0 Scope

Short note on definition of simple implementation of determination of offset in position for peaking up in flight.

1 Simulator

I've written set of IDL programs simulates chop mode and placing a single mode Gaussian response anywhere in a 2-D "map" of detector positions on the sky. Fig 1 shows the map of detector positions in arcsec for the PSW 250 µm array. Also shown is a contour plot of a series of positions of a single mode Gaussian generated at 250 µm for an f/8.68 beam – i.e. the equivalent of the detector response to a point source. Fig 2 shows a close of the central detector where we suppose that the source has been rastered across the nominal position of the central pixel with some error applied to nominal pointing direction to simulate the satellite pointing inaccuracy.

At each position of the source it is further supposed that the source is chopped on and off the detector (fig 3 shows an example) and the detector has a 5 Hz RC type time response. At each on and off position a single sample is taken sufficiently after the mechanism movement to allow the signal from the detector to stabilise (points in fig 3). A five second integration (10 chop cyles at 2 Hz chop frequency) is taken at each raster point. The signal is recovered by simply subtracting the off samples from the on samples and summing all ten values. The chop simulator accounts for a DC signal level (to simulate the telescope); a source signal level that is modulated; an NEP and a detector response frequency that is taken as the bandwidth of the system. No electrical filter is included in the simulation although 1/f noise is added in with a knee frequency at 0.5 Hz this is not really filtered properly. The simulation is, therefore, a worst-case treatment of any 1/f noise component. The source and the NEP are specified in relation to the signal from the telescope. There for a signal of "1.0" represents a source with the same equivalent brightness as the telescope. Again, in order to make this a worse case, I take the NEP as 1e-3 W rt(Hz) - about 10x greater than it should be.

Fig 4 shows an example of the signal as a function of offset position from the nominal in "x" and "y" position. The actual offset is taken as being the position of the maximum signal.

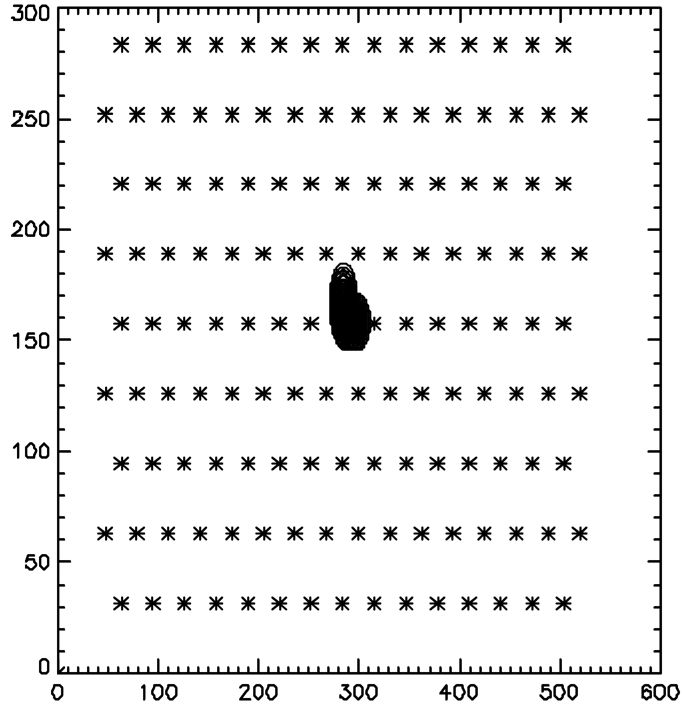

Fig 1: PSW detector positions as seen on the sky and a simulated single mode "source" rastered in "x" and "y" cross form.

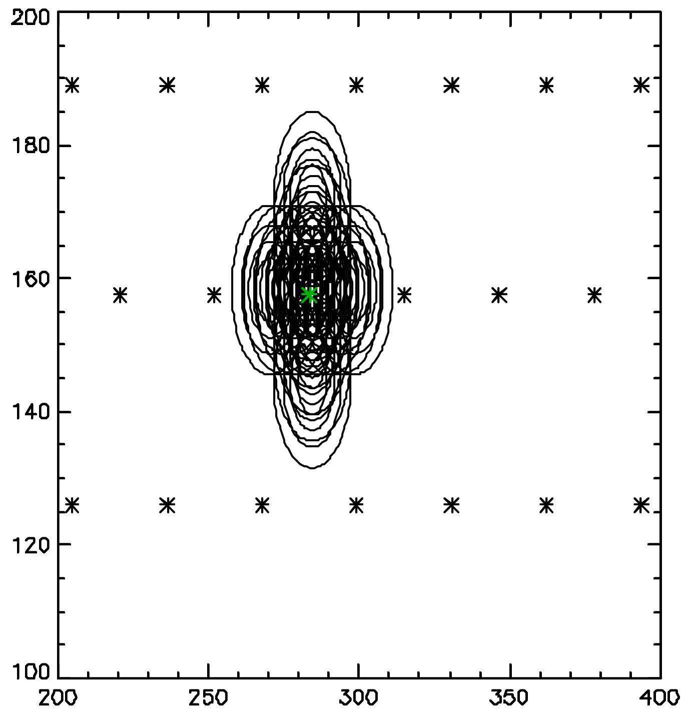

Fig 2: As fig 1 but blown up around the central pixel (green star) and with some ratser positions left out for clarity.

