Simulations of the Point-Source AOT sensitivity

M. Sauvage

Document change record

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>October 5, 2009</td>
<td>Release of the document</td>
</tr>
</tbody>
</table>

Reference documents

<table>
<thead>
<tr>
<th>Ref.</th>
<th>ID</th>
<th>Version</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>RD1</td>
<td>SAp-PACS-MS-0616-06</td>
<td>1.0</td>
<td>FM Photometer Focal Plane Unit User’s Manual</td>
</tr>
<tr>
<td>RD2</td>
<td>SAp-PACS-MS-0719-09</td>
<td>2.1</td>
<td>1.1.10 &amp; 1.1.16 - Time constants</td>
</tr>
</tbody>
</table>

Contents

1 Introduction 2

2 Principles and input data 2

3 Noise propagation in the data reduction 2
   3.1 Preliminaries .............................................................. 2
   3.2 Step 1 - Averaging the signal by groups of 4 images .......................... 3
   3.3 Step 2 - Differentiating the OFFs from the ONs .................................. 5
   3.4 Step 3 - Averaging per dither position and deglitching .......................... 8
   3.5 Conclusion ......................................................................... 8

4 The actual bolometer response 8

5 Conclusion 13
1 Introduction

In light of the surprisingly low sensitivity in flight of the point-source AOT with respect to HSPOT expectations, I use in-flight calibration measurement to try and understand where possible sensitivity losses could creep-in.

2 Principles and input data

The principles of the simulation are quite simple: I use very long data cubes designed to derive the noise spectral density of the bolometer matrices to simulate a point source observation. There are no sources in these datacubes thus they are ideal to follow the evolution of the noise properties at the different steps of the point-source data reduction. I thus reproduce the different data reduction steps and see their impact on the noise spectral distribution in the resulting data cubes. Comparing this to the assumptions that went into the computation of the HSPOT sensitivity can tell part of the story of sensitivity loss.

There is a second side to the sensitivity: the bolometer response. Again we make some assumptions on the response to derive the sensitivity. With the time constant measurements we can simulate what the effective bolometer response is a function of the data processing. This again will tell us how far we are from our original assumptions.

At the center of the simulation, there obviously is the pipeline processing. In this note I concentrate on the first few steps of the pipeline. These are the following:

1. Average 10 Hz images by chopper plateau. The baseline is that a chopper plateau is 4 images long.
2. Subtract the OFF chopper position images from the ON chopper position images.
3. Average all images obtained on a single dither position. We usually have 25 of those, corresponding to the 25 chopper cycles commanded per dither position. At that stage deglitching is performed through a single pass of \( n\sigma \) clipping.

In this note I am first going to study the noise property propagation in the above pipeline steps, and then study the effective response.

3 Noise propagation in the data reduction

3.1 Preliminaries

Let us first start by showing what the noise spectral density (hereafter NSD) looks like, and give some properties of the NSD that will allow some intuitive understanding of what is going on in the data reduction. In figure 1 I show the NSD of matrix 5 and 10 obtained in direct mode with the blue filter for the nominal bias values (2.6 V on the blue side, and 2.0 V on the red side). The NSD depends very little on the input flux and is identical in the blue and green photometer bands.

These NSDs are characterized by the dominance of the \( 1/f \) noise, in fact, the NSD goes almost exactly as \( f^{-1/2} \) for the whole sampled range. We have shown during the ILT that when we sample the data at 40 Hz, we start to see the famous knee corresponding to the transition from pink noise (\( f^{-1/2} \)) to white noise (\( f^0 \)). The comparison of the un-deglitched data (blue curves) to the deglitched ones (cyan curves) show the impressive contribution of glitches above 0.1 Hz. The downward turn in the blue curves observed around 3 Hz is likely due to the intrinsic bolometer bandpass that starts to cut higher frequencies.

The relation of the NSD to more familiar quantities is this one:

\[
\sigma^2 = \kappa \int_{f_0}^{f_c} \text{NSD}(f')^2 df'
\]  

where \( f_0 \) is the smallest frequency that is sampled by our data (set by the length of the observation) and \( f_c \) is the highest frequency sampled by our data, i.e. half of the image frequency, and \( \kappa \) is a constant that essentially
depends on the normalization factors in the fourier transform used to compute the NSD. $\kappa$ does not depend on the actual noise probability distribution. Using the \texttt{RandomGauss}, \texttt{RandomUniform} and \texttt{RandomPoisson} random generators of \texttt{hipe} I've computed that for our method of computing the NSD $\kappa \approx 1.27$ (there is a high chance that this is some function of $\pi$).

