Visibility Amplitude Scaling: A Comparison of DiFX to the Mark4 Hardware Correlator

Roger Cappallo
MIT Haystack Observatory
2011.2.14

During testing of the DiFX correlation software at Haystack it was noticed that there were systematic differences in the *fourfit*-estimated amplitudes between DiFX and the Mk4 hardware correlator. The sense of the difference was that the DiFX values were always lower, and the magnitude of the differences varied from about 17% to 5% (see Table 1). There was no evidence that the hardware correlator was producing higher SNR results – if anything, the phase residuals over time or frequency were very slightly noisier. Overall, a point-by-point comparison of phase residuals and amplitudes shows a great deal of agreement between output from the two correlators. Clearly the principal differences lie in the scaling of the amplitudes in *fourfit*, which for the case of a hardware correlator can be quite subtle. These test cases, though in some sense non-optimal, provide insight into the complexities that are involved.

Exp. type	**band or pol**	**bit rate Mb/s**	**sample rate MS/s**	**bits / samp**	**DiFX amp**	**Mk4 amp**	**ratio difx/mk4**
UVLBI	LL	2000	64	2	2.25	2.71	0.83* (0.94)
VLBI2010	LL	1000	64	2	6.19	6.75	0.92* (0.95)
"	RR	1000	"	2	5.65	6.28	0.90* (0.94)
K08161 geodesy	X	160	16	1	15.77	16.95	0.93
"	S	96	"	1	7.95	8.47	0.94
T2072	X	80	8	1	34.40	36.17	0.95
"	S	40	"	1	9.54	10.44	0.91
"	X	80	"	1	37.34	39.46	0.95
"	S	40	"	1	13.58	14.21	0.96
"	X	80	"	1	5.29	5.64	0.94
"	S	40	"	1	16.81	17.75	0.95

Table 1 Initial results, warts and all. Values marked by an asterisk are amplitudes prior to LSB bug fix (see text), and in the case of UVLBI, a correction due to the slow bit shift rate; corrected values are in parentheses

Hardware correlation amplitude scale factors in *fourfit*

1. *Van Vleck clipping correction* – for 1 bit data the amplitude is scaled by π/2, for 2 bit data the corresponding factor is 1.13 (Thomson, Moran, & Swenson).
2. *hardware rotator correction* – due to the mean ratio of the 3-level rotator as compared to the ideal sine wave, is π / (4cos π/8) = 0.8501.

3. *fractional bit correction* – since the hardware correlator can only align data, prior to multiplication, to the bit level, there is up to a ± ½ bit error, which causes a phase slope across the video band of up to ± 45°. Averaged over multiple bit shifts during an accumulation period, the loss ends up being $1/(1- \pi^2/288) = 1.0355$.

Of these corrections, only the first applies to DiFX output. The hardware rotator loss is minuscule, as a floating point representation of a sinewave is multiplied times the data values. The fractional bit correction is also much smaller (and presently ignored), since in the FX correlation the video band is filtered into multiple (typically 32) subbands, and each is phase-corrected prior to integration.

Arithmetic corrections -- Both correlators use a system of arithmetic for the correlation, which must be corrected for in order to get normalized correlations.

In the Mk4 correlator, the correlation chip uses a scheme originally devised by Albert Bos, where the high state is given a value of 3x the low state, and products of low x low are ignored. For such a system the optimal sampler threshold is set to 0.91 x the rms voltage. This would result in ~64% of the samples in the low state, and ~36% of the samples in the high state, if the samplers and gain stages were ideally adjusted. In practice this is seldom the case. Thus in the mk4 correlator we accumulate sample state statistics – what fraction of the samples are in each of the 4 possible states. A correction is then applied to the amplitude based upon these sample counts, using an algorithm that takes into account the 2-D Gaussian probability distribution integrated over the non-ideal x-y sample regions. The algorithm linearizes a region around the nominal threshold value of 0.91 σ. The corrections get increasingly inaccurate far away from the nominal values.

DiFX uses floating point arithmetic for the products, and is able to use a more optimal representation of the data values as (-3.336, -1, 1, 3.336). Going along with these values are the optimal sampler levels of ~1.00 σ. There are multiple ways that the DiFX data can be scaled within DiFX, but the best solution for use with *fourfit* appears to be achieved by setting Tsys = 1.0. This is a flag to DiFX to normalize the data by dividing baseline visibilities by the square-root of the product of the autocorrelation visibilities. Any issues having to do with over-representation of sample states will divide out at this point.