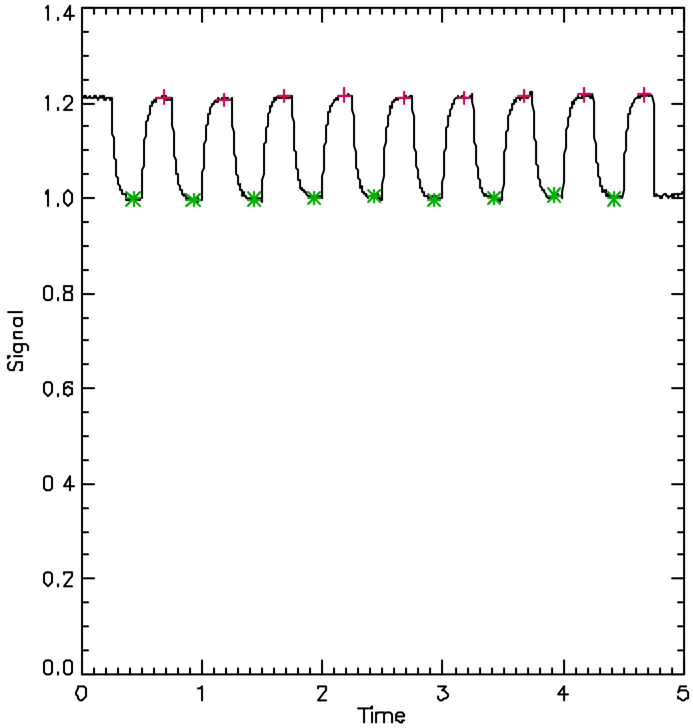

Fig 3: Example of chopped signal timeline – sampling at 400 Hz is shown as the line – sub-sampling of the chopped signal at 4 Hz is represented by the off (green star) and on (red cross) signal values.

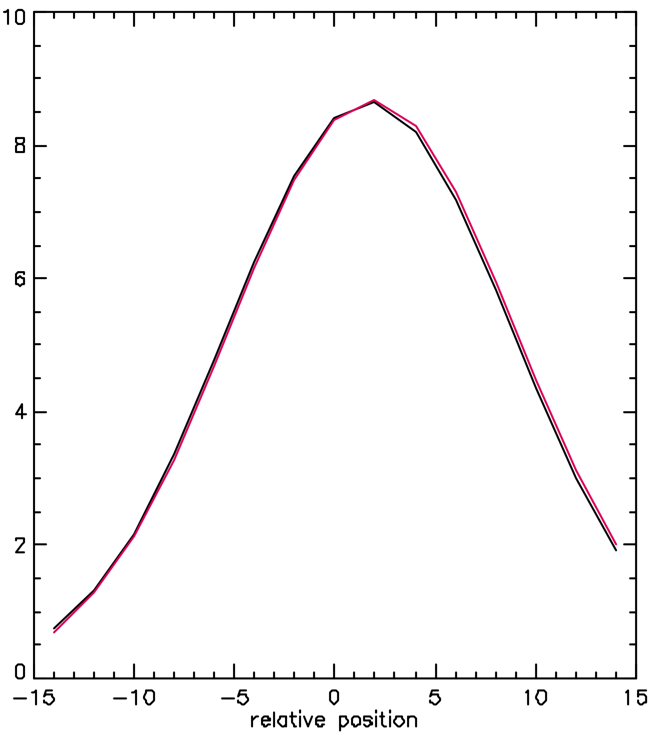

Fig 4: Example of net signal at each offset position from nominal pointing direction (with added error position). The "x" scan is in red – the "y" in black. In this case the x and y error positions were nearly the same by chance only.

2 Results

I started with a source of unit intensity and checked to see what the maximum raster step size was that would achieve the required residual accuracy (<1.5 arcsec 1-sigma). I ran the simulation for 20 randomly selected pointed error positions. I found that for a 1-sigma pointing error of 3.7 arcsec I needed 2 arcsec step size. I took fifteen steps of 2 arcsec in each of x and y. The resulting recovered pointing error and the input error pointing are plotted together in fig 5. We can see here the effect of the 2 arcsec sampling. Fig 6 shows the residual error between actual and recovered positions – i.e. that pointing error we will be left with after the peak up procedure. The standard deviation here is just over 1 arcsec.