Knowing this it becomes more intuitive to guess what should happen to the NSD when we apply some classical data reduction steps such as averaging the images by groups of $n$ or differentiating them.

To start with simple things, let us assume that we have a signal cube containing only white noise, i.e. its NSD is flat over the full sampling range. Averaging this cube by groups of $n$ images decreases the r.m.s. of the signal by a factor $\sqrt{n}$. What does it mean for the the NSD? Averaging our cube by groups of $n$ images means that we have decreased the sampling frequency by a factor $n$, and thus the interval over which the NSD is defined has shrunk by the same factor. Therefore, in order to observe the expected decrease by a factor $n$ of the variance of the signal, the NSD in the averaged-by-$n$ signal has to remain at the same level as the NSD in the original signal.

Similarly we know that when we make differences by pairs in a signal that contains only white noise, the r.m.s. is increased by $\sqrt{2}$. Making differences by pairs means that we decrease the sampling frequency of the signal by a factor of 2. To observe that the variance of the signal (i.e. the r.m.s.$^2$) is multiplied by 2 requires that the NSD in the differentiated signal is twice that of the original signal.

The effect of these manipulations are demonstrated “experimentally” using the \texttt{RandomGauss} function on Figure 2.

### 3.2 Step 1 - Averaging the signal by groups of 4 images

In the first step of the point source pipeline, we take the data and average images per chopper plateau. The default chopper frequency is 1.25 Hz which corresponds to 4 images per chopper plateau. Therefore I have taken the low frequency noise observation and averaged the cube by groups of 4 images. Then I have applied MMT deglitching on this cube, this time with \texttt{nScale=2} to reflect the fact that I have significantly decreased the sampling of the signal. I have computed the NSD of the resulting signal and compare it to the original NSD in Figure 3. I have also applied the same averaging process to the deglitched data to compare the effectiveness.
Figure 2: A simulation of a datacube containing only white noise following a gaussian distribution. I have assumed that the sampling frequency of the cube is 10 Hz. Averaging the original cube by groups of 4 frames leaves the NSD unchanged as the sampling interval has decreased by a factor of 4, implying that the variance has decreased as expected by a factor of 4. Differentiating the original cube by pairs multiplies the NSD by two as the sampling interval is decreased by a factor 2, since the differentiation should multiply the signal variance by 2.
of deglitching before or after averaging. The first point that can be made while looking at this figure is that for multiresolution deglitching, there is no performance difference between doing it before averaging or after. Obviously in a real point source AOR we cannot do it before averaging as the chopper modulation is too strong and to fast.

The second point is that the effect of averaging on a \( 1/f \) noise is rather similar to the white noise example: we observe very little change in the NSD apart from a reduction by a factor of 4 or the sampled frequency range. This does not mean that the variance has been divided by 4 since in this case, it is dominated by the low frequencies that are present with identical power in the averaged signal.

Finally I have used a simulation of a pink noise realization (NSD \( \propto f^{-1/2} \)) to investigate the expected effect of averaging (last row of Figure 3). As with the real data, we see that averaging has no effect on the low frequency part of the NSD. The high frequency is apparently affected: the NSD fall is now steeper than \( f^{-1/2} \). This is not observed on the real data. However (1) simulating pink noise is complicated, and (2) the high frequency part of the real data is where we combine the \( f^{-1/2} \) regime, the white noise regime and the effect of the bandpass cutoff so it is hard to comment on the observed differences between data and simulation.

### 3.3 Step 2 - Differentiating the OFFs from the ONs

The second step of the pipeline subtracts each OFF chopper position from the preceding ON position to produce one image per chopper cycle. To simulate that I simply differentiate by pairs the signal that I have created above. In fact, to follow what the point source pipeline does I create two cubes: one is the original 10Hz signal undeglitched, averaged by groups of 4 and then differentiated by pairs, that I then deglitch with the MMT method using nScale=2 (to see how well this data can still be deglitched), and the other is the original 10 Hz signal, MMT-deglitched, averaged by groups of 4, then differentiated by pairs. For clarity of the graphs, I will not show the NSD of the second signal since, as with the previous step, it is identical to the NSD of the first signal. This means that glitches are strong enough that even after averaging by groups of 4 and differentiating by pairs, they still stand out clearly enough from the noise. I thus compute the NSD of the first signal and plot it Figure 4.

