RFI Rejection for the GMRT using GPUs

Rohini Joshi, Drexel University

Vinay Deshpande, NVIDIA - Pune, India

Collaborators

1. NCRA, India
    - Yashwant Gupta
    - Niruj Ramanujam
2. NVIDIA, India
    - Vinay Deshpande
    - Pradeep Kumar Gupta
3. Acknowledgments
    - Mihir Arjunwadkar, NCRA
    - Kaushal Buch, NCRA-GMRT

Outline

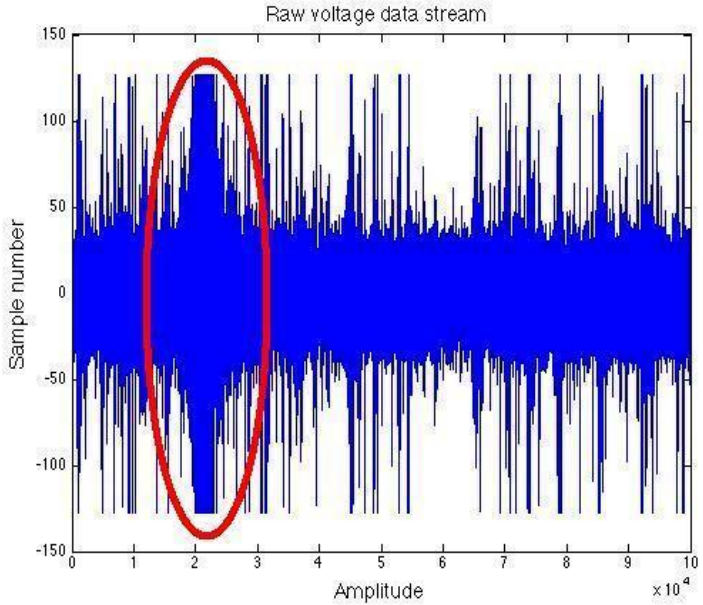
- Introduction
  - What is GMRT?
  - Why RFI rejection?
- Algorithm
- Implementation
- Results
- Future Plans

Giant Metrewave Radio Telescope (GMRT), India

- The GMRT is a low frequency interferometric array having 30 antennas, each of diameter 45m, with a collecting area of 60,750 sq.m.
- It is the world's largest radio telescope at metre wavelengths.
- It is equipped with a CPU based software backend receiver that processes the 60 baseband signals, each of 32 MHz BW, from the 30 antennas. Recently, started upgrading to GPUs.
- Final output is the cross-spectra between every pair of antennas, and a beamformed output from the array

RFI Rejection

- Radio frequency interference (RFI) is the biggest challenge for low frequency telescopes all over the world. RFI rejection is an ongoing research area.
- RFI sources: sparking, Satellite transmission, aircraft navigation signals
- RFI is of 2 types : broadband in frequency, bursty in time domain V/S narrowband in frequency, continuous in time
- Traditional methods of removal: RF filters or manual identification



Raw voltage data plagued with bursty in time domain RFI

How to tackle this problem?

- We are addressing the problem of removing bursty time domain RFI before it corrupts the data downstream – in real time
- Spurious time domain noise (RFI) is statistically modeled as outliers in the normal distribution of the broadband voltage signals from the antennas
- Track the variance of the data to be able to determine statistical outliers
- Robust estimators! – Median Absolute Deviation (MAD)