As a further check on the robustness of the algorthim I reran the simulation with a source signal 1/10th of the telescope (still with the same noise). An example chop sequence for one end of the raster (i.e. v. low signal) is shown in fig 7 – note the 1/f component seen as pseudo sinusoidal low frequency modulation. The extracted residual positions for the 20 runs are shown in fig 8 – again the standard deviation here is ~1 arcsec

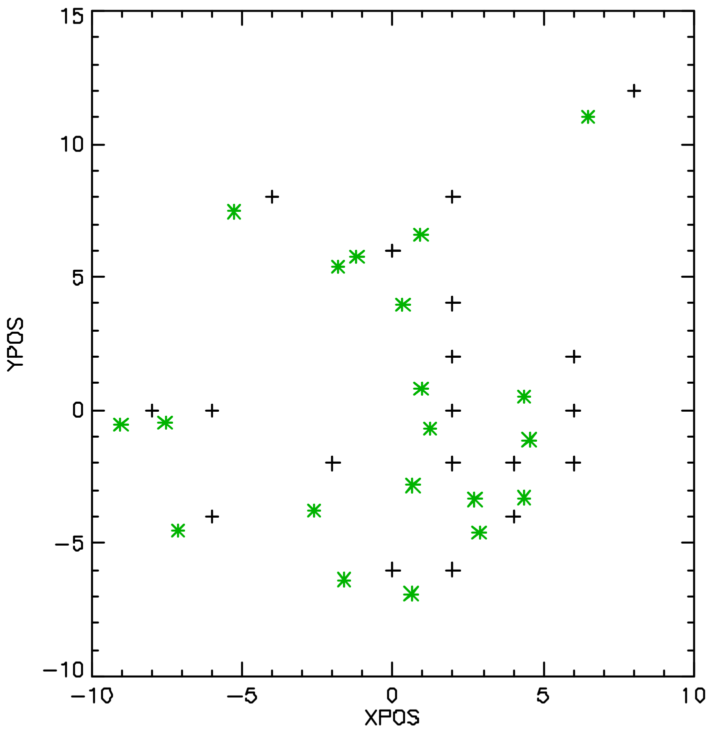

Fig 5: Extracted (black crosses) and actual (green stars) pointing errors for 20 simulator runs with source strength equal to the telescope.

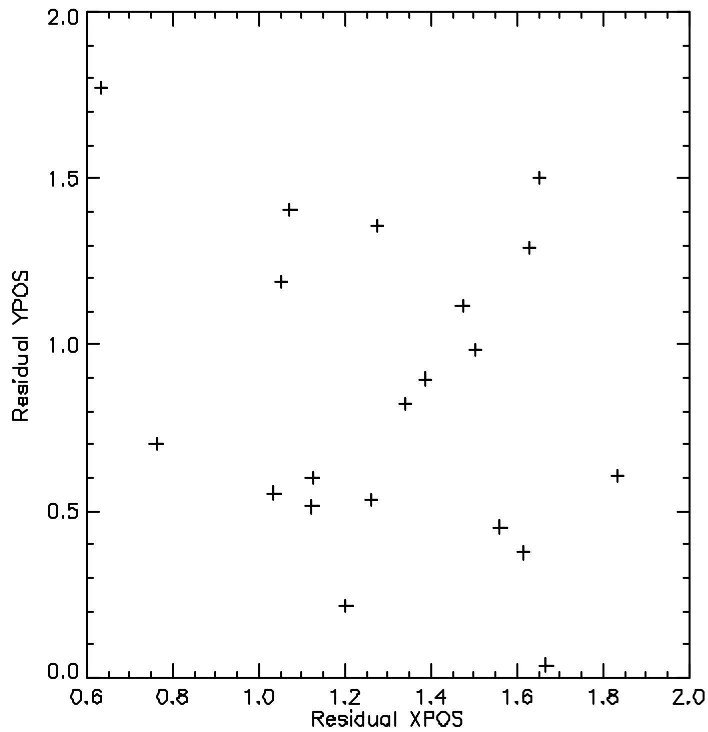

Fig 6: Residual between derived and actual error positions for the same data as in fig 5. This gives the net pointing error that will be achieved following peakup.