The striking feature of this figure is that the NSD is essentially flat. That is what we wanted: the chopping is meant to beat the \( 1/f \) noise and that is exactly what it does. So far so good.

The more interesting bit here is in the number printed in the graphs' legends. Since the NSD is now essentially flat, I have computed it median value. As recalled above, for a white noise it is rather simple to predict what will happen to the noise spectral density as we averaged and differentiate the signal. We see that for the blue matrix M5, the median NSD value is \( 17.4 \times 10^{-6} \text{V}/\sqrt{\text{Hz}} \). This value can be used to compute what would be the white noise spectral distribution at 10 Hz that would result in the same NSD after averaging by 4 and differentiating by pairs: this is simply \( 1/2 \) of it or \( 8.7 \times 10^{-6} \text{V}/\sqrt{\text{Hz}} \).

If we come back to the middle row of Figure 3, we see that this is the median value of the averaged by groups of 4 signal. So interestingly, and probably incidentally, differentiating produces a similar effect on our rather pink noise signal than on a pure white noise signal.

More importantly, our computation of the point-source sensitivity rests on (1) the assumption that the observing strategy will give us at some point a signal that is white noise dominated, and (2) an estimate of the equivalent white noise NSD level in a 10 Hz signal\(^1\). We have traditionally used the measured NSD level at 3 Hz, and this value is given in the legend of Figure 4. We see that for matrix 5 it is \( 6.4 \times 10^{-6} \text{V}/\sqrt{\text{Hz}} \), or 0.74 times lower. For the red matrix 10 we have to compare the value \( 16.4 \times 10^{-6} \text{V}/\sqrt{\text{Hz}} \) to \( 9.8 \times 10^{-6} \text{V}/\sqrt{\text{Hz}} \), i.e. the NSD level at 3 Hz is 0.6 times lower than the equivalent white noise level resulting from the pipeline processing.

What does this mean? It means that our assumption (1) is correct, but that we have underestimated the value of the equivalent white noise level we had to use in our simulation. As the NEP is the NSD divided by the response, the actual NEP will be higher than the prediction by factors that are simply obtained from the ratio of half the median noise level observed after differentiating to the noise level measured in the original data at 3 Hz. Table 1 lists these values for all 10 matrices.

\(^1\)This is not the amount of white noise in our real 10 Hz signal, but rather the amplitude of a white noise that would, after going through the same data reduction steps as the real data, give us the same final noise level. This is quite different.
Figure 3: The blue and red NSDs (matrices 5 and 10) before and after averaging the images by groups of 4 (second row is a zoom of the first row). The most visible curves are the bright green and brown ones that correspond respectively to the original MMT-deglitched and undeglitched NSDs (i.e. the curves of Figure 1). Hidden behind I show the NSD of the signal averaged by groups of 4 images. There are two curves: the dark blue one results from the original undeglitched signal averaged by groups of 4, then deglitched with the MMT method, while the cyan one corresponds to the application of the averaging by groups of 4 on the MMT deglitched signal. They are identical.
Figure 4: The NSD of the original 10 Hz signal, averaged by groups of 4, differentiated by pairs and then deglitched with the MMT. In cyan I report the original 10 Hz deglitched signal NSD. In the legend I list the median value if the noise spectral density for the averaged and differentiated signal, and the value of the noise spectral density at 3 Hz for the original signal.

Table 1: The error ratio between the actually observed effective white noise NSD in point source mode after we averaged frames per plateau and differentiated the result by pairs, and the assumption that went in the sensitivity computation.

<table>
<thead>
<tr>
<th></th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
<th>M8</th>
<th>M9</th>
<th>M10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.33</td>
<td>1.41</td>
<td>1.37</td>
<td>1.39</td>
<td>1.36</td>
<td>1.41</td>
<td>1.31</td>
<td>1.66</td>
<td>1.67</td>
<td></td>
</tr>
</tbody>
</table>
3.4 Step 3 - Averaging per dither position and deglitching

This is where the pipeline creates a single image from the 25 chopper cycles that are obtained on a single dither position. As it is the first time that it works on a reasonable set of images, this is also where deglitching can occur. By default it is done with a single iteration of $n\sigma$ clipping. To simulate this I start again from the 10 Hz low-frequency noise measurement. I create two cubes, one containing the raw un-deglitched signal and one containing the raw MMT-deglitched signal. Both are then averaged by groups of 4 images, and differentiated by pairs. Then I average the resulting cubes by groups of 25 images, performing a single pass of $n\sigma$ clipping this way: I compute the mean and $\sigma$ for each pixel and then recompute the mean rejecting for the sample of 25 values all those that differ by more than $n\sigma$ from the mean. I have explored the effectiveness of $n = 3 - 5$ on the deglitching performance. I compute the NSD of the resulting cubes and plot it in Figure 5.