The Math behind MAD

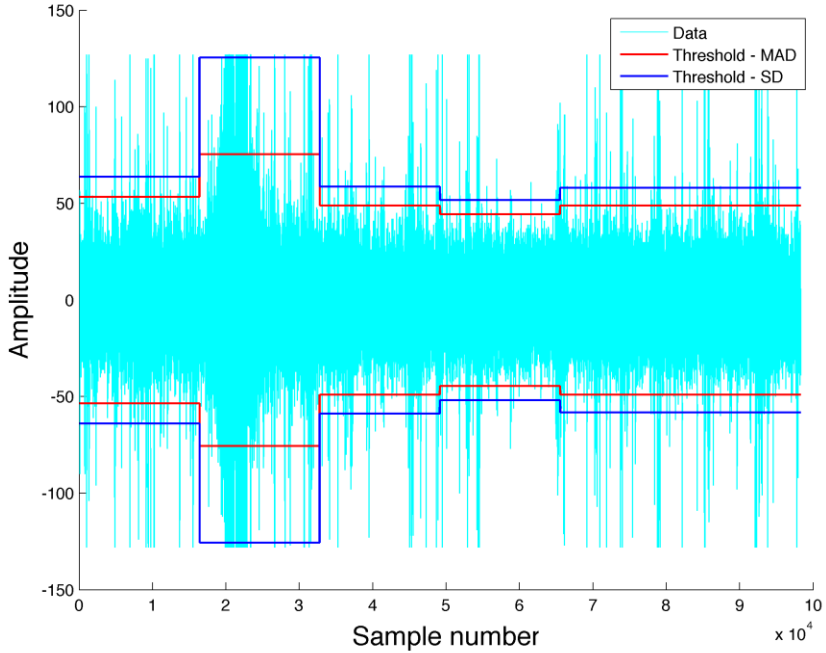
$$MAD_n = b \, med_i \, |x_i - med_j \, x_j|$$

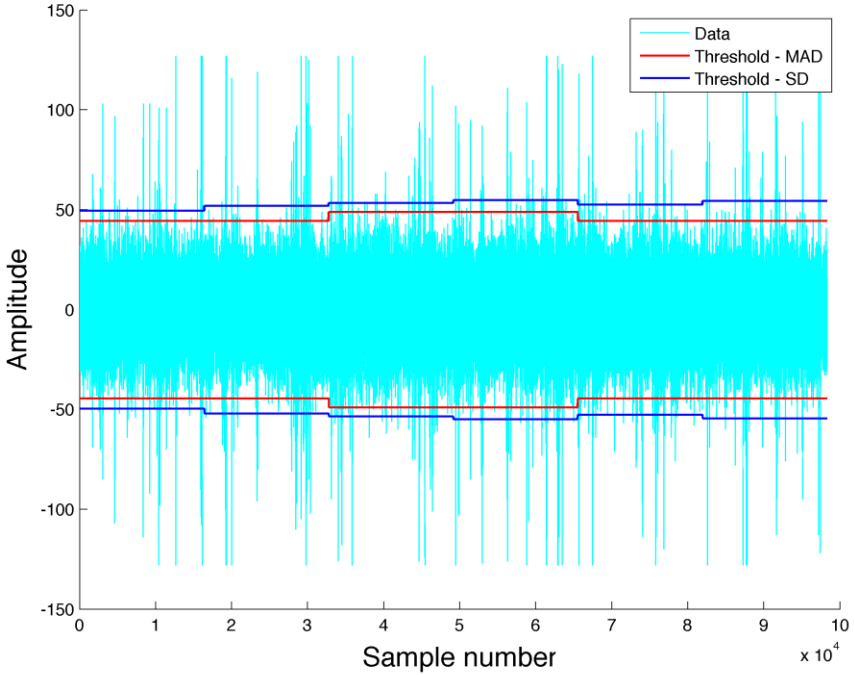
- Estimates the true variance by looking at the distance amongst samples rather than distance from a certain metric
- Set b = 1.4826 to make MAD consistent with standard deviation [1]
- MAD's robustness is quantified by the [1]
  - Breakdown point - 50%
  - Influence function – bounded
  - Efficiency - 37%

MAD filtering – Part I

We would like to identify each data sample as signal VS. RFI

- There may be a wandering baseline raw voltage. So, we must recompute MAD from time to time
- Using windows! Fix a window size and compute MAD for each non-overlapping window
- Set threshold as a multiple of MAD (~3 to 5 times)
- What to do with flagged samples?
  - Simply note them in a flags file
  - Replace them with median, MAD, zero
- A CPU prototype proves it to be effective at removing RFI

