# Amplitude scaling in General

At the end of the day, a properly calibrated visibility dataset should be an accurate representation of the received signal strength on a given baseline, in Janskys. This can be neatly separated into seven distinct stages, which are often blurred together:

1. Accumulating the unnormalised correlation between a pair of antennas
2. Scaling the unnormalised correlation to correct for the length of integration
3. Correcting for the nominal occupation of sampler levels
4. Correcting for the actual occupation of sampler levels
5. Correcting for quantisation effects (the Van Vleck correction)
6. Scaling the corrected correlation coefficients by system temperature
7. Scaling the corrected correlation coefficients by antenna gain to attain actual flux densities

The first part **has** to be done by the correlator. This is what correlators do! The second part is almost always done online as well. The remaining five steps can then either be done at the correlator, or in offline processing. With DiFX, it is possible to do as few as the first two steps or as many as all the steps (with quite a bit of “estimation”). However, it is generally accepted that the final two steps should be done using measured system temperatures and tabulated gain curves (by e.g., AIPS task APCAL), and thus should not be done online by the correlator.

The level of correction done online by DiFX is controlled by the TSYS entry in the DATASTREAM table entries of the .input file. If TSYS < = 0, only the first two steps are done. If TSYS=1.0, all steps excepting the last two are done. If TSYS = (some nominal tsys value), then all steps except the last are performed. This was originally the default (and indeed only, in early versions) behaviour in DiFX, since it was what the old LBA hardware correlator did. It has the advantage of allowing quick-look processing of the data without any further calibration. However, it also means that the a priori values of system temperature must be un-applied when measured system temperatures are later applied, and hence this is not widely used outside the LBA.

Each step will now be considered in more detail.

## 1. Forming unnormalised correlation counts

The correlator unpacks quantised data to a floating point representation, fiddles with its phase a bit, FFTs it, and then accumulates. The only step in this process that has any bearing on the mean amplitude is the unpack stage. For one bit data this is easy - you unpack the values to 1 or -1. For two bit data things are a little more complicated. The sampler thresholds are typically not chosen so that all four levels are equally occupied (“sampler stats” of 0.25, 0.25, 0.25, 0.25) - that would be inefficient for Gaussian-distributed data. The most common scheme in use aims for sampler statistics of 0.17, 0.33, 0.33, 0.17 for the high negative, low negative, low positive and high positive states respectively. In this scheme, the ratio of a “high” value to a “low” value should be 3.336 to incur the smallest quantisation error. This is what is used for VLBA data, although currently LBA data uses a ratio of 3.0 to match prior practice. The remainder of this discussion assumes the ratio of 3.336. So, nominal unpack values might be -3.336, -1, 1, 3.336. This is what was used for the VLBA hardware correlator. But its important to remember that the **ratio** of the high to low values is important, not the absolute scale at this stage. For data sampled with three or more bits per sample the difference in quantization noise between optimal reconstructed values and a linear scale is so small that it is expected that linear mappings will always be used.

Some useful extra info can be found in Leonia Kogan's memos on the VLBA hardware correlator at http://www.vlba.nrao.edu/memos/sci/, in particular see http://www.vlba.nrao.edu/memos/sci/sci12memo.pdf.

## 2. Scaling the unnormalised correlation to correct for the length of integration

At the end of one accumulation period the correlator calculates the number of valid samples accumulated and divides the visibilities by this number. Thus, the mean value of an autocorrelation at this stage should be the expectation value of a sample squared. In the scheme outlined above, this is equal to 0.17*-3.336*-3.336 + 0.33*-1*-1 + 0.33*1*1 + 0.17*3.336*3.336 = 4.444. Thus, after this stage, the mean values of the autocorrelations will be 4.444, if the sampler level occupation was exactly as planned. The cross correlations will be some value less than 4.444 – they have been attenuated by the system temperature noise and (in the low correlation regime, as is typical) the attenuation due to quantisation. At this stage, when going the DiFX format output → difx2fits → FITS-IDI route, the correlator is done, and writes out a DiFX format record.