The NSD now cover a small frequency range as we have decreased the sampling rate again by a factor 25. Two points are worth noting. First, contrary to my expectation, $n\sigma$-clipping is remarkably efficient at getting rid of glitches: $n = 4$ gives a signal were the NSD is almost the same as that measured on the MMT-deglitched data. Second, the level of the NSD after averaging per groups of 25 is the same as what it was before, which is what we expect for a signal dominated by white noise. This is a positive sign as it confirms that the observation and data processing method do indeed get rid of the $1/f$ noise.

Looking at the numbers in the legends, one can even see that the noise level is slightly reduced after the 25 chopper cycles are averaged. This is because the quoted number is the median level, which is influenced by the high frequency range where we do see a slight upturn in Figure 4. This upturn is mostly gone in Figure 5. The reduction is in the 10% range, and does not compensate for the mismatch between the hypothesis used in the sensitivity computation and the actual performance. Nevertheless, since the objective of this note is to compare the measured performance with the prediction, it is worth including this in the comparison. Thus Table ?? lists the ratio between the equivalent white-noise spectral density value (1/2 of the level observed on Figure ?? since at this stage we have only done various averagings and one difference) and the observed noise spectral density at 3 Hz in the original data (the value that is used in the sensitivity computation).

<table>
<thead>
<tr>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
<th>M8</th>
<th>M9</th>
<th>M10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.27</td>
<td>1.34</td>
<td>1.28</td>
<td>1.32</td>
<td>1.30</td>
<td>1.26</td>
<td>1.31</td>
<td>1.33</td>
<td>1.48</td>
<td>1.45</td>
</tr>
</tbody>
</table>

3.5 Conclusion

We see thus see that the improper assumption that we could used the noise spectral density measured at 3 Hz to describe an equivalent white noise affecting our data in the sensitivity estimation leads to an increase of the NEP by a factor 1.3 on the blue side and 1.5 on the red side. Given that the sensitivity is directly proportional to the NEP, these factors should be present when we compare the performance of the point-source AOT to the HSPOT expectations. The reason why the increase is more important on the red side is likely due to the shape of the intrinsic NSD: on the blue side we seem to be reaching a knee at high frequencies, while on the red side, the $f^{-1/2}$ regime is present over the whole spectrum.

4 The actual bolometer response

As mentioned before, the response of the bolometer is another fundamental element of the NEP. We measure this response in dedicated measurements where we let the bolometers ample time to settle on the different flux

\footnote{In the presence of outliers I should have used the median rather than the mean.}
Figure 5: The NSD of the signal after we have simulated the averaging of all chopper cycles per dither position. From top to bottom I show the effect of using \( n = 3, 4, 5 \) in the \( n\sigma \)-clipping to reject glitches. This is obviously not applied to the cyan curves that correspond to deglitched signals to begin with. \( n = 4 \) provides already a reasonable rejection of glitches.
levels. In other words, the chopper frequency used in response measurements is significantly smaller than in point-source observations. For instance in the in-flight measurement we used from 8 to 64 images per chopper plateau to compute the response.

We know now that the time constant of the bolometers is of the same order of magnitude as the time interval between two readouts, and looking at Figure 7 of RD2, we see that on the blue side the time constants typically range from 35 to 55 ms and on the red side from 30 to 42 ms. The effect of these rather long time constant is that for a plateau of 4 10 Hz images, even though the chopper moves extremely fast (faster that 1/40 s) the first image is far from the final plateau level. Since this is not recognized by the task phtCleanPlateau (this is not a problem of chopper plateau), all 4 images are averaged thus decreasing the amplitude of the signal. For the sensitivity comparison, this is the same as working with an instrument that has a smaller response.