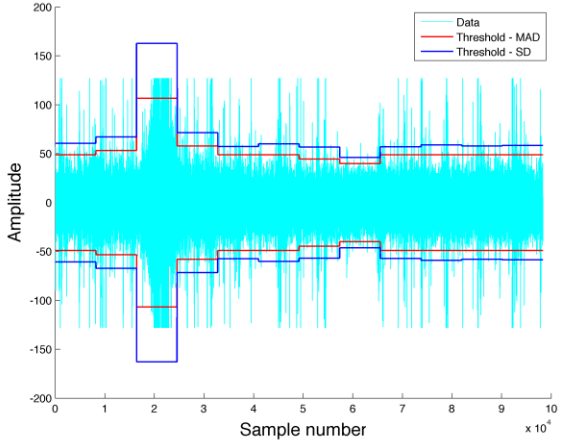


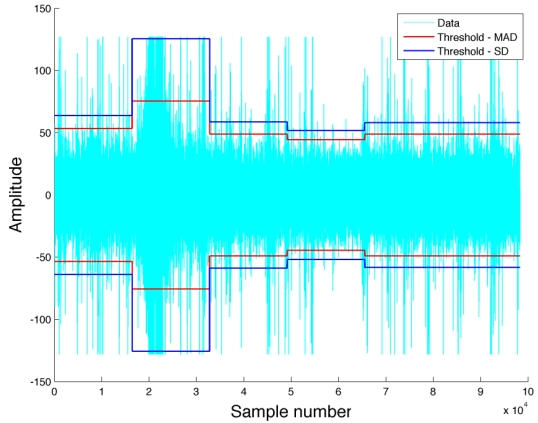


MAD tracks the variance of the data in presence of RFI. In absence of RFI, it is fairly close to the standard deviation

MAD filtering – Part II

- Choice of window size is crucial. Depends on typical scale of RFI





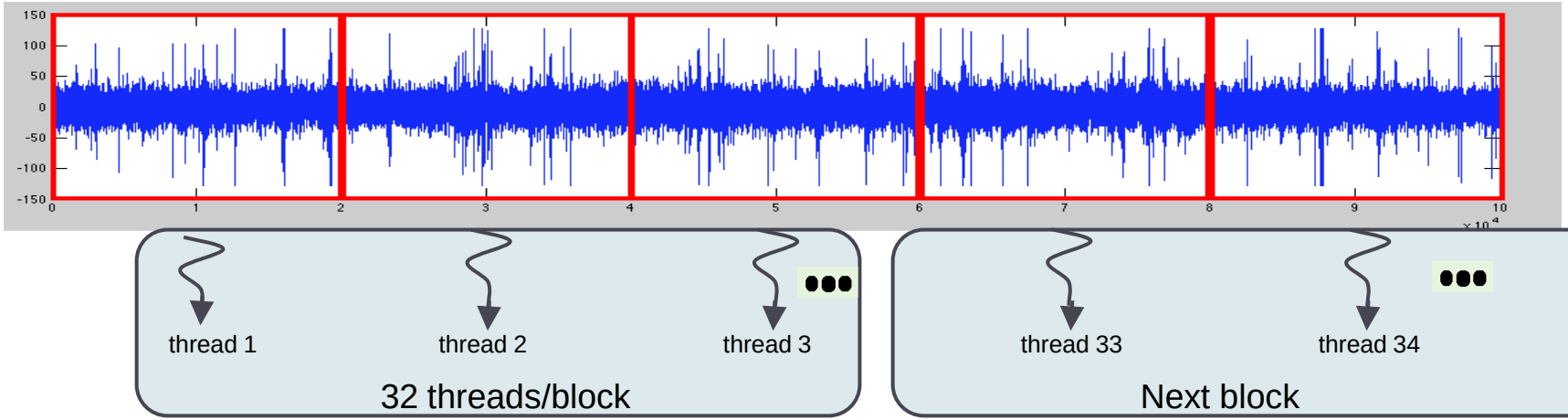
- Too small - RFI will bias the estimator
- Too big – Cannot track the data well
- Two versions of the code for small (~1k samples) and large (~16k samples) window sizes since scale of RFI is unknown

Need for GPUs

- Highly parallel task!
- Time taken by CPUs to perform MAD filtering is an order of magnitude longer than actual length of real time data
- RFI rejection is proposed to be a part of larger computationally intensive correlator pipeline.
- Requirement: Real-time RFI rejection scheme that works on the voltages before correlation stage
- GPU used: Tesla Card C2075 (Fermi Architecture)