This step is skipped in DiFX when TSYS>0 - as we shall see, the division by autocorrelations later on makes it redundant, since this same correction is applied to both autocorrelations and cross correlations

## 3. Correcting for the nominal occupation of sampler levels, and for the unpack values chosen

This is where things start to get muddled. In the existing VLBA infrastructure, FITLD applies part of this correction when DIGICOR is turned on, and the rest is applied by a fudge factor in difx2fits. This mirrors what happened for hardware correlator FITS-IDI files. Now, we could get rid of the scaling in both difx2fits and FITLD by changing the unpack values to 0.474 and 1.583, which would make the expectation value for the power of a single sample equal to 1, removing the need for later scaling. Or the raw visibilities could be divided by the number of samples * 4.444, which would achieve the same thing. The trouble is, this requires an update to FITLD, so it may not be practical in the near term. Also, at some level this is dependent on the assumptions about sampler stats, and while 0.17/0.33/0.33/0.17 is pretty standard, it is still just an assumption. Although, as we will see later, it will all come out in the wash when the “actual” autocorrelation corrections are applied.

This step is skipped in DiFX when TSYS>0 - as we shall see, the division by autocorrelations later on makes it redundant, since this same correction is applied to both autocorrelations and cross correlations

## 4. Correcting for the actual occupation of sampler levels

So, at this stage we have autocorrelations that are equal to 1, and cross correlations that are equal to the correlation coefficients - **IF AND ONLY IF THE SAMPLER STATS WERE PERFECT**. Of course, this is not the case. (For DiFX when TSYS > 0.0, both auto and cross correlations have not been divided by a large constant - but their ratios are still the same). So a correction is needed to account for this. Online in the correlator (TSYS > 0.0) this is done by averaging across the autocorrelation bands, and dividing each auto- and cross-correlation by the geometric mean of the contributing band averages. After this step, the average value of an autocorrelation is by definition equal to 1, and the cross correlations are now perfect correlation coefficients. Offline (TSYS < = 0.0) this step is performed by ACCOR in AIPS. It does essentially the same thing. The advantage of the offline route is that transparency of what has been applied is maintained, and the option remains to fiddle with the data at finer granularity - eg you could subdivide a band into smaller chunks and perform this function on each chunk – assuming you had fine-grained TSYS measurements to apply in the final step.

This step is the saving grace that sweeps any early mistakes under the carpet. Even if the correction for nominal sampler occupation was not made at all and the auto and cross correlations were too high by a factor of 4.4, they are both too high by the same factor, and this step will fix the problem. Still, thats no excuse not to try and get things right the first time!

### 5. The Van Vleck correction

This is an upscaling which takes care of the mean decorrelation due to the coarse sampling. For 2 bit sampling in the low correlation limit, it is equal to 1/0.88. DiFX will apply this online when TSYS>0 - it cannot currently be disentangled from the autocorrelation correction. Until recently, online application of this correction was affected by a bug where autocorrelations were also scaled by 1/0.88, despite being in the high correlation limit where the correct correction is unity. DiFX's online correction is only perfect for 1 bit data correlated with other 1 bit data, and 2 bit data with 2 bit data. This is another reason why the offline route is safer in general. It is my understanding that it is DIGICOR in FITLD that applies these corrections in the offline route.

### 6./7. Scaling by measured system temperature/gain

This is always done offline using [ANTAB and] APCAL in AIPS. If a priori values were applied at the correlator (TSYS > 1, equal to some nominal value for the telescope) then the measured TSYS values must be divided by the a priori values. This also screws up the weighting of the data, since the data are not weighted by TSYS at the correlator. This is an oversight that could be fixed, and will be if deemed important enough. However, for now it is just another reason why setting TSYS to 0 is the best approach.

## GOTCHAS

DIGICOR in AIPS may actually check the array name and refuse to function correctly for non-VLBA data. I'm going to look into this.

Walter A.:
Just checked FITLD and it checks if the array name is VLBA. If it isn't, it will set DELCOR = .FALSE. and DOCORR = -1. This means it switches the corrections off! So DiFX has to declare every array to be VLBA unless we change FITLD.
I suggest to change FITLD's behaviour for the case (CORREL.EQ.'DIFX') to be the same as array VLBA (plus CORREL .EQ. 'VLBA' or 'DiFX').

Walter B.:
FITLD does this now. Just use 31Dec09 or later to get this behavior