Correlation of synthetic data – in order to try to make sense out of the disparate numbers in Table 1, I wrote a small program called *bnoise* to generate Mk5B format files with synthetic correlated data. bnoise generates 2 data streams having Gaussian-distributed values, which have been sampled to 2 bits with a user-specifiable threshold. A known amount of correlated noise is added to each data stream, and there is an option to mix one of the streams with an LO in order to achieve non-zero fringe rates. Test data sets were generated, having a 1% correlation (100 "Whitneys" in *fourfit*) and a 1.0 σ threshold, both with and without

a 50 Hz fringe-rate. Both datasets were correlated in the two correlators (see Table 2).

threshold (σ)	correlation	fringe rate (Hz)	DiFX amp	Mk4 Amp	ratio	corrected mk4 amplitude (see text)
1.00	0.0100	0.0	99.47	109.90*	0.905	99.74
1.00	0.0100	50.0	100.82	106.42	0.947	100.01

Table 2 * - the raw mk4 correlation amplitude with 0 fringe rate was actually 93.42. Since the rotator is fixed, with cos = 1.0 and sin = 0.0, while *fourfit* assumes the rotator is going through all states, a correction factor of 1.176 has been applied.

As can be seen in Table 2, the fringe-fitted DiFX amplitudes are quite good, within about half a percent of the expected value. The Mk4 amplitudes are consistently high, but for easily interpreted reasons. The amplitude correction code in *fourfit* (as well as similar code in *FRNGE* before that) mistakenly applied the fractional bit correction (correction #3 above) twice. This has been known for some time, but since the loss due to the video bandpass shape was **not** being corrected for, the two effects tended to cancel, and the extra factor was left in place. This causes an overestimate of amplitude by 3.55% for every scan using fourfit. For the first example in Table 2, though, there was also a 0 fringe rate, which *fourfit* does not take into account, so the overestimate was a factor of 1.0355^2.

In both mk4 cases in Table 2 there is a slight error due to the sample statistics correction, which loses accuracy away from the ideal threshold of 0.91 σ. Since the synthetic data used a σ = 1.0, the error was an overestimate of 2.76% (found by re-doing the numerical integrations of the bivariate Gaussian probability distribution, using the corrected thresholds). In the last column of Table 2 we have removed these effects from the Mk4 amplitudes, and see excellent agreement with the expected amplitudes.

Analysis – armed with the above background we can start to analyze the data shown in Table 1. The most glaring disagreement is in the UVLBI result: upon close inspections it turned out that there was a bug deep within DiFX (for LSB data only), which caused the upper ~1/8 of the spectrum to have a significantly depressed amplitude. This LSB bug affected both the UVLBI and the VLBI2010 values. After this bug was removed the UVLBI amplitude came up to 0.90, still an outlier.

Upon further examination, it was discovered that the UVLBI scan (which was a Carma-Carma baseline) had a very slow bit shift rate (4.3×10^{-4} shifts/s), despite having a reasonable fringe rate due to the high observing frequency. In addition to the scan-wide fractional bit amplitude correction (#3 above), *fourfit* applies a further correction specific to each accumulation period. This correction attempts to track the variation in the fractional bit amplitude correction, as the mean offset and

shifting rate change during a scan. The specific scan that was used had a model fractional bit offset of 0.41 samples, while the true offset (as determined from the single band delay residual of -2.04 ns) was actually at 0.28 samples. The rate was so small that the bit error stayed at this value for the duration of the 20 second scan. The *fourfit* amplitude correction is based upon the modeled delay, not the delay at which the fringes are found. The result was that *fourfit* applied a correction factor to the mk4 amplitude that was 4.0% too large; after accounting for this fact the ratio in table 1 becomes 0.94.

These changes put most of the amplitude ratios within the range of 0.94 – 0.95, with a few outliers. Most of this low-biased amplitude (3.5%) can be attributed to the effect of the (errant) doubled fractional bit correction in *fourfit*. The remainder (1-2%) of the bias is still unexplained, and probably not worth the time for further investigation, given the excellent agreement when using the synthetic data. The noise about the mean bias could be a combination of inexact matching of data intervals (start and stop times, invalidated data, etc.), and inaccuracies in sample statistic corrections. As mentioned above, though, based upon examination of the fringe plots *there is no reason to think that the lower amplitude actually represents a lower signal-to-noise ratio*.