Using a simple low-pass filter, I have simulated the effect of these time constants on the response: I take a square wave sampled at 40 Hz with thus 16 readouts per plateau. I filter it and compute the average per plateau. I then compare the measured amplitude to the unfiltered one as a function of the ratio between the time constant and the sampling time interval (25 ms). I explore a range for this ratio that is consistent with what we have in PACS, i.e. from 1.2 to 2.2 and report in Table 3 the effective response for a chopping frequency of 1.25 Hz.

Table 3: The effective response in a 1.25 Hz chopped signal for various values of the time constant over sampling time ratio.

<table>
<thead>
<tr>
<th>$\tau/25$ ms</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>1.8</th>
<th>2.0</th>
<th>2.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{eff}$</td>
<td>0.90</td>
<td>0.88</td>
<td>0.86</td>
<td>0.83</td>
<td>0.81</td>
<td>0.78</td>
</tr>
</tbody>
</table>

This shows that in the point-source AOR we never have the full response of the bolometer. Using the peak of the time constant histogram as a representative value (see RD2) we see that in theory, the blue response is only 86% of the response that went in the NEP computation while the red response is 88% of the full response.

These numbers seem however smaller than the common experience of looking at point-source AOT signals. In Figure 6 and 7 I have made a comparison of signals measured during the chopping sequence of a point-source measurement for two pixels of the blue and red arrays (each time a representative and a slow one). On the right-hand side of these figures I have placed a simulation of the effect of a low-pass filter on a chopped signal with a time constant equal to that measured on the pixel. The simulated signal has 16 samples per plateau, and thus mimics the 40 Hz readout. The green crosses show the filtered signal averaged by groups of 4 images, so as to represent the 10 Hz blue crosses on the left-hand column.

It is quite evident that the first value of every plateau is closer to the plateau value in the simulation than in the real data while the data and simulation agree rather well when only the last part of the chopper plateau is concerned. This is probably pointing to the fact that a single low-pass filter is a very approximate representation of the detector behavior in frequency-space. This should be looked into in more details but is beyond the scope of this report.

The left-hand side columns of these figures also show the signal values after averaging all images per chopper plateau. This is where we see the dramatic effect of the finite response speed of the detectors: the average value is always significantly affected by the first image. On the selected pixels we observe that the ratio between the amplitude of the average values, and the maximum amplitude (generally given by the last image on the plateau) is 0.68 for pixel [8,8] blue, 0.54 for pixel [30,30] blue, 0.70 for pixel [10,6] red and 0.68 for pixel [4,24]. The representation of the pixels as low-pass filter thus systematically underestimates the loss on the response. Taking the observed ratio for the two representative pixels ([8,8] on the blue and [10,6] on the red, we thus expect that the finite response times of the pixels lead to the effective response in point source more being ~70% of the full response. This immediately increases the effective NEP by a factor 1.43.

There is a final caveat here: I am only computed the effect of using all of the frames in a plateau versus the peak one on the response. There is however no guarantee that during the chopper plateau we have time to reach the final value of the signal. The simulations of low-pass filtering on chopped data seem to indicate that in fact it is the case, however I’ve shown that this representation, as a low-pass filter, has its shortcomings. There could thus be further response losses introduced by the fact that for some pixel, a 1.25 Hz chopper frequency is
Figure 6: Comparison between the observed chopped signal and its prediction from a simple low-pass filter representation for the blue photometer. On the left column the brown line represents the chopper position as a function of time, the blue crosses the value of the individual 10 Hz samples for the selected pixels, and the cyan crosses, the plateau-averaged signal value. On the right column, the blue square line is the input signal, and the cyan line is the filtered signal. The x-axis is a sample index, and we have 16 samples per chopper plateau, as in the 40 Hz readout. The green crosses correspond to the filtered signal averaged 4 samples by 4 samples, to represent the 10 Hz signal.
Figure 7: Same as Figure 6 for the red photometer
simply too fast to let them reach the full signal amplitude. We will soon be able to test that however I point to the curves presented in RD2 where we do not see significant maximum amplitude losses at that frequency.

5 Conclusion

Combining the noise and response section we thus find that the NEP that we should have used in the sensitivity computation is $1.3/0.7$ times higher than what we have used on the blue side, or a factor of $1.9$ times higher. On the red side the effective NEP is $1.5/0.7$ times higher or $2.1$ times higher. Therefore, compared to our HSPOT expectations, we should be observing signal-to-noise ratio lower by a factor of $2$. 