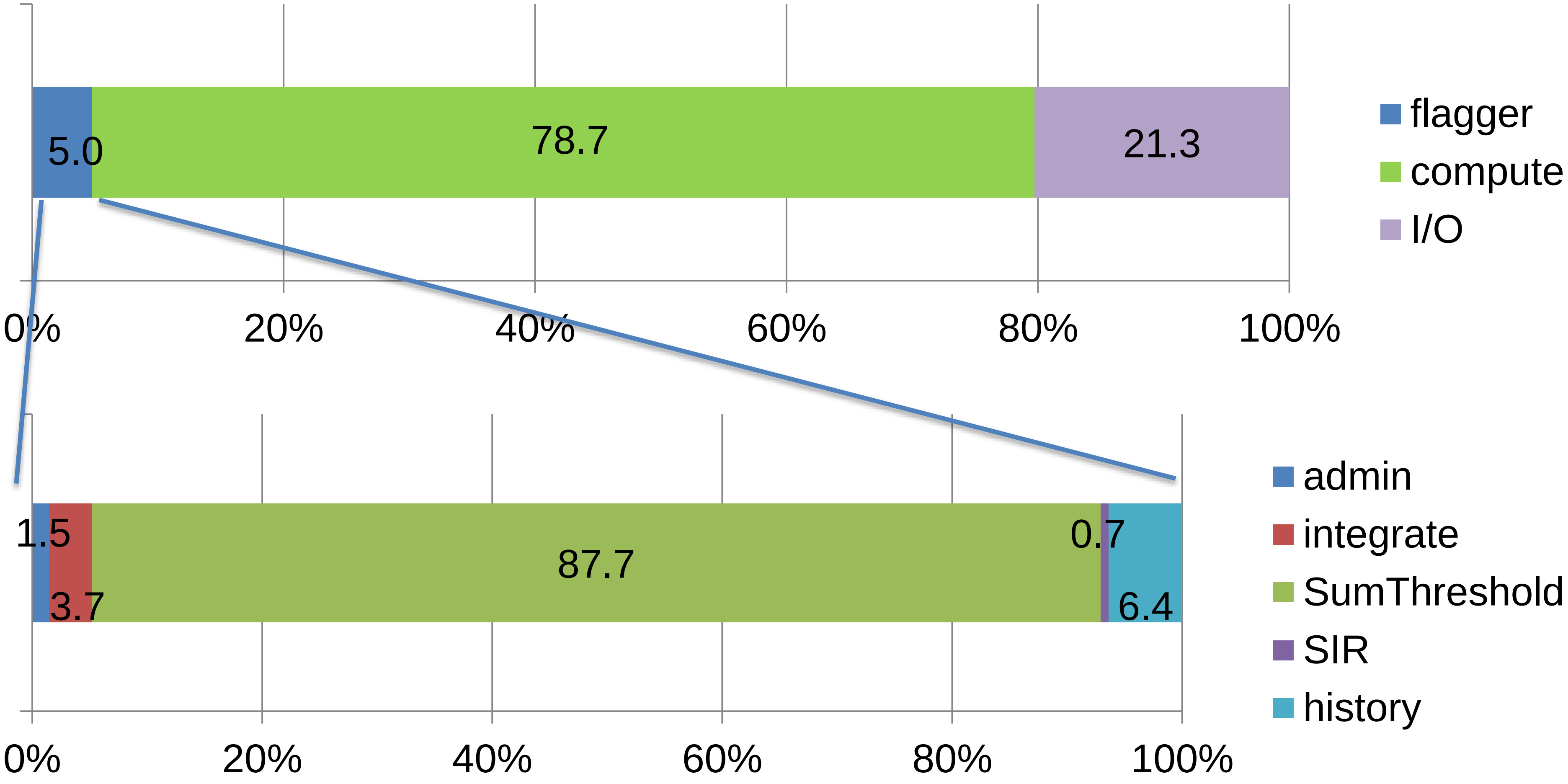
Threshold vs LOF



Performance Breakdown Beam Forming Mode (pre-correlation flagger)



Performance Breakdown Imaging Mode (post-correlation flagger)



Conclusions and next steps

- **Much better than thresholding is possible in real time, even on a distributed system**
- **Scalable: linear computational complexity, adds little overhead**
- **Flexible in storage requirements**
- **SumThreshold originally tested tested on visibilities mostly, now:**
  - **raw voltages, pre-correlation, post-correlaton, post-beam forming**
- **One robust algorithm for different scales (µs - hours)**

- **Currently working on**
  - **GPU implementation**
  - **Commissioning**
  - **Performance model to scale to the SKA**
  - **Dome ExaBounds tool for power dissipation**