AUTOMATED TUNING OF RFI IDENTIFICATION AND FLAGGING ALGORITHMS

U.Rau[1] and B.Martins[2]

[1] National Radio Astronomy Observatory, www.nrao.edu
[2] UNISUL, Tubarao, Brazil & Univ. of Virginia, Charlottesville, USA

Radio frequency interference (RFI) affects a significant fraction of data from radio telescopes. Current approaches to identify and eliminate these data require manual labor for tuning the parameters of existing autoflagging algorithms or even marking individual regions of the data on a visual display.

We describe a technique that can help automate the process of identifying and flagging RFI by automating the tuning of parameters of any autoflag algorithm. We present a genetic algorithm that simulates the tuning of these algorithm parameters throughout several generations, taking in no input and producing a set of parameters that can be used to run the existing autoflag algorithms to flag the data. The algorithm evolves parameter sets to optimize a fitness metric based on expected statistical properties of clean data versus data affected by RFI. Figure 1 compares the flags calculated with the default behaviour of an autoflag algorithm on a use case where it is non-optimal and where our algorithm finds a set of best-fit parameters.

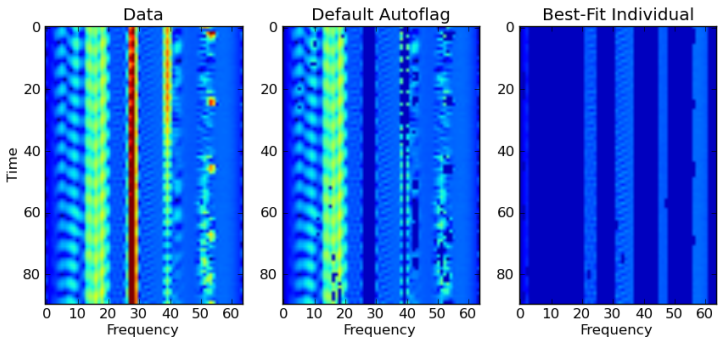

Figure 1: Comparison between applying the default values to the raw data (left) using the tfcrop autoflag algorithm (middle) and using our approach (right). The plots show the amplitude of observed visibilities on a time frequency plane for one baseline. Brighter colors on the left panel represent RFI that needs to be masked out (dark blue in the middle and right panels).

Operationally, this approach requires running the autoflag algorithms themselves a few thousand times on a tiny subset of the data, thus necessitating the use of parallelization to make it usable in practice. Also relevant are strategies of how to partition a large dataset into blocks that are expected to have similar RFI characteristics within each block and how to choose the minimal subset on which to run the tuning algorithm. We demonstrate and discuss the efficiency with which this is possible, making such an approach viable.