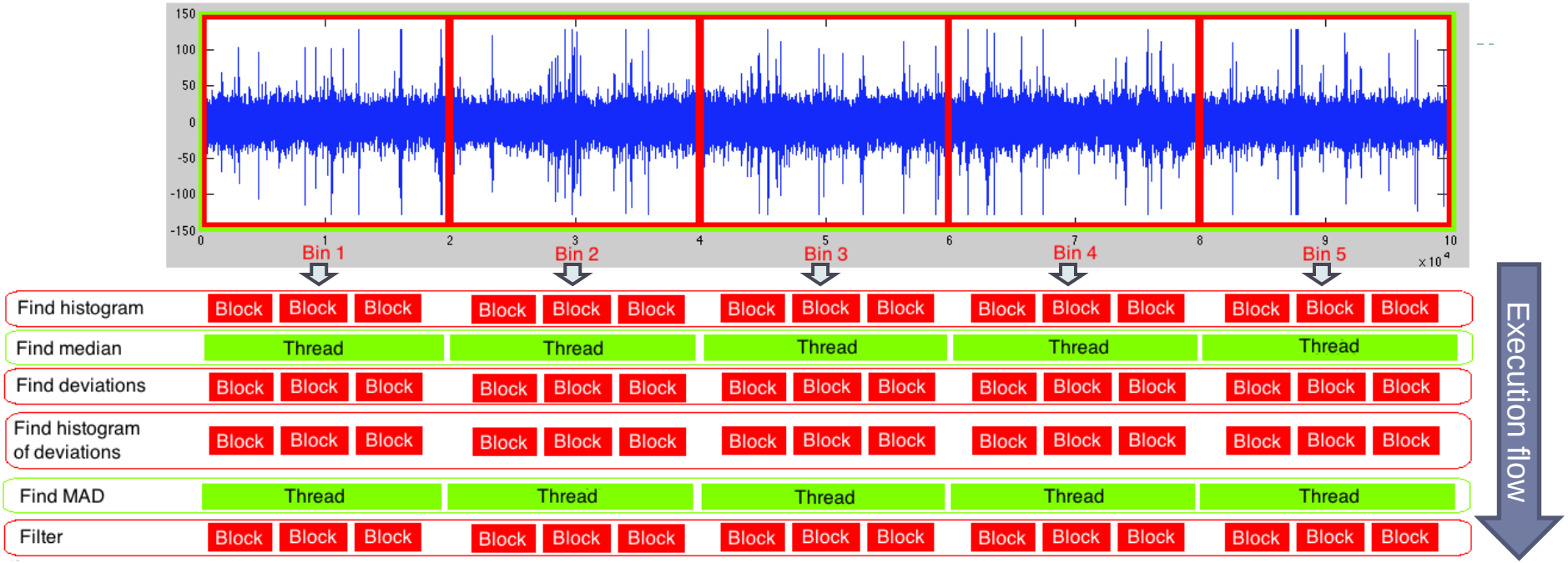
Implementation 1 - Divide the data

- MAD-based filtering is an independent computation across windows
- One thread = one window
- Launch 32 threads/block to reduce divergence
- One kernel = CPU code for MAD filtering
- Branch divergence, memory access not coalesced



