LBWG memo 12

Finding the delay calibrator: faster and better

Leah Morabito 21.8.2018

Finding a delay calibrator faster The pipeline parses a file containing a list of possible in-field delay calibrators and then runs through them to find the best one. This is done by phase-shifting to each potential delay calibrator, averaging in time, and then calculating a closure phase statistic (see memo 3). The calibrator source with the best closure phase statistic is then selected as the delay calibrator.

This process was taking a very long time, and we determined the bottleneck to be the phase-shifting and concatenation steps run for each individual source. There are several different ways to parallelise this:

1. `genericpipeline` loop over potential calibrators: parallel over subbands (up to max number of allowed cpus) to phase shift, average, and run station adder. Output subbands are then concatenated.
2. `genericpipeline` loop over potential calibrators: phase shift, average, station add, and run concatenation all in one step.
3. `python parallel`: rather than loop over calibrators, parallelisation the same as in the second case, but each CPU is given a calibrator source to work on, rather than looping over the calibrators. (code slightly modified from N. Jackson)

Results

I tested these three methods for 100 subbands of a 10-minute data set, using 12 CPUs for all methods. For one source, Method 1 takes about 3 minutes per source, Method 2 takes about 18.5 minutes, and Method 3 takes about 22 minutes. However, when scaling up to more sources, Method 1 increases time proportional to N while Method 3 increases time proportional to $\sqrt{N}$. Therefore Method 1 is faster only if there are 7 or fewer potential calibrator sources, while Method 3 is faster if there are more than 7 sources.

The best way to pick a delay calibrator We rely on a closure phase statistic to determine what the best delay calibrator will be – see Memo 3. Here we look at `closure_v3` and `closure_v4`, using the closure phase on the triangle ST001 – DE601 – DE605.

Results: v3 vs. v4

The table above shows that `v3` and `v4` of the closure phase statistic calculations select a different 'best' delay calibrator. The `v3` calculation is a straightforward mean of the gradient of the unwrapped closure phase, i.e., it is the scatter in the local slope of the closure phases – and thus a measure of how coherent the closure phases are. Whether or not this is a good measure of phase coherence depends on how much the data have been averaged in time, as averaging more than the atmospheric coherence time will not be a good measure of coherence. `v4` is a bit more sophisticated, as it considers closure phases averaged with different smoothing lengths, and then performs a polynomial fit on the scatter as a function of the smoothing length. Negative values are possible, and good calibrators will have closure phase statistics very close to zero. In the table below are the closure phase statistics for the 16 potential delay calibrators in P205+55, calculated both ways.

For `closure_v3`, anything with a closure phase statistic of 1.64 or above is random noise, i.e., the source is not coherent on this closure triangle. However, there is not a large difference between 'random noise' and 'noisy' when using this statistic, a problem which is

Table 1: Closure phase statistics for P205+55 potential delay calibrators, calculated via two different methods. In bold are the minimum values for each method.

Source	`closure_v3`	`closure_v4`
ILTJ132737.2+550406.2	**0.1481850174**	0.0242173162
ILTJ133749.7+550102.7	0.4909443174	0.0061813394
ILTJ134934.6+534117.9	0.5838284769	**-0.0267758018**
ILTJ134443.6+550300.5	0.5959868953	0.0851400639
ILTJ134158.5+541524.9	0.8044136569	0.0505633186
ILTJ135146.8+551818.7	1.0296986339	0.163153196
ILTJ133915.6+563829.2	1.1220507325	0.1974670841
ILTJ134255.2+541432.9	1.1250419677	0.5958891771
ILTJ132635.7+551443.0	1.1998326879	0.3625401578
ILTJ134455.0+534829.2	1.4271537355	0.5947044845
ILTJ133534.9+563115.8	1.4487124083	0.5410218828
ILTJ135044.0+544752.8	1.4705579728	0.9209013257
ILTJ135209.6+545835.6	1.6199067816	1.6133934126
ILTJ134442.1+555312.3	1.6445005777	1.5422775854
ILTJ133437.2+563147.5	1.6869923679	1.4906760002
ILTJ133932.8+525748.1	1.6936455001	1.5678865126

alleviated with `closure_v4`, where more closure phase statistic values are closer to zero. That makes it easier to set a threshold when trying to determine if baselines contain coherent signal. However, `closure_v4` selects ILTJ1349+5341 as the best delay calibrator. We saw in Memo 4 that ILTJ1349+5341 has less than half of the integrated flux of ILTJ1327+5504, and much lower signal to noise ratios for individual stations. For this reason, I think we should continue to use `closure_v3` to determine the *best* delay calibrator, and `closure_v4` for other purposes, e.g., testing whether there is coherent signal.

Results: linear vs. circular We can also speed up the selection of an appropriate delay calibrator if we can avoid converting to circular polarisation. I therefore calculated the `closure_v3` closure phase statistic for the calibrators in the previous table, both for linear and circular polarisation. The same ‘best’ delay calibrator is selected regardless of polarisation; therefore in the interests of time we don’t have to convert to circular during the delay calibrator search step.

Table 2: Closure phase statistics for P205+55 potential delay calibrators, calculated for circularly vs. linearly polarised data.

Source	circular (RR,LL)	linear (XX,YY)
ILTJ132737.2+550406.2	0.1481850174	0.1134971515
ILTJ133749.7+550102.7	0.4909443174	0.409029844
ILTJ134934.6+534117.9	0.5838284769	0.609395487
ILTJ134443.6+550300.5	0.5959868953	0.522753873
ILTJ134158.5+541524.9	0.8044136569	0.8278203847
ILTJ135146.8+551818.7	1.0296986339	1.0339262676
ILTJ133915.6+563829.2	1.1220507325	1.1677521711
ILTJ134255.2+541432.9	1.1250419677	1.2341848988
ILTJ132635.7+551443.0	1.1998326879	1.1292844786
ILTJ134455.0+534829.2	1.4271537355	1.3302176983
ILTJ133534.9+563115.8	1.4487124083	1.5158661467
ILTJ135044.0+544752.8	1.4705579728	1.4622615633
ILTJ135209.6+545835.6	1.6199067816	1.7218373238
ILTJ134442.1+555312.3	1.6445005777	1.6675516255
ILTJ133437.2+563147.5	1.6869923679	1.6430704236
ILTJ133932.8+525748.1	1.6936455001	1.6456200025