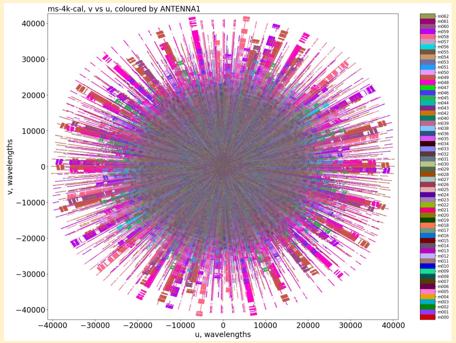
shadeMS: Rapid Plotting Of BIG Radio Interferometry Data

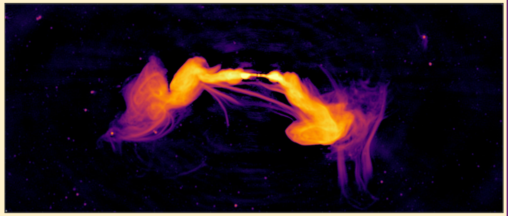
Oleg Smirnov[1,2], Ian Heywood[3,1], Simon Perkins[2], Ruby van Rooyen[2]

[1]Centre for Radio Astronomy Techniques & Technologies (RATT), Rhodes University [2]South African Radio Astronomy Observatory (SARAO) [3]Oxford Astrophysics



No, this is not a rendering of the coronavirus on acid. This is a multi-frequency synthesis *uv*-coverage shadeMS plot for a short observation using the MeerKAT telescope, colour-coded by first antenna in each baseline pair. This plot contains about 20 billion points.

The Context. Radio interferometry data is **big**. For example, MeerKAT spits out billions of raw visibilities per hour. From the end-user's point of view, this data is deeply uninteresting, until it is fed through the blender (aka the Fourier Transform) to make beautiful radio maps such as these.



However, to those of us concerned with making the telescope work, every visibility is sacred and every metadatum great. *Visualizing the visibility data and associated metadata at various stages of calibration reveals instrumental and algorithmic systematics that need to be addressed to make the resulting radio maps both beautiful and useful.* This calls for fast and flexible visualization tools.

The Workhorse. Despite decades of research into advanced visualization techniques and even virtual reality, it is still difficult to beat a good 2D scatter plot for insights into data systematics.

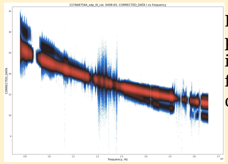
The Problem. A scatter plot with a million points can be a quick way to get valuable insights into your systematics. A scatter plot with 10 billion points is often just an extremely slow way to render an ugly and impenetrable mess.

The Solutions. The Datashader graphics pipeline implements an *aggregation* approach to data rendering. Rather than representing each datum by a point, it bins the data into 2D pixels, and converts the aggregated distributions into colour and alpha. Why this is a superior way to render large datasets is aptly explained here.

Datashader works with standard data structures such as Pandas dataframes and, crucially, Dask arrays. The dask-ms framework (see talk & focus demo by Simon Perkins at this conference) uses Dask to provide access to radio data stored in the standard Measurement Set format, and to process it in a highly parallel fashion (i.e. *orders of magnitude faster*).

Datashader + dask-ms ⇒ shadeMS

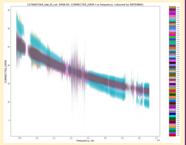
The Philosophy. The shadeMS way is to plot anything versus anything, coloured by anything, all from a simple command-line syntax.



Here is the simplest possible plot: total intensity versus frequency, for a calibrator source.

```
$ shadems data.ms -x FREQ
  -y CORRECTED_DATA:I
```

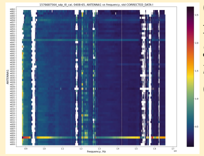
There are ~10 billion data points here: colour indicates the density of points in each pixel. We can clearly see outliers (due to radio frequency interference), but also a systematic causing a deviation from the normal spectrum. We can identify it by using colour as a category instead:



```
$ shadems data.ms -x FREQ
  -y CORRECTED_DATA:I
  -c ANTENNA1 --cnum 64
```

The culprit is clear: confess, antenna m003.

This plot illustrates the **Principle of Maximal Beigeness**: when using colour to categorize, good data is boring, bad data is colourful. Fundamentally, this is because Datashader blends colour and alpha when aggregating bins, so datapoints following the same statistical distribution will tend to blend into a nondescript colour.

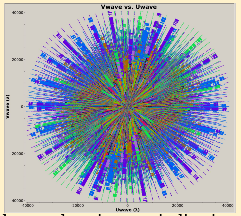


Colour need not only represent density. Here we give shadeMS a different *reduction*, namely standard deviation:

```
$ shadems data.ms -x FREQ
  -y ANTENNA1 --field 0
  -a CORRECTED_DATA:I --ared std
  --cmap pride
```

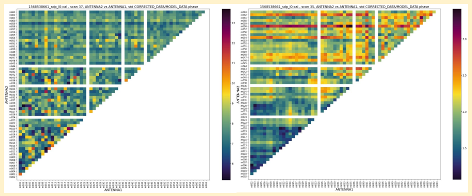
Antenna m003's sins are even clearer -- and now m048 is a suspect too.

Antecedents. CASA includes an interactive tool called plotms, which provides some of the same (anything vs anything coloured by anything) plotting functionality via standard matplotlib scatter plots.



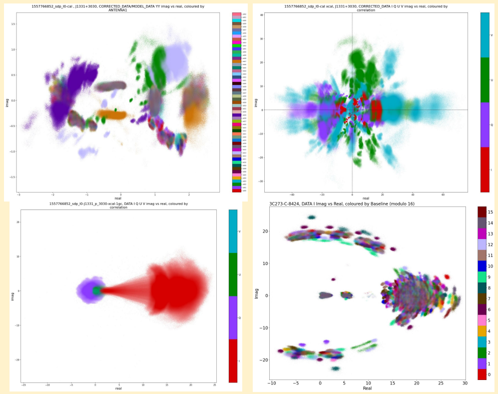
Plotms has a hard limit of ~4 billion visibility points, so we must average the data by a factor of 4 to 8 in order to make a comparison. The "coronavirus" plot demonstrates the overplotting problem inherent to crowded scatter plots: points plotted later mask points plotted previously, so colour is more indicative of data ordering than any real category.

In terms of performance, plotms (CASA 5.5) takes over an hour to render such a plot from 4x or 8x averaged-down data. On the same (64 core) machine, shadeMS (v0.5.0) takes <3m to render native resolution data. *This is largely due to the parallelism automatically leveraged by dask-ms.*



You may not be interested in the ionosphere, but the ionosphere is interested in you. Also, aggregation is not just for scatter plots! Each visibility is a product of a *baseline* formed by two antennas. The plots above aggregate visibilities onto an antenna-vs-antenna (64×64) canvas, and uses standard deviation as the reduction function. Colour now tells us the scatter of visibility data measured by each antenna pair. In the ideal and error-free case, this would be uniform. The left plot is from a normal scan, showing nothing particularly special. The right plot shows increased scatter on higher-numbered antenna pairs, which predominantly form *longer* baselines. We suspect that this is a signature of an ionospheric disturbance passing over the array.

You may not be interested in the polarization either, but it certainly makes for some interesting plots. What is going on here? Answers on a postcard in Discord please.



This research is supported by the South African Research Chairs Initiative of the Department of Science and Technology and National Research Foundation.