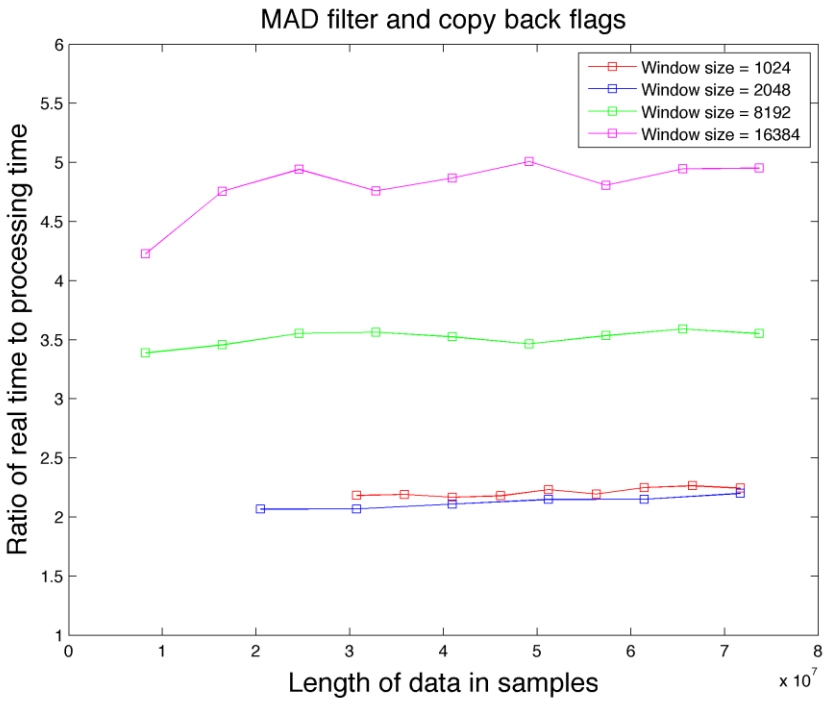
Implementation 2 - Divide the work

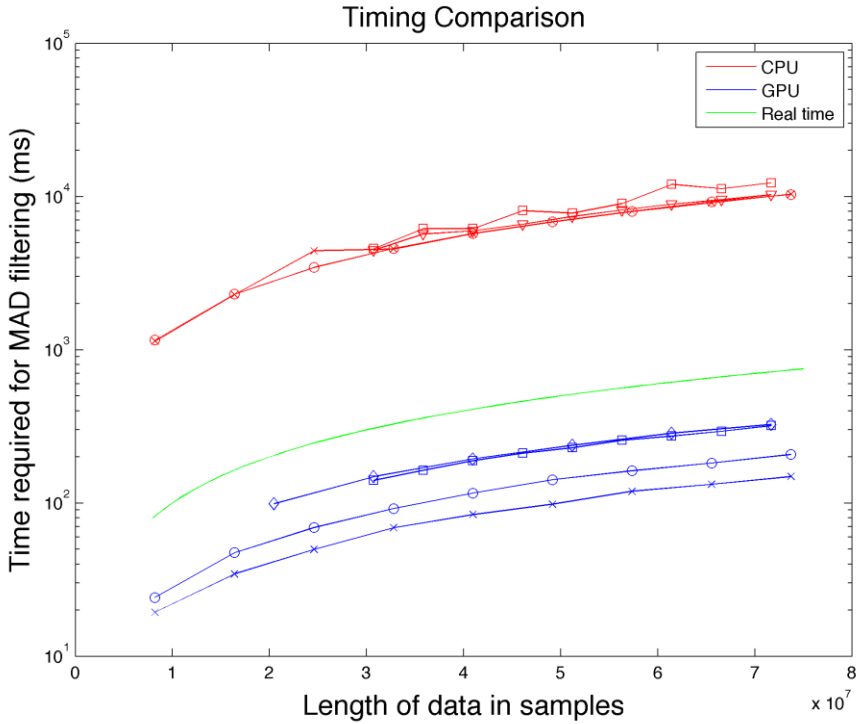
- Larger windows cannot be filtered with one thread - computation becomes very slow
- MAD filter using 6 kernels. Each with unique launch configuration to optimally perform each of the sub-tasks involved in finding MAD
- Median computed using a cumulative sum over the histogram of the window
- Optimized histogram calculation
- Pros: Shared memory utilised, memory coalesced
- Cons: Some kernels have low occupancy



Schematic for Version 2

Results - Real time!





Future Plans

- More efficient ways to find median?
- GMRT upgrade: sampling rate will be 10x current rate. Substantial improvement will be needed!
- Once scaled to work at these data rates, this algorithm will naturally form a part of the development effort towards the Square Kilometre Array (SKA)
- SKA is an international radio telescope to be built starting 2017. Will be the world's largest radio telescope!
- SKA is expected to detect the faintest radio sources to date. To achieve such high sensitivities, it must be equipped with a strong RFI rejection pipeline

References

1. Rousseeuw P. and Croux C. (1993), ‘Alternatives to the Median Absolute Deviation’, J. of Am. Stat. Assoc., 88, 424
2. Croux C. and Rousseeuw P. (1992), ‘Time-Efficient Algorithms for Two Highly Robust Esti- mators of Scale’,Computational Statistics, 1, 411
3. P. A. Fridman "Statistically stable estimates of variance in radio-astronomy as tools for radio-frequency interference mitigation", *Astron. J.*, vol. 135, pp.1810 -1824 2008
4. Hampel, F. R. (1971), ‘A general qualitative definition of robustness’, Ann. Math. Statist. 42, 1887–1896