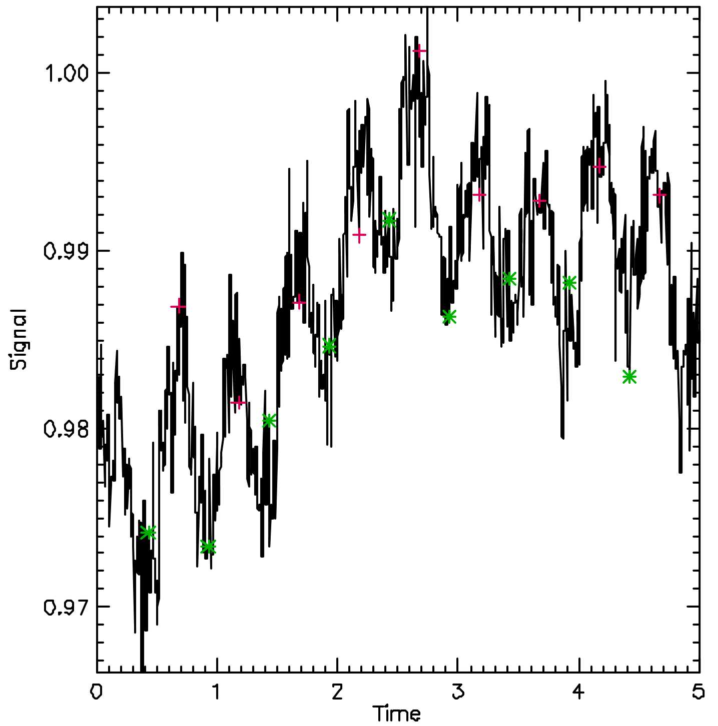

Fig 7: Chop trace (as per fig 3) for a 0.1 intensity source at one end of the raster.

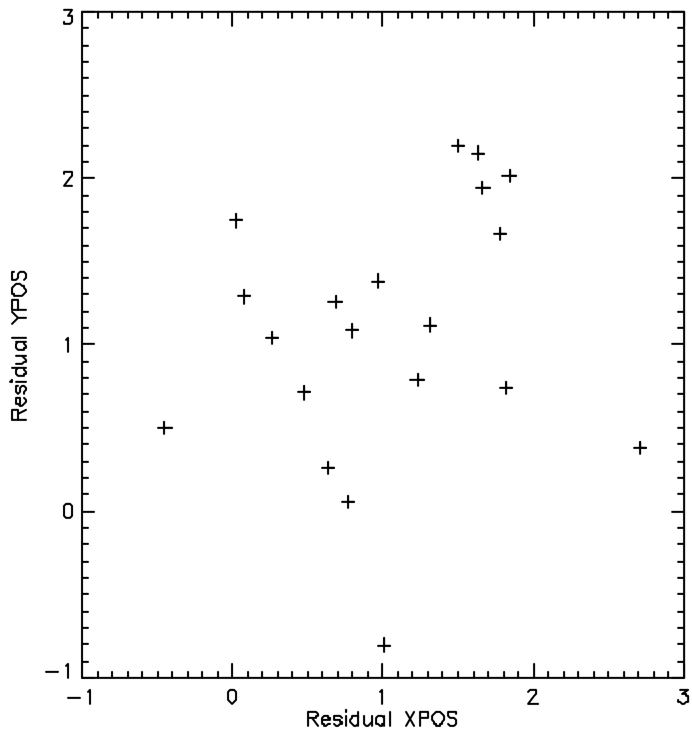

Fig 8: Residual position error for 20 simulator runs with 0.1 intensity source.

3 Conclusions

A robust algorithm can be established for finding the pointing offset in flight by taking a cross raster across the central pixel of the PSW array and chopping the source on and off the pixel at each position. Only the central pixel signal is required so the chop distance can be the minimum required to move the source off the pixel – i.e. it is not necessary to place it on another pixel nor to have a chop distance that finds a pixel on another array. Initial simulations show that a 2 arcsec step size with 15 positions in each direction will be adequate to reduce the pointing error to within the required <1.5 arcsec 1-sigma. This method appears robust against 1/f noise and still works with sources down to 0.1 of the equivalent intensity of the telescope.

Appendix

A possible algorithm for the OBS is as follows:

Data:
- Offset positions from (0,0) in a table OFFPOS
- Number of positions in OFFPOS
- Distance to chop the BSM CHOPDIST – only applied to cop position in OFFPOS table
- Location of sampled detector DETLOC within PSW frame
- Delay time SAMPTIME to wait after movement of mechanism before sample taken

Variables:
- ONS – on source signal
- OFS – off source signal
- CURSIG – current value of signal
- PREVSIG – signal from previous position
- POS – offset position at maximum of signal

Do once for chop direction and once for jiggle direction

```
PREVSIG = 0
do while i < NPOSITIONS
    do while j<10
        set BSM to OFFPOS(i)
        wait SAMPTIME
        Sample PSW
        Extract value from DETLOC
        Store value as ONS
        Set BSM to OFFPOS(i)+CHOPDIST
        Wait SAMPTIME
        Sample PSW
        Extract value from DETLOC
        Store value as OFS
        CURSIG = CURSIG + (ONS-OFS)
        j=j+1
   end while
   if CURSIG > PREVSIG then POS = OFFPOS(i)
   i=i+1
end while
```

Store POS as offset value for this axis